Robust Autoencoders

By

Janki Mehta

Under the Supervision of

Dr. Angshul Majumdar

Submitted in partial fulfilment of the requirements for the degree of

Master of Technology

to

Indraprastha Institute of Information Technology, Delhi

May, 2016

Certificate

This is to certify that the thesis titled **"Robust Autoencoders"** being submitted by Janki Mehta to the Indraprastha Institute of Information Technology - Delhi, for the award of the Master of Technology, is an original research work carried out by her under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

April, 2016

Dr. Angshul Majumdar
Department of Electronics and Communication Engineering
Indraprastha Institute of Information Technology, Delhi
New Delhi – 110020

©Indraprastha Institute of Information Technology - Delhi (IIITD),New Delhi, 2016

Acknowledgements

This work would not have been possible without the guidance and support of several individuals who in one way or the other assisted in the preparation and completion of this work.

I would first like to express my sincere gratitude and respect towards my supervisor, Dr. Angshul Majumdar, under whose able and encouraging guidance, I was able to complete my thesis. He has constantly inspired me to do good work and has supported me all along the way.I owe special thanks Dr. Pankaj Jalote for making IIIT-Delhi such a wonderful place for research. I also am sincerely thankful to all the professors under whom I took different courses which helped me gain knowledge of tools and techniques used in my thesis work.

Finally, I must express my profound gratitude to my parents and to my family and friends for providing me with unfailing support and continuous encouragement throughout my years of study. This accomplishment would not have been possible without them.

Lastly, I would like to thank my batch mates and PhD students for providing such a good environment for fruitful discussions which were of immense help.

Thank you,
Janki Mehta

Abstract

An autoencoder is an artificial neural network used for learning efficient codings. The aim of an autoencoder is to learn a representation of data, which can then be used for better classification or any such application. Recently, the autoencoder is being used for learning generative models of data.

Usually the Euclidean distance (l_2-norm) is used as data fidelity constraint for training the autoencoder. The Euclidean distance is optimal when the noise is Normally distributed but not so in the presence of outliers.

In inverse problems like impulse denoising and de-aliasing, the noise/artifact is sparse but large. Hence a more robust data fidelity cost function is desired. So we use the l_p-norm asit makes the problem robust to outliers.Both linear and non-linear activation functions are used at the output end of autoencoder. The linear problem is solved using alternating minimization and Iterative Reweighted Least Squares (IRLS). The non-linear formulation is solved using split-Bregman technique. Experimental results show that our proposed method yields considerably better results for impulse denoising and de-aliasing compared to previous formulations.

Also,stacked linear autoencoder gives a representation for images which significantly improves the results for classification problem even using simple classification techniqueslike KNN or SVM. This performs much better than complex methods like deep belief networks or non-linear autoencoders.

Yet another application of this robust autoencoder is for inverse problems like signal recovery, especially for sparse signals. This produces results almost at par with compressed sensing techniques like Basis Pursuit, Lasso or Iterative Soft-thresholding and is also much faster.

Contents

List of Figures

List of Tables

Chapter-1

Introduction

1.1 Background

An autoencoder is an unsupervised neural network setting the target values to be equal to the inputs. Here, the encoder maps the input to a latent space, and the decoder maps the latent representation to the data.

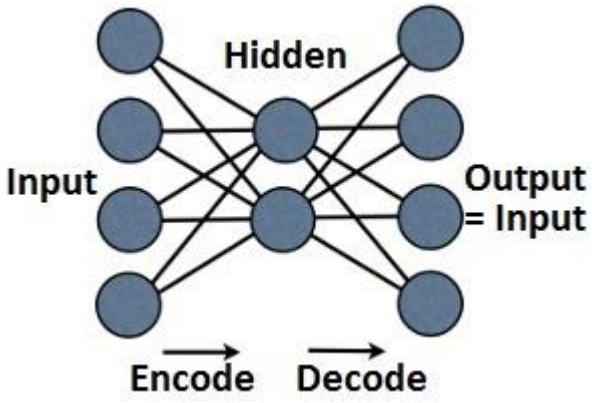

Fig. 1.1 Autoencoder

In signal processing terminology, we can say that an autoencoderlearns the analysis and the synthesis weights by minimizing the errorbetween the input (training samples / corrupted training samples) and the output(training samples). When corrupted training samples are given as input, it is called denoising autoencoder. A stacked autoencoder has more than one hidden layers and is trained by agreedy layerwise approach.

The error is calculated as l_2-norm and it is perhaps the most widely used data fidelity constraint in signal processing and machine learning. It arises from the Gaussian / Normal assumption of the distribution which fits a large class of problems in practice. But the practical reason behind popularity of the l_2-norm stems from the fact that it is easy to solve; it is smooth and convex and has a closed form solution (for linear problems).The l_2-norm minimization works when the deviations are small – approximately Normally distributed; but fail when there are large outliers.In statistics there is a large body of literature on robust estimation. The Huber function [1] has been in use for more than half a century in this respect. The Huber function is an approximation of the more recent absolute distance based

measures (l_1-norm). Recent studies in robust estimation prefer minimizing the l_1-norm instead of the Huber function [2]-[4]. The l_1-norm does not bloat the distance between the estimate and the outliers and hence is robust. The problem with minimizing the l_1-norm is computational. However, over the years various techniques have been developed. The earliest known method is based on Simplex [5]; Iterative Reweighted Least Squares [6] used to be another simple yet approximate technique. Other approaches include descent based method introduced by [7] and Maximum Likelihood approach [8].

1.2 Inverse Problems

A linear inverse problem is expressed in the following form:

$$y = Ax \tag{1}$$

If the system is orthogonal, solving (1) is trivial. One just requires applying the transpose of A to both sides of (1). If A is square (and full rank), one can solve (1) by finding the inverse of (1) or via Gaussian elimination.

In most practical problems, the system is not determined (square); it is either overdetermined or underdetermined. For the overdetermined case, there is no solution to (1) – but it is possible to find an approximate solution by minimizing the following least squares problem –

$$\hat{x} = \min_x \|y - Ax\|_2^2 \tag{2}$$

The solution to (2)turns out to be the pseudo-inverse of A.

$$\hat{x} = A^\dagger y = A^\dagger Ax \tag{3}$$

For the underdetermined case, there are infinitely many solutions. To search for a plausible solution, one needs some prior knowledge.For example, in the simplestcase one may be interested in the minimum energy solution, i.e.

$$\hat{x} = \min_x \|x\|_2^2 \text{ such that } y = Ax \tag{4}$$

This has a closed form solution –

$$\hat{x} = A^T \left(AA^T\right)^{-1} y \tag{5}$$

In recent years, after the advent of compressed sensing, researchers have become interested in sparse solutions of underdetermined systems. Such problems are ideally solved by minimizing the l_0-norm.

$$\hat{x} = \min_x \|x\|_0 \text{ such that } y = Ax \tag{6}$$

It is well known that solving (6) is an NP hard problem. There are two approaches to bypass this – a) greedy solution, b) relaxation of NP hard l_0-norm to nearest convex surrogate l_1-norm. Both are non-linear inversion techniques. They do not have analytic solutions and need to be solved iteratively.

Solving an inverse problem by l_1-norm or l_0-norm or l_p-norm are 'designed' inversion techniques. In this work instead of 'designing' such inversions, we propose to 'learn' the inversion. More specifically we learn the inversion using autoencoder.

1.2.1 Impulse Denoising

We look into two inverse problems. The first one is the simple denoising problem. This is expressed as

$$x = x_0 + n \tag{7}$$

where n is the noise.

If the noise is Gaussian, the solution can be recovered by least squares minimization. For sparse / impulse noise, one needs to minimize the more robust l_1-norm.

$$\hat{x} = \min_x \|x - x_0\|_1 \tag{8}$$

1.2.2 De-aliasing

The second problem we are interested in the de-aliasing problem. This arises in magnetic resonance imaging (MRI). In MRI, the data is acquired in the Fourier space. To accelerate the scan, the Fourier space is sub-sampled. This is expressed as,

$$y = RFx \tag{9}$$

where F is the Fourier transform, R is the sub-sampling operator, y is the collected data and x is the image.

If the Fourier space is fully sampled, inversion is straightforward. But for the said scenario a straightforward application of inverse FFT (after zero-filling the unsampled locations) would lead to aliasing (10).

$$\hat{x} = F^H R^T y \tag{10}$$

The aliasing artifacts are shown in Fig.1.2:

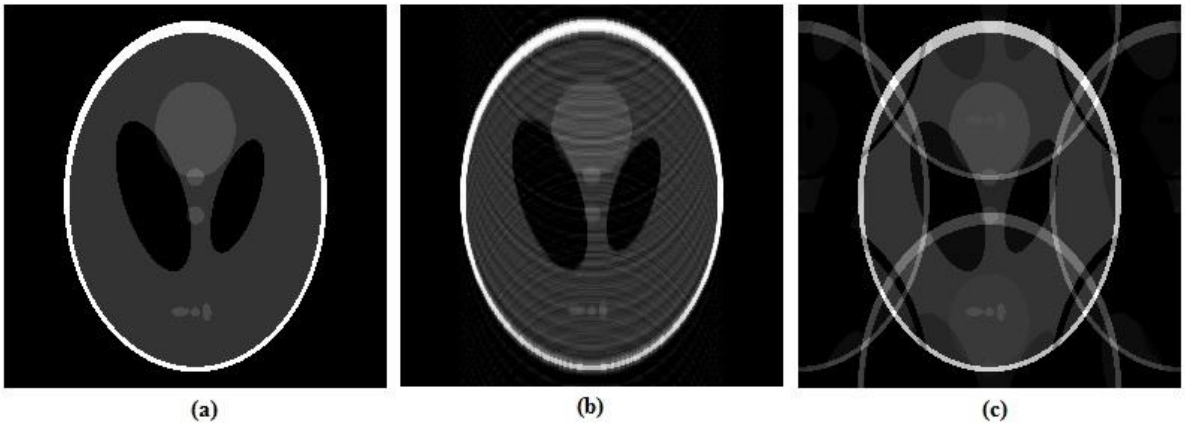

Fig. 1.2 (a) Original (b) Reconstruction from Variable Density Random Sampling (c) Reconstruction from Periodic Undersampling

Clearly the aliasing artifacts appear as outliers not as Gaussian noise. Compressed Sensing (CS) [9] formulates the inversion as an optimization problem of the following form,

$$\min_{x} \|Wx\|_0 \text{ subject to } y = RFx \tag{11}$$

where W is the wavelet transform.

This is an NP hard problem. It can be solved using greedy algorithms like Orthogonal Matching Pursuit (OMP) or by relaxing the NP hard l_0-norm to its nearest convex surrogate the l_1-norm.Our aim is to recover the de-aliased MRI, from this aliased one, as quick as possible.

1.3 Aim of the Thesis

In this work, we propose a generalized l_p-norm autoencoder, where the value of p can be adjusted to best fit the error in corrupt and clean signals.l_p-norm fidelity is robust to outliers. Neither impulse noise nor aliasing artifacts are Normally distributed. Hence, they cannot be solved using least squares. For impulse denoising, one needs to solve (6) and for de-aliasing, compressed sensing (CS) is employed. Both (6) and CS are non-linear recovery techniques. Instead of designing the inversion formulation as an optimization problem, we will learn the inversion operation using autoencoders. However the standard autoencoder formulation (discussed in the next section) uses an l_2-norm data fidelity term; this is not optimal for the said scenarios. Since the noise / artifact appear as outliers we need to robust version of autoencoder, i.e. we need to migrate to a robust l_1-norm data fidelity term instead of the standard Euclidean cost function.

We use a simple Iterative Re-weighted Least Squares (IRLS) based technique to solve the l_p-norm minimization problem for autoencoders with linear activation function.We also have solved non-linear case using split-Bregman technique and soft thresholding.

We test both our proposed formulationof autoencoder for classification as well. The features generated by our robust autoencoder are used as input to simple classification algorithms like KNN, SVM, etc. We found that even with such simple classifiers, our features give significant improvement in classification accuracy over deep autoencoder or DBN.

This work contributes to the body of machine learning and signal processing in two different aspects. First, this is the only work that introduces robust autoencoders. Second, there is hardly any work on solving inverse problems using autoencoders; most studies use autoencoders for learning features to be used in classification problems. This would be one of the foremost studies that show how autoencoders can 'learn' to solve inverse problems.

1.4 Organization of the Thesis

This thesis is organized as follows:

- Chapter-2 presents the literary survey on the basics of autoencoders, theirvariations and its connection to the dictionary learning.
- Chapter-3 proposes formulation and algorithm for linear autoencoder.
- Chapter-4 proposes formulation and algorithm for non-linear autoencoder.
- Chapter-5 explains the experimental setup, datasets used and gives results and comparisons of proposed method against the existing ones.
- Chapter-6concludes this thesis by summarizing the contribution of the work and research done.

Chapter-2

Literature Review

2.1 Autoencoders

An autoencoder consists of two parts – the encoder maps the input to a latent space, and the decoder maps the latent representation to the data [10, 11]. For a given input vector (including the bias term) x, the latent space is expressed as:

$$h = Wx \tag{12}$$

Here the rows of W are the link weights from all the input nodes to the corresponding latent node. The mapping can be linear [12], but in most cases it is non-linear. Usually a sigmoid function is used, leading to:

$$h = \phi(Wx) \tag{13}$$

The sigmoid function shrinks the input (from the real space) to values between 0 and 1. Other non-linear activation functions (like tanh) can be used as well.

The decoder portion reverse maps the latent variables to the data space.

$$x = W'\phi(Wx) \tag{14}$$

Since the data space is assumed to be the space of real numbers, there is no sigmoidal function here.

During training, the problem is to learn the encoding and decoding weights $-W$ and W'. In terms of signalprocessing lingo, W is the analysis operator and W is the synthesis operator. These are learnt by minimizing the l_2- norm data fidelity constraint:

$$\arg\min_{W,W'} \left\| X - W'\phi(WX) \right\|_F^2 \tag{15}$$

Here $X = \left[x_1 \mid \ldots \mid x_N \right]$ consists all the training sampled stacked as columns. The problem (15) is clearly non-convex. But can be solved by gradient descent techniques since the sigmoid function is smooth and continuously differentiable.

Denoising autoencoders [12, 13] are a variant of the basic autoencoder where the input consists of noisy samples and the output consists of clean samples. Here the encoder and decoder are learnt to denoise noisy input samples.

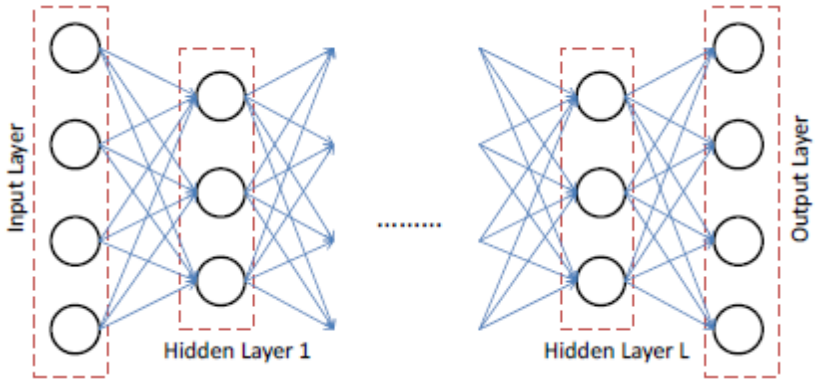

Fig. 2.1 Stacked Autoencoder

There are several extensions to the basic autoencoder architecture. Stacked / Deep autoencoders [14] have multiple hidden layers (see Fig. 2.1). The corresponding cost function is expressed as follows:

$$\begin{aligned} &\underset{W_1,\ldots,W_{L-1},W'_1,\ldots,W'_{L-1}}{\arg\min} \left\| X - g \circ f(X) \right\|_F^2 \\ &\textit{where } g = W_1'\phi\left(\left(W_2'\ldots W_L'\left(f(X)\right)\right)\right) \\ &\textit{and } f = \phi\left(W_{L-1}\phi\left(W_{L-2}\ldots\phi\left(W_1 X\right)\right)\right) \end{aligned} \tag{16}$$

Solving the complete problem in(16) is computationally challenging. The weights are usually learned in a greedy fashion – one layer at a time [15], i.e. the outermost weights (layer 1) are first learnt. The features (product of the learned weights multiplied with the input) are used to learn the second and deeper layers.

Stacked denoising autoencoders [13] are a variant of the basic autoencoder where the input consists of noisy samples and the output consists of clean samples. They too have multiple hidden layers which are learnt in a greedy fashion as explained above.

Another variation for the basic autoencoder is to regularize it, i.e.

$$\underset{(W)s}{\arg\min} \left\| X - g \circ f(X) \right\|_F^2 + R(W, X) \tag{17}$$

The regularization can be a simple Tikhonov regularization – however that is not used in practice. It can be a sparsity promoting term [16] or a weight decay term (Frobenius norm of the Jacobian) as used in the contractive autoencoder [17]. The regularization term is usually chosen so that they are differentiable and hence minimized using gradient descent techniques.

2.2 Connection to Dictionary Learning

The KSVD [18] popularized dictionary learning in signal processing and machine learning. It has been used in almost every perceivable image processing scenario. The original KSVD formulation learns a dictionary that can represent the training data in a sparse fashion. The model is $X = D_S Z$ where X is the data, D_S is the learnt dictionary and Z is the sparse coefficients.In general dictionary learning can be expressed as,

$$\min_{D_s, Z} \left\| X - D_s Z \right\|_F^2 + \lambda \left\| Z \right\|_1 \tag{18}$$

The problem (18) is solved by alternately updating DS (dictionary / codebook) and Z (sparse code).

This is the so called synthesis prior formulation where the task is to find a dictionary that will synthesize / generate signals from sparse coefficients. There is an alternate co-sparse analysis prior dictionary learning paradigm [19] where the goal is to learn a dictionary such that when it is applied on the data the resulting coefficient is sparse.The model is $D_A \hat{X} = Z$. The corresponding learning problem is framed as:

$$\min_{D_A, \hat{X}} \left\| X - \hat{X} \right\|_F^2 + \lambda \left\| D_A \hat{X} \right\|_1 \tag{19}$$

The autoencoder learns the analysis and the synthesis dictionaries simultaneously. The learning technique for a single layer autoencoder is (20); we repeat it for the sake of convenience.

$$\arg\min_{W, W'} \left\| X - W' \phi\left(WX \right) \right\|_F^2 \tag{20}$$

It does not specifically look for a sparse solution (hence there is no sparsity promoting term). For the linear activation function, it is the same as jointly learning the analysis and the synthesis dictionaries.

Chapter-3

Linear Robust Autoencoder

We do not change the autoencoder architecture. We only change the data fidelity constraint from l_2-norm to l_p-norm. Several studies [20, 21] have indicated that despite the non-linearity defining the relationship, hidden layerneurons essentially function in the linear region. Following the analysis and results reported in these works, we model our autoencoder using simple linear mapping; this reduces the computational costs considerably. For a single layer autoencoder this results in solution to the following problem for finding the weights.

$$\arg\min_{W,W'} \left\| X - W'WX \right\|_p^p \tag{21}$$

It is possible to solve this problem using sophisticated techniques like variable splitting and Augmented Lagrangian (split Bregman), but this would introduce several hyper-parameters which need to be tuned. Empirically tuning these for large scale machine learning problems is cumbersome and time consuming. Therefore, we adopt a simple Iterative Reweighted Least Squares (IRLS) based technique to solve the l_p-minimization problem (10).

3.1 Linear Inverse Problem: l_p-norm Data Fidelity

IRLS is a popular technique in sparse recovery [22]. It has been used to solve linear inverse problems of the form:

$$y = Ax + \eta \tag{22}$$

where y is observation, A is the measurement operator, x is the solution and η is some non-Gaussian noise.

The task is to solve the following l_p-minimization problem:

$$\arg\min_{x} \left\| y - Ax \right\|_p^p \tag{23}$$

In IRLS, the l_p-norm is substituted by a weighted l_2-norm.

$$\begin{aligned} &\sum_i |v_i|^p = \|v\|_p^p = \|Wv\|_2^2 \\ &where\ \ w_i = |v_i|^{\frac{p}{2}-1}, W = diag(w) \end{aligned} \tag{24}$$

For solving (23), the IRLS formulation is:

$$\arg\min_{x} \left\| W(y - Ax) \right\|_2^2$$
$$where \;\; w_i = \left| y_i - (Ax)_i \right|^{\frac{p}{2}-1} \tag{25}$$

The solution progresses iteratively. In the first iteration *W=Identity*. In the subsequent iterations, *W* is updated based on the previous solution (*x*). At convergence, (25) is the same as (23); the IRLS reaches the desired solution asymptotically.

The IRLS technique has been also used profusely for sparse recovery problems [23, 24] where the demand was to minimize the following:

$$\arg\min_{x} \left\| x \right\|_p^p \;\; such \;\; that \; y = Ax \tag{26}$$

3.2 l_p-norm to weighted l_2-norm Autoencoder

Here the objective function is replaced by the weightedl_2-norm and is solved iteratively. A formal derivation for the IRLS is given in [24] for solving sparse recovery.

In this work, we recast the l_p-norm autoencoder as a reweighted least squares problem, i.e.

$$\arg\min_{W,W'} \left\| X - W'WX \right\|_p^p = \arg\min_{W,W'} \left\| D \odot (X - W'WX) \right\|_F^2 \tag{27}$$

where $d_i = \left| \left(X - W'WX \right)_i \right|^{\frac{p}{2}-1}$, D is a matrix defined by d_i at every point i and $\odot$ implies element-wise product.

In every outer loop, the matrix D is computed based on the values of *W, W'*. The inner loop solves the weighted least squares problem. In the following sub-section we will outline the derivation for solving (27).

3.3 Solving weighted l_2-norm Autoencoder

In this section, we derive the algorithm to solve (27). We repeat it for the sake of convenience:

$$\arg\min_{W,W'} \left\| D \odot (X - W'WX) \right\|_F^2 \tag{28}$$

This can be also expressed as,

$$\arg\min_{W,W'} \left\| D^T \odot \left(X^T - X^T W^T W'^T \right) \right\|_F^2$$
$$= \arg\min_{W,W'} \left\| Y - YW^T W'^T \right\|_F^2, \; Y = D^T \odot X^T \tag{29}$$

Now we employ a Majorization Minimization (MM) technique [25] to simplify the expression.

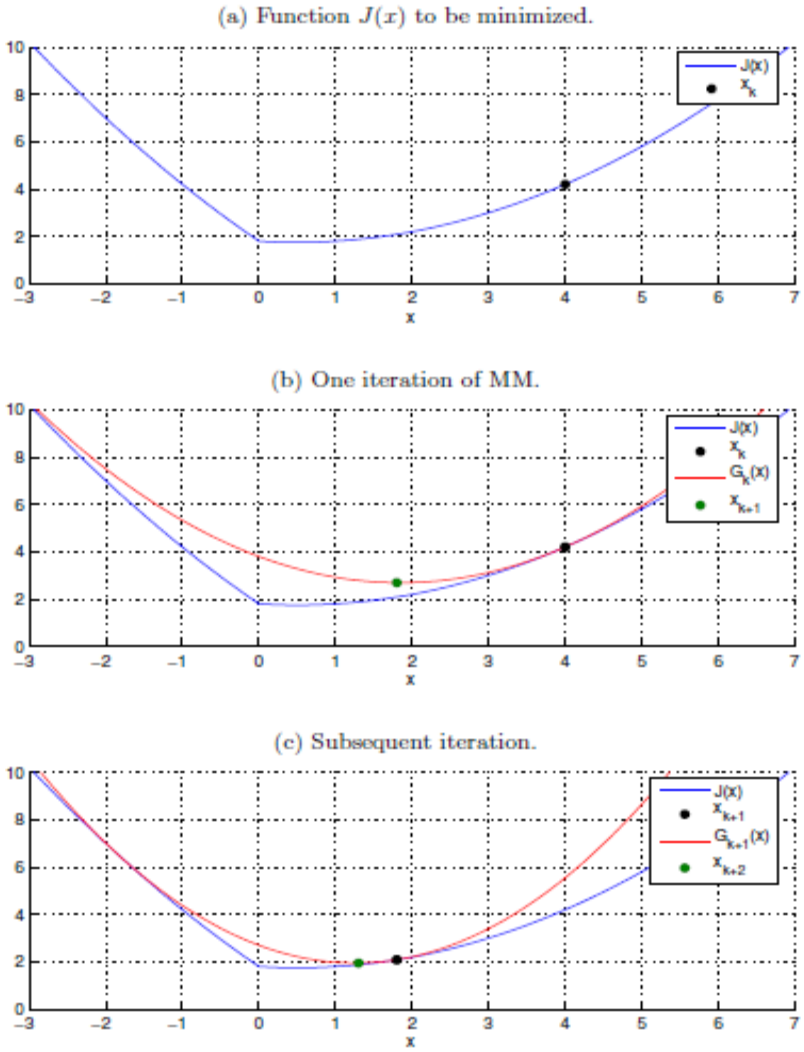

Fig. 3.1 Majorization Minimization [25]

We can simplifyevery iteration of (29)using MM. Let,

$$J = \|Y - YZ\|_F^2 \text{ , where } Z = W^T W^{'T} \tag{30}$$

For this minimization, $G_k$is chosen to be,

$$G_k = \|Y - YZ\|_F^2 + (Z - Z_k)^T (aI - Y^T Y)(Z - Z_k) \tag{31}$$

where a is the maximum eigenvalue of the matrix YY^T-this ensures that the second term is positive definite and hence $G_k$is indeed a majorizer of J.

$$G_k = \|Y - YZ\|_F^2 + (Z - Z_k)^T \left(aI - Y^T Y\right)(Z - Z_k)$$
$$= Y^T Y - 2Y^T YZ + Z^T Y^T YZ + (Z - Z_k)^T \left(aI - Y^T Y\right)(Z - Z_k)$$
$$= Y^T Y + Z_k^T \left(aI - Y^T Y\right) Z_k - 2(Y^T Y + Z_k^T \left(aI - Y^T Y\right))Z + aZ^T Z$$
$$a(-2B^T Z - Z^T Z) + C$$
$$where\ B = Z_k + \frac{1}{a} Y^T (Y - YZ_k).$$
$$C = Y^T Y + Z_k^T (aI - Y^T Y) Z_k$$

Using the identity $\|B - Z\|_F^2 = B^T B - 2B^T Z + Z^T Z$, one can write,

$$G_k = a\|B - Z\|_F^2 - aB^T B + C \tag{32}$$

Therefore, minimizing (32)is the same as minimizing the following,

$$G_k = \|B - Z\|_F^2 \tag{33}$$

since the other terms are devoid of *Z*.

Substituting *Z* back in (33), we reach the following optimization problem,

$$\begin{aligned} &\arg\min_{W,W'} \left\|B - W^T W'^T\right\|_F^2 \\ &where\ B = \left(W^T W\right)_k^T + \frac{1}{a} Y^T (Y - Y(W'W)_k) \end{aligned} \tag{34}$$

Solving (34)is straightforward – it is a matrix factorization problem. We solve it using alternating least squares.

In very iteration (say k)

$$W_k = \arg\min_W \left\|B - W^T W'^T_{k-1}\right\|_F^2 \tag{35}$$

$$W'_k = \arg\min_{W'} \left\|B - W_k^T W'^T\right\|_F^2 \tag{36}$$

The full algorithm for solving the l_pAutoencoder can be succinctly represented in a few steps

Initialize: $D = Identity$

Outer loop

Inner loop (*say* k)

Define: $Y = D^T \square X^T$

Compute: $B = \left(W'W\right)_{k-1}^T + \frac{1}{a} Y^T \left(Y - Y\left(W'W\right)_{k-1}\right)$

Solve W: $W_k = \arg\min_W \left\| B - W^T W'^T_{k-1} \right\|_F^2$

Solve W': $W'_k = \arg\min_{W'} \left\| B - W_k^T W'^T \right\|_F^2$

repeat inner loop

Compute D *from* $d_i = \left| \left(X - W'WX \right)_i \right|^{\frac{p}{2}-1}$

repeat outer loop

Chapter-4

Non-linear Robust Autoencoder

In the previous section, we focussed on the autoencoder with linear activation function. In this section we solve the denoising autoencoder with non-linear activation function like sigmoid or tanh. Below is the formulation we intend to solve:

$$\arg\min_{W,W'} \left\| X - W'\phi(WX') \right\|_p^p \tag{37}$$

where X' is the corrupted set of training samples.

This is a non-smooth problem; therefore the age-old backpropagation algorithm will not work directly. We substitute Z= $X - W'\phi(WX')$, converting the equation (37)to (38) by split Bregman technique as explained in the following sub-section:

$$\arg\min_{Z,W,W'} \left(\left\| Z \right\|_p^p + \mu \left\| Z - \left(X - W'\phi(WX') - B \right) \right\|_F^2 \right) \tag{38}$$

where*B* is the Bregman's constant.

4.1 Brief overview of split Bregman technique

Here we briefly review the spilt Bregman technique. Bregman distance forms the basis for formulation of thesealgorithms. For a convex function $E : X \to R$, where *u,v*belongs to*X* and *p*belongs to the set of sub gradient of the function, Bregman distanceis given by [32]

$$D_E^p(u,v) = E(u) - E(v) - \langle p, u - v \rangle \tag{39}$$

Consider the objective function given in (40) where Φ(*u*) and *H(u)*are convex and *H* is differentiable.

$$\min \left| \Phi(u) \right|_1 + H(u) \tag{40}$$

Split Bregman technique focuses on decomposing a complex optimization problem (with multiple norm terms) such that theyform different sub problems which can be solved more easily thanthe original composite objective function[33]. Rewriting (40) by letting*d*=Φ(*u*) we get,

$$\begin{aligned} &\min_u \left| d \right|_1 + H(u) \\ &subject\ to\ d = \Phi(u) \end{aligned} \tag{41}$$

The unconstrained equivalent of (41) is obtained by adding apenalization function to the problem as in (42).

$$\min_{u} |d|_1 + H(u) + \frac{\lambda}{2}\|d - \Phi(u)\|_2^2 \tag{42}$$

Equation (42) can be solved as follows:

$$\begin{aligned} u^{k+1} &= \min_{u} H(u) + \frac{\lambda}{2}\|d - \Phi(u) - b^k\|_2^2 \\ d^{k+1} &= \min_{d} |d|_1 + \frac{\lambda}{2}\|d - \Phi(u) - b^k\|_2^2 \\ b^{k+1} &= b^k + \left(\Phi(u^{k+1}) - d^{k+1}\right) \end{aligned} \tag{43}$$

Since *H(u)* is smooth and differentiable everywhere, update for *u* can be solved analytically. Solution for *d* is nothing but the solution for synthesis prior formulation and is obtained directly by shrinkage (soft thresholding) operator. Last step is the update of Bregman variable.

Use of split Bregman technique aids in faster convergence and lower recovery errors, as no cooling of regularization parameter is required and thus optimal values of regularization parameters for each of the sub problem can be set [33].

4.2 Solving non-linear l_p-norm autoencoder

Now we solve equation (38) using soft-thresholding and alternating least squares.

First we solve(38) for *Z*. Rearranging terms in (38), we get,

$$\arg\min_{Z}\left(\|Z\|_p^p + \mu\|(X + B - W'\phi(WX')) - Z\|_F^2\right)$$

$$\therefore Z = softTH(X + B - W'\phi(WX'), \frac{\mu}{2} p|Z|^{p-1}) \tag{44}$$

$$\left[soft(x,T) := sign(x)\max(0, |x| - T)\right]$$

Now, for *W'* and *W*, we just need to solve second term of (38).

$$\arg\min_{W,W'}\left(\|(X + B - Z) - W'\phi(WX')\|_F^2\right) \tag{45}$$

Putting $Y = X + B - Z$ and $R = \phi(WX')$, we get

$$\arg\min_{W',R}\left(\|Y - W'R\|_F^2\right) \tag{46}$$

This turns out to be a tractable problem that can be solved using alternating minimization.

$$W' \leftarrow \arg\min_{W'} \left(\|Y - W'R\|_F^2 \right) \tag{47}$$

$$R \leftarrow \arg\min_{R} \left(\|Y - W'R\|_F^2 \right) \tag{48}$$

From, $R = \phi(WX')$, we get $W = \phi^{-1}(R) / X'$.

The complete algorithm can be succinctly represented as follows:

Initialize: $Z = U(0,1),\ B = 0$
Outer loop
  Define: $Y = X + B - Z$
  Inner loop (*say* k)
    Define: $R = \phi(WX')$
    Solve: $R \leftarrow \arg\min_{R} \left(\|Y - W'R\|_F^2 \right)$
    Solve: $W' \leftarrow \arg\min_{W'} \left(\|Y - W'R\|_F^2 \right)$
    Compute: $W = \phi^{-1}(R) * X'^{\dagger}$
  repeat inner loop
  Update: $Z = softTH(X + B - W'\phi(WX'), \frac{\mu}{2} p|Z|^{p-1})$
  Update: $B = B + (X - W'\phi(WX')) - Z$
repeat outer loop

Chapter-5

Experiments and Results

5.1 Inverse Problems

5.1.1 Dataset

We show the denoising and de-aliasing performance on the CIFAR-10 dataset. The 50,000 training images were used to learn the autoencoder and the remaining 10,000 test images were used for testing denoising/de-aliasing performance. The color images were converted to grayscale and the pixel values were normalized between 0 and 1. The sample gray images are shown in the figure below.

Fig. 5.1 Cifar-10 Dataset

Some of the sample noisy and aliased images are as shown below

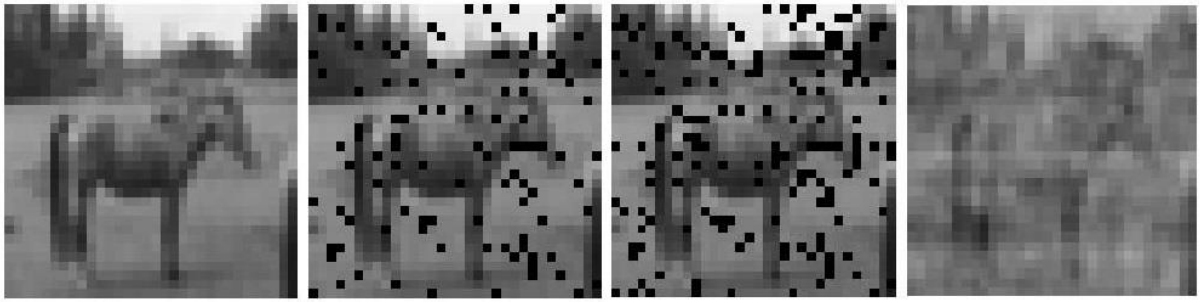

Fig. 5.2(a) Original Image, (b) 10% Impulse Noise, (c) 15% Impulse Noise, (d) Randomly Aliased Image

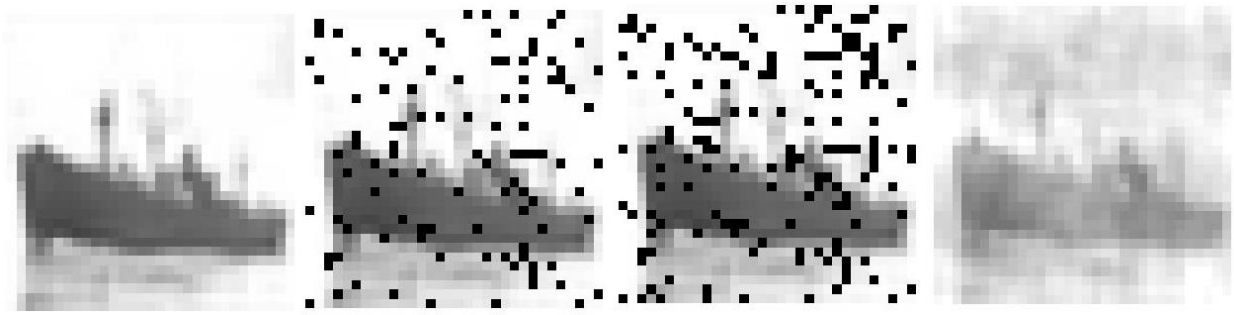

Fig. 5.3(a) Original Image, (b) 10% Impulse Noise, (c) 15% Impulse Noise, (d) Randomly Aliased Image

5.1.2 Experimental Setup

We polluted the Cifar-10 images with 10% and 15% impulse noise as shown in fig. 5.2 and 5.3. The aliased images are made by randomly sub-sampling the image by 50% in the Fourier space - a realistic sampling scenario in MRI reconstruction.The sub-sampled positions are zero-filled and reconstructed by applying inverse FFT.All the images are 32x32 pixels in size, which are converted to vectors of length 1024. So net, we have two sets of 50,000 such vectors; one clean and one corrupted. These are used for training the robust autoencoder; corrupted ones being the input and the clean ones being the output. We tried experiments with 512 and 768 hidden nodes in the first layer. We haven't gone any deeper as in neural network literature, that a single hidden layer can approximate arbitrary functions [26].Going deeper does not help in functional approximation capacity; going deeper with greedy learning is a trick to prevent over-fitting; when the number of training samples are limited learning too many parameters cause over-fitting – in such a case instead of learning a large number of

nodes in the hidden layer, one goes deeper and learns greedily; so that in each layer relatively smaller number of parameters need to be learned.

The value of *p*and number of hidden nodes is a variable parameter and is chosen as per the optimality of the result.

5.1.3 Results and Comparison

We compare the results with the standard denoising autoencoder (DAE) [13] and with the sparse denoising autoencoder (SDAE) [16]. Comparison was also carried out with a CS technique, Spectral Projected Gradient for l_1 minimization (SPGL1) [27], used for basis pursuit denoising. The non-linear robust autoencoder was compared against non-linear deep autoencoder [28].

All the prior studies were based on minimizing the *l2*- norm; and reported results in terms of mean squared error (MSE). Quiet obviously the MSE was low. However, it is well known in image processing literature that MSE is not a useful metric for quality assessment. One can progressively blur the image and improve MSE; yielding a high MSE but poor quality image. Peak Signal to Noise Ratio (PSNR) has been more popular conventionally; in recent years the commonly used metric has been Structural Similarity Index (SSIM) [29]; SSIM is by far the most well-known metric for image quality assessment and reflects the quality assessment of the human visual system. In this work, we will report the results in terms of PSNR and SSIM.

The results show that our method yields significantly superior results compared to other autoencoders and state-of-the art compressed sensing based methods.

Table 5.1.1 Linear AE Results

Dataset	DAE		SDAE		$SPGL_1$		Robust Linear AE	
	PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
10% noise	22.6486	0.6713	22.3000	0.6600	**33.1729**	**0.97082**	23.2323	0.6951
15% noise	22.8295	0.6842	22.251	0.6806	**30.8216**	**0.9530**	23.5031	0.7109
50% sub-sampled aliasing	20.6218	0.5775	20.2935	0.7084	19.2538	0.6632	**22.6276**	**0.7445**

Here the number of hidden nodes for denoising is 512 and for de-aliasing it is 768.

Table 5.1.2 Non-Linear AE Results

Dataset	Deep Non-linear AE		$SPGL_1$		Robust Non-Linear AE	
	PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
10% noise	23.8313	0.7509	33.1729	0.97082	**39.3837**	**0.9903**
15% noise	23.5916	0.7379	30.8216	0.9530	**37.5349**	**0.9868**
50% sub-sampled aliasing	20.7273	0.5898	19.2538	0.6632	**23.3575**	**0.7674**

Here the number of hidden nodes is 768 in all cases. Also value of p is taken to be 1 for all above results. Also tanh is used as the non-linear activation function.

Below are some images displaying the denoising and de-aliasing on Cifar-10 dataset:

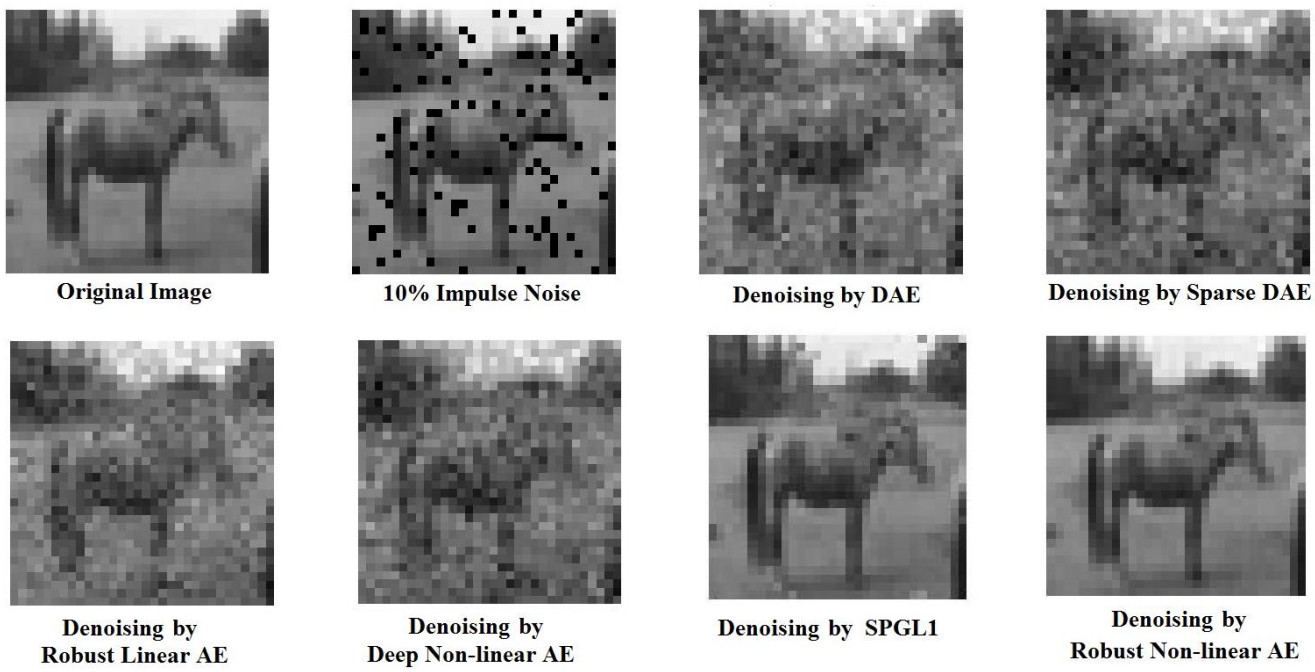

Fig. 5.4 10% Impulse Denoising Results

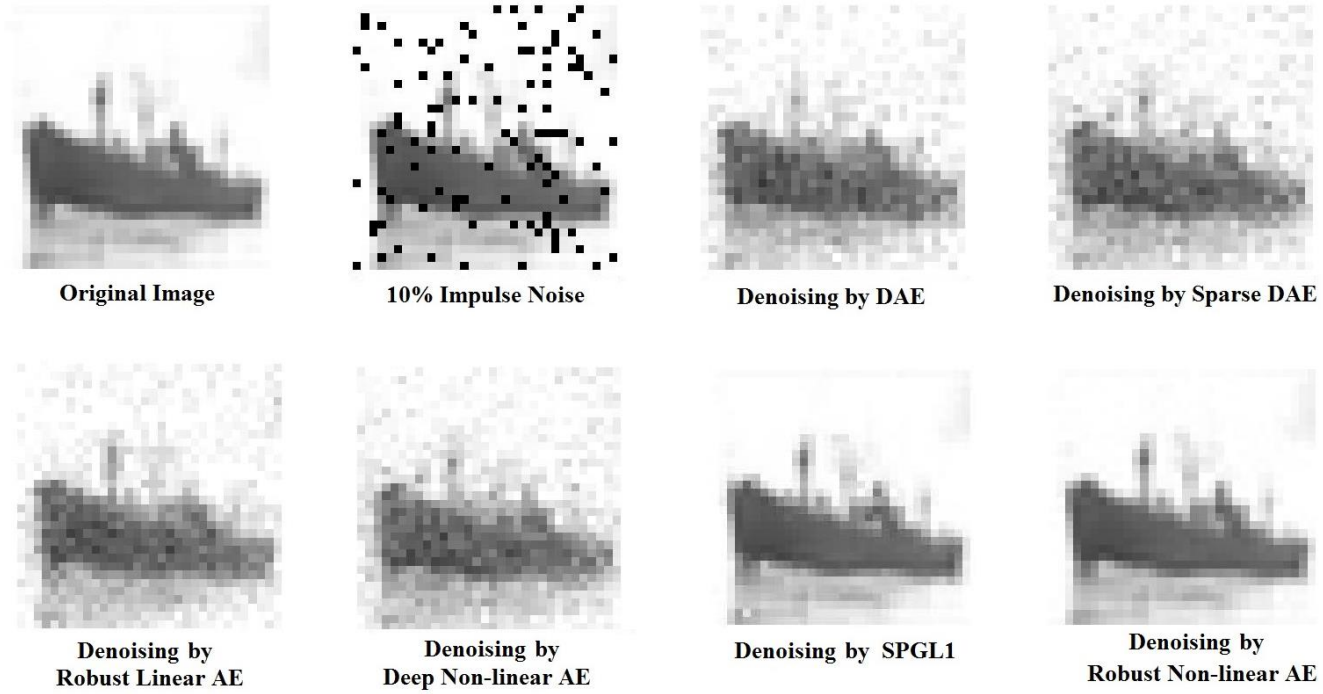

Fig. 5.5 10% Impulse Denoising Results

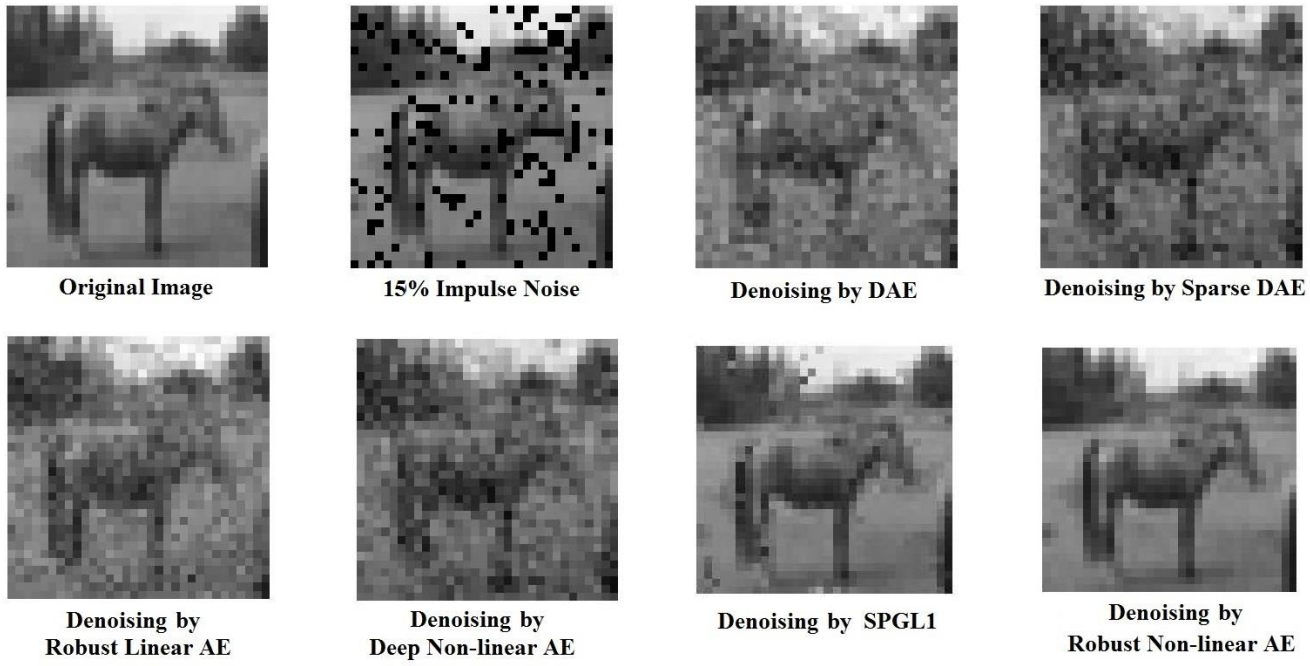

Fig. 5.6 15% Impulse Denoising Results

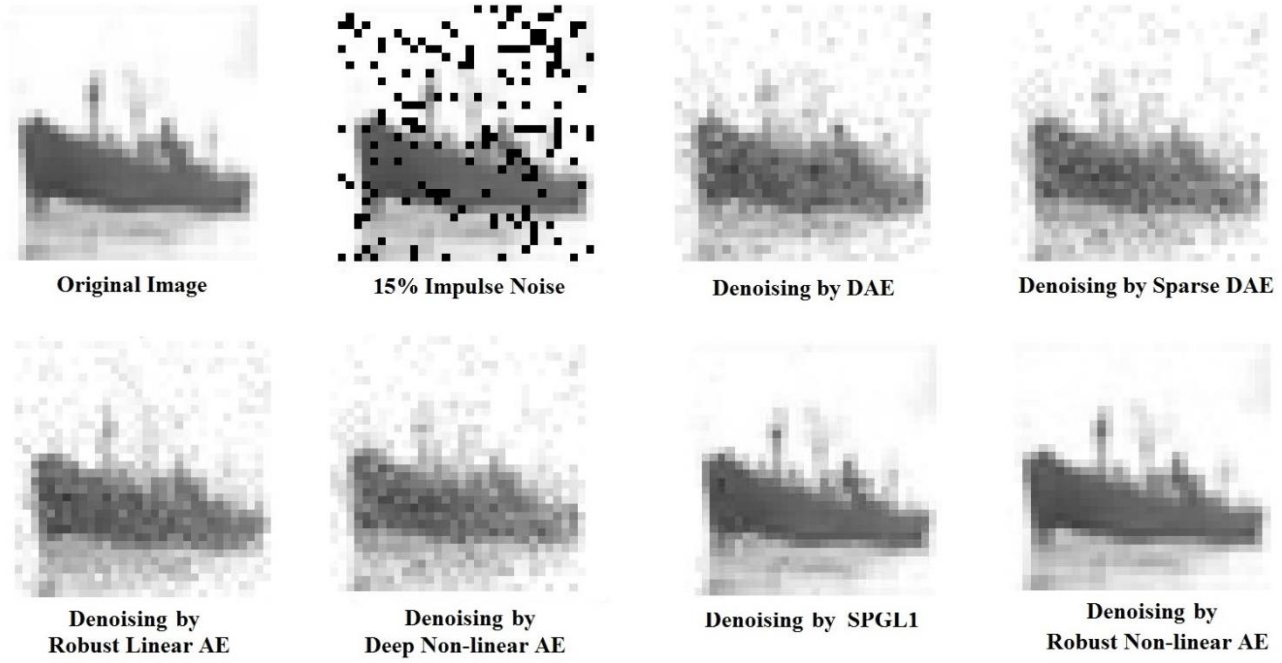

Fig. 5.7 15% Impulse Denoising Results

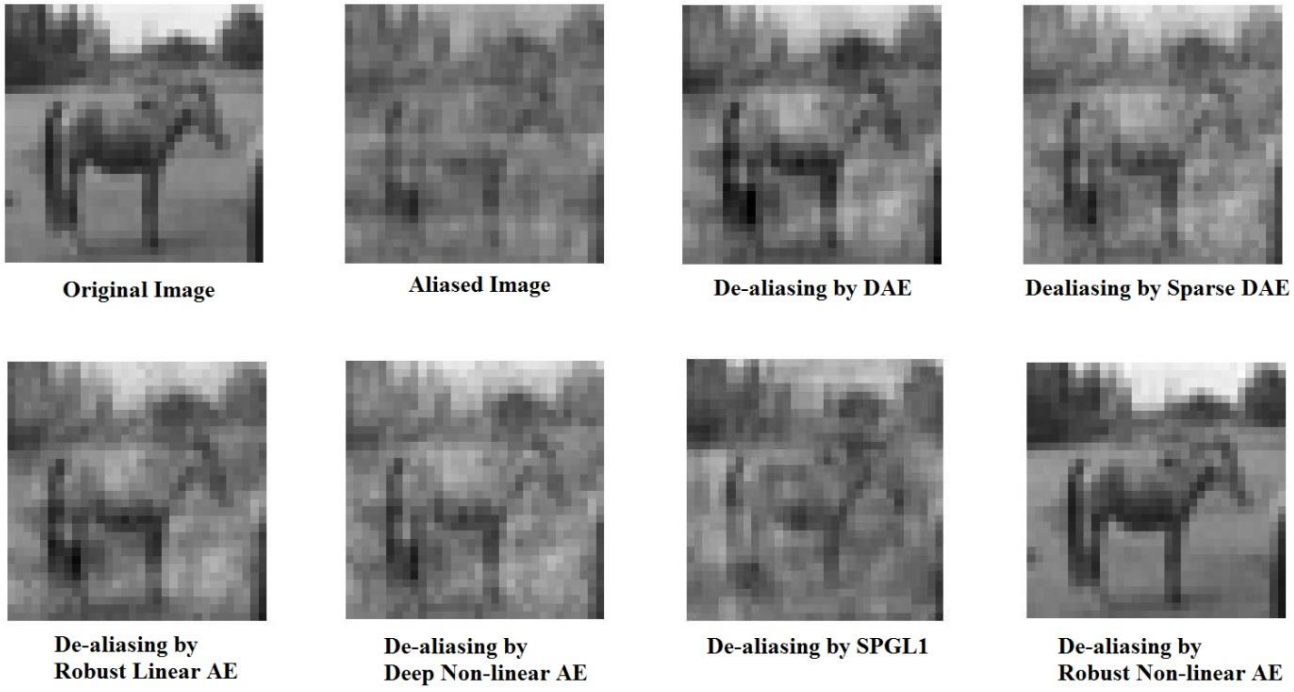

Fig. 5.8 De-aliasing Results

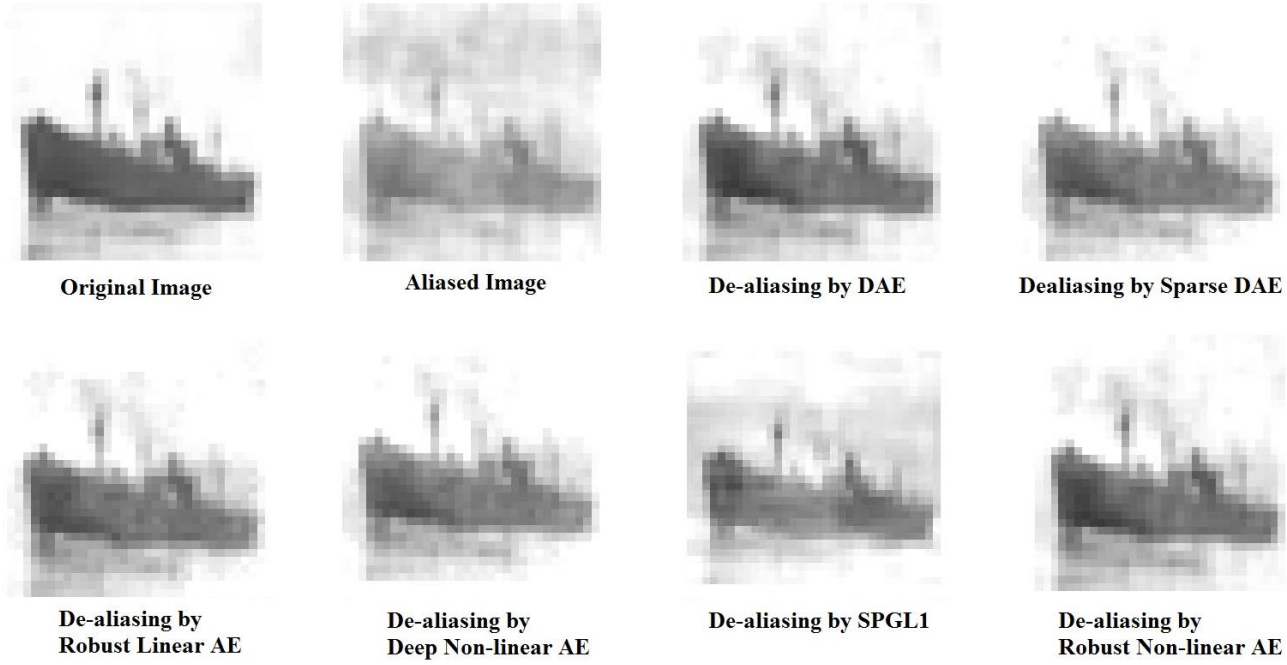

Fig. 5.9 De-aliasing Results

5.1.4 Sparse Recovery

A sparse signal x can be recovered from the underdetermined system $y = Ax$ using robust denoising autoencoder trained with $x' = A^T y = A^T Ax$ (i.e. poor man's inverse) as input and x as output. For training the autoencoder, 5000 samples of randomly generated signal x of length 100 with 5% sparsity were taken and for testing, 1000 samples were used.

Below is the comparison of accuracy and run-time for recovery of sparse signal x using trained autoencoder and SPGL1.

Table 5.1.3 Sparse recovery Results

Technique	Error	Run-Time (s)
Robust Autoencoder	1.7×10^{-2}	0.005
SPGL1	2.7×10^{-5}	30

Even the time taken for training this autoencoder was about 17 seconds. This can be very useful for real time signal recovery in image guided surgery where there is a need for de-alising MRI signals instantly.

5.2 Classification Problem

5.2.1 Dataset

We carried our experiments on several benchmarks datasets. The first one is MNIST dataset which consists of 28x28 images of handwritten digits ranging from 0 to 9. The dataset has 60,000 images for training and 10,000 images for testing. No pre-processing has been done on this dataset.

The other datasets are variations of MNIST, which are more challenging primarily because they have fewer training samples (10,000) and larger number of test samples (50,000).

1. basic (smaller subset of MNIST)
2. basic-rot (smaller subset with random rotations)
3. bg-rand (smaller subset with uniformly distributed noise in background)
4. bg-img (smaller subset with random image background)
5. bg-img-rot (smaller subset with random image background plus rotation)

Samples for each of the datasets are shown in Fig. 5.10.

Fig. 5.10 Top to bottom; basic, basic-rot, bg-rand, bg-img, bgimg-erand

5.2.2 Linear vs. Non-linear Autoencoder

We saw that at least for the benchmark datasets used in these experiments, the simple linear (Identity) activation function yields better classification accuracy than their non-linear (sigmoid) counterpart.

Linear autoencoder: $\min_{W',W} \|X - W'WX\|_F^2$

Non-linear autoencoder: $\min_{W',W} \|X - W'\phi(WX)\|_F^2$

The autoencoder architectures remain same otherwise; both (linear and non-linear) are three layer architectures with 392-196-98 hidden nodes. The implementation of the non-linear autoencoder is from [28]. The representation from the deepest layer is used for classification. We employ two non-parametric classifiers – KNN and Sparse Representation based Classification (SRC) [30]. We want to test the representation / feature extraction capability of the linear and non-linear autoencoders; this is best done using simple non-parametric classifiers. Parametric classifiers like Neural Network and SVM may be fine-tunedto yield better results, but in such a case it is difficult to gauge if the improvement in results is owing to the feature extraction or owing to the fine tuning.

The results (Table 5.2.1) show that the linear one always yields better results. The improvement is small when the number of training samples is larger (MNIST) but for the more challenging datasets, the linear autoencoder improves by a large margin.

Table 5.2.1 Linear vs. Non-linear Autoencoder

Datasets	KNN		SRC	
	Linear	**Non-Linear**	**Linear**	**Non-Linear**
MNIST	**97.33**	96.11	**98.33**	97.29
Basic	**95.25**	94.86	**96.91**	96.43
basic-rot	**84.83**	80.71	**90.04**	84.29
bg-img	**77.16**	70.97	**84.14**	76.94
bg-rand	**86.42**	81.11	**91.03**	85.49
bg-img-rot	**52.21**	44.60	**62.46**	62.46

We show that with linear autoencoder, the implementation is significantly more efficient than the non-linear one – it is about an order of magnitude (10 times) faster. The training times are shown in Table 5.2.2. Both the linear and non-linear stacked autoencoders were run till

convergence. For comparison the training time of a deep belief network is also shown. The experiments were run on a PC running on Intel i7 3.1 GHz on 16 GB RAM. The OS is Windows 7 64 bit. The environment is Matlab 2012a.

Table 4.2.2Training time

Algorithm	Run Time (s)
Linear SAE	13113
Non-Linear SAE	120408
DBN	30071

5.2.3 Results for Robust Autoencoder

Since we have established that the linear autoencoder indeed yields better results for classification, we will use it in the rest of the paper. We show that by using the l_p-autoencoder, we can improve upon the standard linear autoencoder (l_2-norm). The number of nodes remains as before. The value of p is varied to obtain the best result. Below is the graph for change in accuracy over different values of p:

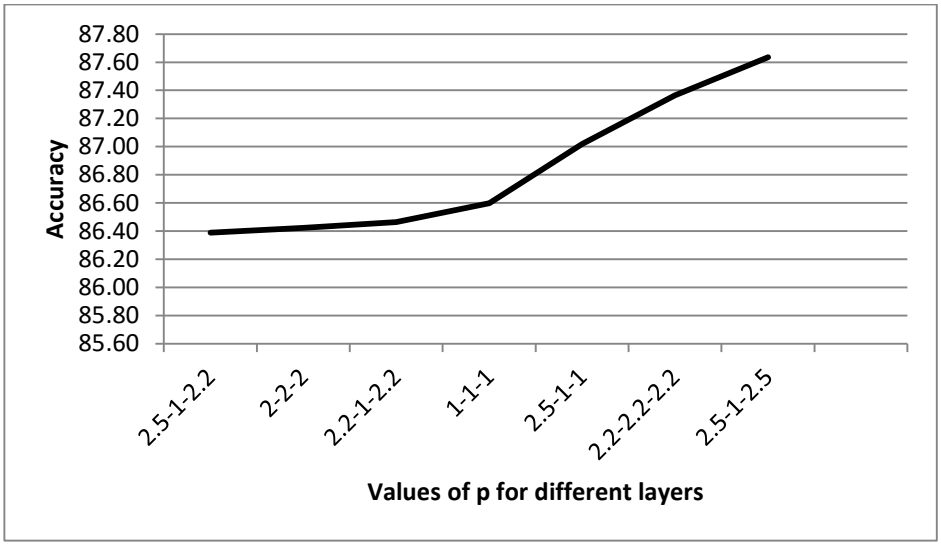

Fig. 5.11 Effect of variation of p

For the sake of comparison we also employ the deep belief network (DBN) for feature extraction; the implementation for the same is available at [31]. We choose to use two non-parametric classifiers KNN (Table 5.2.3); Sparse Representation based Classifier (SRC) (Table 5.2.4) and a parametric classifier – SVM with RBF kernel (Table 5.2.5). The SVM was tuned to yield the best results for proposed l_p-autoencoder, SAE and DBN.

Table 5.2.3Classification with KNN (K=1)

Datasets	KNN		
	Proposed	**SAE**	**DBN**
MNIST	**97.44**	97.33	97.05
Basic	**95.44**	95.25	95.37
basic-rot	**85.51**	84.83	84.71
bg-rand	**87.66**	86.42	86.36
bg-img	**77.73**	77.16	77.16
bg-img-rot	**52.95**	52.21	50.47

Table 5.2.4Classification with SRC

Datasets	SRC		
	Proposed	**SAE**	**DBN**
MNIST	**98.36**	98.33	88.43
Basic	**97.16**	96.91	87.49
basic-rot	**90.54**	90.04	79.47
bg-rand	**85.29**	84.14	79.67
bg-img	**92.28**	91.03	75.09
bg-img-rot	**63.82**	62.46	49.68

Table 5.2.5Classification with SVM

Datasets	SVM		
	Proposed	**SAE**	**DBN**
MNIST	**98.64**	97.05	88.44
Basic	**97.26**	95.21	90.31
basic-rot	**90.06**	89.83	76.59
bg-rand	**86.08**	85.34	78.59
bg-img	**92.12**	84.99	75.22
bg-img-rot	**62.02**	59.14	48.53

The results show that the proposed l_p-autoencoder always yields the best results (except for KNN on the small and simple MNIST basic dataset).

We have also compared the results of our proposed approach with the stacked denoising autoencoder (SDAE) [16] and the deep belief network (DBN) from [31]; here the SDAE and DBN are fine-tuned with neural network classifier. These are quite sophisticated techniques. Table 5.2.6 shows that even our proposed method yields better results than these highly tuned techniques in most cases.

Table 5.2.6 Comparison with SDAE and DBN

Dataset	Proposed			SDAE [16]	DBN [31]
	KNN	SRC	SVM		
MNIST	97.44	98.36	98.64	98.72	**98.76**
basic	95.44	97.16	**97.26**	97.16	96.89
basic-rot	85.51	**90.54**	90.06	90.47	89.70
bg-rand	**87.66**	85.29	86.08	83.32	83.69
bg-img	77.73	92.28	92.12	89.70	**93.27**
bg-img-rot	52.95	**63.82**	62.02	56.24	52.61

Chapter-6

Conclusion

In this work we make a fundamental change to the basic autoencoder cost function. Instead of using the popular Euclidean norm to learn the encoding and decoding weights, we propose employing the l_p-norm. Small values of p (less than 1) make the autoencoder more robust to outliers.

Minimizing the l_p-norm is more involved compared to the l_2-norm. Here we follow the iterative reweighted least squares (IRLS) approach to solve l_p-autoencoder cost function. This basically requires solving a series of weighted l_2-autoencoder problems. We also use split Bregman and alternating least squares to solve the non-linear formulation of the autoencoder.

We carry out experiments on classification and denoising. For classification, we employ a stacked 3 layer architecture. The representation from the deepest layer is used for classification. Three classifiers (KNN, SRC andSVM) were tested upon. In all three cases the proposed method yields the best results compared to the standard l_2-autoencoder and the deep belief network (DBN).

For denoising, single layer architecture is used and it is seen that non-linear activation function yields better results than linear one. We show that when the noise is sparse but large (impulse noise) our proposed method outperforms the denoising autoencoder (l_2-norm), sparse denoising autoencoder, deep non-linear autoencoder and CS technique. Also outstanding results were obtained in de-aliasing an image, subsampled at 50% in Fourier space. For sparse recovery, accuracy could not be better than CS techniques but it was significantly faster.

This is the first work that shows how autoencoders can ‘learn’ to solve inverse problems. We solve generic inverse problems; as an example we solve the problem of denoising and de-aliasing. But this autoencoder can be made to ‘learn’ any inverse problem in general.

Bibliography

[1] P. J. Huber, "Robust Estimation of a Location Parameter", The Annals of Mathematical Statistics, Vol. 35 (1), pp. 73-101, 1964.

[2] R. L. Branham Jr., "Alternatives to least squares", Astronomical Journal 87, pp. 928-937, 1982.

[3] M. Shi and M. A. Lukas, "An L1 estimation algorithm with degeneracy and linear constraints", Computational Statistics & Data Analysis, Vol. 39 (1), pp. 35-55, 2002.

[4] L. Wang, M. D. Gordon and J. Zhu, "Regularized Least Absolute Deviations Regression and an Efficient Algorithm for Parameter Tuning", IEEE ICDM. pp. 690-700, 2006.

[5] I. Barrodale and F. D. K. Roberts, "An improved algorithm for discrete L1 linear approximation", SIAM Journal on Numerical Analysis, Vol. 10 (5), pp. 839-848, 1973.

[6] E. J. Schlossmacher, "An Iterative Technique for Absolute Deviations Curve Fitting", Journal of the American Statistical Association, Vol. 68 (344), pp. 857-859, 1973.

[7] G. O. Wesolowsky, "A new descent algorithm for the least absolute value regression problem" Communications in Statistics - Simulation and Computation, Vol. B10 (5), pp. 479-491, 1981.

[8] Y. Li and G. R. Arce, "A Maximum Likelihood Approach to Least Absolute Deviation Regression", EURASIP Journal on Applied Signal Processing, Vol. (12), pp. 1762-1769, 2004.

[9] A. Majumdar and R. K. Ward, “On the Choice of Compressed Sensing Priors: An Experimental Study”, Signal Processing: Image Communication, Vol. 27 (9), pp. 1035–1048, 2012.

[10] D. E. Rumelhart, G. E. Hinton and R. J. Williams, “Learning representations by back-propagating errors”, Nature, Vol. 323, pp. 533-536, 1986.

[11] P. Baldi and K. Hornik, “Neural networks and principal component analysis: Learning from examples without local minima”, Neural Networks, Vol. 2, pp. 53-58, 1989.

[12] M. Chen, Z. Xu, K. Weinberger and F. Sha, "Marginalized Denoising Autoencoders for Domain Adaptation", arXiv:1206.4683

[13] P. Vincent, H. Larochelle, I. Lajoie Y. Bengio and P. A. Manzagol, “Stacked denoising autoen coders: Learning useful representations in a deep network with a local denoising criterion”, Journal of Machine Learning Research, Vol. 11, pp. 3371-3408, 2010.

[14] Y. Bengio, "Learning deep architectures for AI", Foundations and Trends in Machine Learning, Vol. 2 (1), pp. 1-127. 2009

[15] Y. Bengio, P. Lamblin, D. Popovici and H. Larochelle, "Greedy layer-wise training of deep networks". NIPS 2006

[16] K. H. Cho, "Simple Sparsification Improves Sparse Denoising Autoencoders in Denoising Highly Noisy Images", ICML 2013.

[17] S Rifai, P Vincent, X Muller, X Glorot, Y Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction", ICML 2011.

[18] M. Aharon, M. Elad, and A.M. Bruckstein, "K-SVD: An Algorithm for Designing of Overcomplete Dictionaries for Sparse Representation", IEEE Transactions on Signal Processing, Vol. 54 (11), pp. 4311-4322, 2006.

[19] R. Rubinstein, T. Peleg and M. Elad, "Analysis K-SVD: A Dictionary-Learning Algorithm for the Analysis Sparse Model", IEEE Trans. on Signal Processing, Vol. 61 (3), pp. 661-677, 2013.

[20] H. M. Abbas, "Analysis and pruning of nonlinear auto-association networks", IEE Proceedings-Vision, Image and Signal Processing, Vol. 151 (1), pp. 44-50, 2004.

[21] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition", Biological cybernetics, Vol. 59 (4-5), pp. 291-294, 1998.

[22] P. Rodríguez and B. Wohlberg, "Efficient Minimization Method for a Generalized Total Variation Functional", IEEE Transactions on Image Processing, Vol. 18 (2), pp. 322-332, 2009.

[23] B. Wohlberg and P. Rodríguez, "An Iteratively Reweighted Norm Algorithm for Minimization of Total Variation Functionals", IEEE Signal Processing Letters, Vol. 14 (12), pp. 948-951, 2007.

[24] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing", ICASSP 2008.

[25] Majorization Minimization:http://cnx.org/content/m32168/latest/, 2015

[26] H. Agarwal and A. Majumdar, "Generalized Synthesis and Analysis Prior Algorithms with Application to Impulse Denoising", ICVGIP 2014

[27] SPGL1 Implementation: https://www.math.ucdavis.edu/~mpf/spgl1/index.html

[28] Autoencoder Implementation:http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html

[29] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, Vol. 13 (4), pp. 600-612, 2004.

[30] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry and Y. Ma, "Robust Face Recognition via Sparse Representation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 31, pp. 210-227, 2009.

[31] Deep Belief Network Implementation: http://ceit.aut.ac.ir/~keyvanrad/DeeBNet%20Toolbox.html

[32] Goldstein, Tom, and Stanley Osher. "The split Bregman method for L1-regularized problems." *SIAM Journal on Imaging Sciences* 2.2 (2009): 323-343.

[33] Gogna, Anupriya, and Angshul Majumdar. "Matrix completion incorporating auxiliary information for recommender system design." *Expert Systems with Applications* 42.14 (2015): 5789-5799.

Curriculum Vitae

Janki Mehta

Graduate
Indraprastha Institute of Information Technology - Delhi (IIIT-D)
New Delhi, India, 110020
Phone: (+91) 9871073499
Email: mehta1485@iiitd.ac.in

Education

Master of Electronics and Communication Technology 2014 - 2016
(Specialization: Communications and Signal Processing)
Indraprastha Institute of Information Technology - Delhi (IIIT-D)
New Delhi
CPI: 8.41 (till semester-3)

Bachelor of Information and Communication Technology 2010 - 2014
Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT)
Gandhinagar, Gujarat
CPI: 8.90

Higher Secondary School 2009 - 2010
Shree P. V. Modi High School
Rajkot, Gujarat
Percentage: 90.80%

Secondary School 2007 - 2008
St. Sai High School
Rajkot, Gujarat
Percentage: 86.15%

Master's Thesis

Robust Autoencoders
Guide: Dr. Angshul Majumdar

Making robust autoencoders using l_p-norm minimization for solving inverse problems like impulse denoising and de-aliasing on CIFAR-10 dataset. It was also tested for classification on MNIST dataset and its variations. The robust autoencoder performed better than most state-of-the-art autoencoders and deep-belief networks. Also, any generic inverse problem like sparse signal recovery can be solved using this autoencoder and it gives results at par with CS techniques and is much faster.

Internships

Object Detection as an Application of Graph Matching using POCS Jan, 14 – Apr, 14
Mentor: Dr. Aditya Tatu

Converting images into graphs and then applying graph matching techniques, particularly Fast Projected Fixed-Point algorithm, to find a sub-image.

Rural Dec, 11
Ramkrishna Sarada Sevashram, Bastar (Chhattisgarh) Team Size - 4
Mentor: Dr. Amit Sengupta

Studied the tribal culture and lifestyle and devised some ICT solutions to their day to day problems. We helped in conducting health camps in the tribal villages and computerized the fee collection mechanism in the ashram school.

Academic Projects

Recommender System Design exploiting Latent Factor and Neighbourhood Models Sep, 15 – Present
Guide: Dr. Angshul Majumdar
The main idea of this project is to combine latent factor and neighbourhood models in recommender system to exploit their advantages and to try getting rid of their limitations. We have done clustering using fuzzy k-means for the neighbourhood approach. This technique has out-performed most state-of-the-art techniques in terms of MAE and RMSE.

Finding and analysing networks in Resting State fMRI data Sep,14 – May, 15
Guide: Dr. Anubha Gupta
Finding and analysing Default Mode Network (DMN) and other networks from Resting State fMRI data of human brain using data driven modelling methods like clustering, sparse representation, deep learning, etc.

Recovering partially sampled EEG signals using Learned Dictionaries Sep,14 – Dec,14
Guide: Dr. Angshul Majumdar
Learning a sparsifying dictionary using K-SVD algorithm from the training EEG signals. The trained dictionary is then used as a sparsifying transform in compressed sensing settings to recover the partially sampled test signal.A paper for the same has been **published in International Workshop on Machine Intelligence and Signal Processing – 2014.**

Image separation using Adaptive Quincunx Lifting Sep, 14 – Dec,14
Guide: Dr. Anubha Gupta
Finding approximation and difference coefficients of convolutive mixture of images using AQLS and then using some BSS algorithms like DEDS, JADE or FastICA on approximation co-efficients to seperate the images.

Adaptive Video Compression using PCA Mar, 13 – May, 13
Guide: Prof. Vijaykumar Chakka
Extraction of the features of video frames and compressing them adaptively based on required accuracy. Here we focused on the fact that video is a composition of sequential and correlated frames, so we can apply PCA to these highly correlated frames.

Skills

- Programming Languages: C, Java, OpenCV-Python, PHP, JavaScript
- Markup Languages: HTML, CSS, LaTeX
- Query Languages: SQL
- Specialized Software Tools: MATLAB, LabView, SPM8, Sparco, Protégé

Relevant Coursework

Machine Learning, Compressed Sensing, Wavelet Transform and Applications, Digital Image Processing, Computer Vision, Vector Space Projections, Statistical Signal Processing, Digital Signal Processing, Statistical Communication Theory, Adaptive Systems Design, Signals and Systems, Probability and Random Processes, Linear Optimization, Data Structures and Algorithms, Object Oriented Programming, Linear Algebra, Graph Theory, Calculus and Complex Variables, Discrete Mathematics

Positions of Responsibility

- Artificial Intelligence Club, DA-IICT — Aug, 13 - Apr, 14
- Teaching Assistant for ‘Probability and Statistics’ — Jan, 16 - Apr, 16
- Teaching Assistant for ‘Linear Algebra’ — Aug, 15 - Nov, 15
- Teaching Assistant for ‘Software Engineering’ — Aug, 14 - Dec, 14
- Teaching Assistant for ‘Introduction to C Programming’ — Aug, 13 - Nov, 13