Soft-Autoencoder and Its Wavelet Shrinkage Interpretation

Fenglei Fan, Student Member, IEEE, Mengzhou Li, Yueyang Teng, Ge Wang*, Fellow, IEEE

Abstract—Deep learning is a main focus of artificial intelligence and has greatly impacted other fields. However, deep learning is often criticized for its lack of interpretation. As a successful unsupervised model in deep learning, various autoencoders, especially convolutional autoencoders, are very popular and important. Since these autoencoders need improvements and insights, in this paper we shed light on the nonlinearity of a deep convolutional autoencoder in perspective of perfect signal recovery. In particular, we propose a new type of convolutional autoencoders, termed as Soft-Autoencoder (Soft-AE), in which the activations of encoding layers are implemented with adaptable soft-thresholding units while decoding layers are realized with linear units. Consequently, Soft-AE can be naturally interpreted as a learned cascaded wavelet shrinkage system. Our denoising numerical experiments on CIFAR-10, BSD-300 and Mayo Clinical Challenge Dataset demonstrate that Soft-AE gives a competitive performance relative to its counterparts.

Index Terms—Deep learning, Interpretability, Convolutional Autoencoder, Soft Autoencoder, Denoising.

I. INTRODUCTION

Deep learning [1-4] has made huge strides in many application fields [5-7]. As an important type of successful unsupervised learning models, the autoencoder and its variants [8-12] such as denoising autoencoder [8], contractive autoencoder [9], k-sparse autoencoder [10], variational autoencoder [11] and convolutional autoencoder [12] play a significant role in feature extraction, denoising, dimension deduction, generative tasks, and so on. However, they suffer from lack of governing theory and interpretability like other deep neural networks for machine learning. Hence, it is still difficult to understand the mechanism of the autoencoder, not mention to have any specific guidelines for the optimal design of an autoencoder in a task-specific fashion. As a result, only empirical experience serves as the base for auto-encoder prototyping. Nevertheless, the auto-encoder is rather useful. For example, partially corrupted inputs can be processed by an autoencoder that captures essential characteristics of inputs and removes noise from them. Many tricks can be used to refine the auto-encoder performance, such as the use of a contractive penalty norm [8]. On the other hand, how an autoencoder works, or more generally, how a deep neural network works, remains a hot topic of deep learning so that trust can be gained in deep networks to promote their real-world applications.

The existing methods that explain neural networks can be categorized into four classes [13]: hidden neuron analysis [14], model mimicking methods [15-16], localized interpretation methods [17-18] and advanced physics/engineering methods [19-20]. The hidden neuron analysis methods interpret a neural network by visualizing the features extracted by hidden neurons. The model mimicking methods directly build explainable models that realize a performance as similar as possible to that of the “black-box” models. Given trained neural networks, the localized interpretation methods investigate the importance of inputs by perturbing the input and analyzing changes in the resultant output. Lastly, the advanced physics/engineering methods find significant connections between deep networks and advanced physical or engineering systems to reveal the mechanisms of neural networks. Note that such a classification is qualitative and imprecise, some methods can be put into multiple classes from different perspectives. For example, our fuzzy logic interpretation method [21] analyzes the spectrum of every quadratic neuron and can be categorized as either hidden neuron analysis or advanced engineering methods.

This work is partially supported by Clark & Crossan Endowed Fund. Fenglei Fan (fanf2@rpi.edu), Mengzhou Li (lim23@rpi.edu) and Ge Wang* (E-mail: wangg6@rpi.edu) are with the Department of Biomedical Engineering, Rensselaer Polytechnic Institute, Troy, NY, USA, 12180. Yueyang Teng is with Sino-Dutch Biomedical and Information Engineering School, Northeastern University, Shenyang, China, 110169. Asterisk indicates the corresponding author.

Figure 1. Soft-autoencoder interpreted as a wavelet shrinkage system after the activation functions are appropriately made. Skip-connections are used to improve the system training and performance.

Instead of using multi-layer perceptrons (MLP) structure in other variants of the autoencoder, the convolutional autoencoder is a special type of the autoencoder, which is intrinsically appropriate for 2D and 3D imaging tasks. When dealing with 2D or 3D image formation and analysis, a fully-connected autoencoder is both unrealistic due to the memory requirement and unnecessary yielding redundancy in the space of parameters. In contrast, the convolutional autoencoder incorporates convolution/deconvolution operations in its encoding and decoding processes, thereby reducing computational overhead and network redundancy and permitting multi-resolution analysis in a nonlinear fashion. Along this direction, we propose an interpretable convolutional autoencoder, termed as Soft-AE, in which the activations in the encoding layers are implemented with adaptable soft-thresholding units, and the decoding layers are equipped with linear units. With such a configuration, Soft-AE performs a network-version wavelet transform embedded with soft-thresholding shrinkage operations. Therefore, a deep
Recent improvements in the convolutional autoencoder were made with the stacked what and where autoencoder (SWWAE) [28], residual encoder decoder network (RED-Net) [29], and so on. SWWAE records "what" that are fed into next layer and "where" the corresponding locations are. The synergy of "what" and "where" information renders better reconstruction results. RED-Net includes skipped connections in order to keep image details and speed up training. From technical aspects, [30] convolutional autoencoder configurations can be compared to justify that the convolutional autoencoder with symmetric convolution/deconvolution layers and pooling/un-pooling structure performs the best in the circumstances of no skip-connections.

The contributions of our work are three folds: (1) In the context of convolutional auto-encoding, we make an effort to link deep learning to contemporary signal processing, such as wavelet analysis, compressed sensing [22], dictionary learning [23]. In this initial investigation, we bridge classical wavelet analysis and deep convolutional auto-encoding by modifying activations in convolutional autoencoders in such a way that the wavelet shrinkage scheme is mapped into an autoencoder. (2) Based on the studies on the ReLU activation [24-26] and its variants such as concatenated ReLU, Max-Min Networks and ON/OFF ReLU respectively [24-26], here we approach this type of activation in a different way by turning hard thresholding to soft-thresholding [27], which is known to give less distortion and better restoration in many image processing tasks, especially for image denoising. (3) In the framework of Soft-AE, wavelet and thresholds of soft-thresholding are learned in the training stage from big data. Such a character enables Soft-AE to embrace built-in capability and robustness in contrast to traditional wavelet analysis methods since most comprehensive knowledge is contained in big data.

The recent paper [31] is most relevant to our current study, in which the convolutional framelet theory with a low-rank Hankel matrix was leveraged to represent signals by their local and non-local bases, suggesting an encoding-decoding structure that promises a perfect signal reconstruction. Albeit providing a linearized interpretation, there are several aspects that can be enhanced. (1) As mentioned in the authors' Remark 3, the non-local basis is a general pooling/un-pooling operator, however, pooling reduces the dimension of data, unnatural to the representation framework [31]. (2) To tackle with the nonlinearity from ReLU, the authors utilized two ReLUs with opposite phases to transform the nonlinearity into the linearity so that a perfect recovery conditions can be argued. Although this trick is sound and interesting, it potentially hurts the power of deep learning because it counteracts the nonlinearity commonly accepted as the key ingredient of deep learning. In contrast, our model is in analog to a wavelet shrinkage system, where pooling and un-pooling operations are abandoned to keep structural consistency, therefore our interpretation avoids the difficulty in explaining pooling and un-pooling. Further our model favorably accommodates the nonlinearity as the critical characteristic of our framework in the form of soft-thresholding.

The rest of this paper is organized as follows. In Section II, to put our work in perspective, we describe preliminaries, deep analysis/synthesis filters bank systems, the wavelet shrinkage algorithms, and convolutional autoencoders. In Section III, we develop our Soft-Autoencoder and present its properties and related heuristics. In Section IV, we will demonstrate that the Soft-Autoencoder is competitive compared to several competing networks. In Section V, we discuss model configurations and their wavelet characteristics. Finally, we conclude our paper.

II. Deep Filter Bank System and Convolutional Autoencoder

For completeness, let us first introduce related preliminaries and then review the analysis/synthesis filters bank theory in the single-input-single-output case. More importantly, we will cast light on the convolutional autoencoder in reference of filter bank theory. Furthermore, we review the wavelet shrinkage algorithm and its characteristics.

A. Preliminaries

Z-Transform: The Z-transform of a signal \( f[n] \) is defined as:

\[
F(z) = Z\{f[n]\} = \sum_{n=-\infty}^{+\infty} f[n]z^{-n},
\]

For convolution, Z-transform will give a multiplicative relationship. The Z-domain analysis plays a primary role in understanding the behavior of the filter bank and the requirements of a perfect signal reconstruction.

M-Down-sampling: With respect to a discrete signal \( x[n] \), the M-down-sampling of \( x[n] \) is to pick up the instances with the stride \( M \) to form a new sequence \( \hat{x}[n] \), where \( \hat{x}[k] = x[Mk] \). Technically speaking, down-sampling comprises two steps: set some samples to zeros and compress the unaffected samples into a condensed form.

With the Z-domain analysis, we have the following equation that relates \( \hat{X}(Z) \) and \( X(Z) \):

\[
\hat{X}(Z) = \frac{1}{M} \sum_{l=1}^{M} X\left( e^{-j2\pi l \frac{1}{M}} \right)
\]

M-Up-sampling: The M-Up-sampling for a discrete sequence \( x[n] \) is to expand \( x[n] \) to \( \hat{x}[n] \) by inserting (M-1) zeros between two samples in the sequence. In the Z-domain, \( x[n] \) and \( \hat{x}[n] \) are related by:

\[
\hat{X}(Z) = X(Z^M)
\]

Soft-thresholding: Soft-thresholding [27] has been extensively used in signal processing. Given an input \( x \), the soft-thresholding unit will output

\[
y = \text{sgn}(x)(|x| - b)^+,
\]

where threshold \( b \) is empirically pre-determined and \((\cdot)^+\) truncates negative part of the input. However, in our Soft-AE, \( b \) will be advantageously learned in the training process from a training dataset.
Wavelet transform: The wavelet transform of \( f(x) \) by wavelet \( \Psi(x) \) is defined as follows:

\[
W(f)(a,b) = \int_{-\infty}^{\infty} \Psi \left( \frac{x-a}{b} \right) f(x) dx, \tag{5}
\]
where \( \Psi \) is a pre-determined wavelet. Common wavelets are Morlets, Daubechies wavelets, and so on. \( W(f)(a,b) \) is called wavelet coefficients.

### B. Perfect Reconstruction

![Analysis and synthesis filter banks for perfect signal recovery in the single-input-single-output case.](image)

Filter banks are shown in Fig. 2, where we denote every signal in its Z-domain. The input signal \( x[n] \) is convoluted by \( M \) filters and \( N \)-down-sampled to obtain \( M \) intermediate subband signals \( y_0[l], y_1[l], y_2[l], ..., y_{M-1}[l] \). Then, these intermediate signals are \( N \)-up-sampled before being fed into the corresponding synthesis filters. The final output is obtained by combining the signals processed by the synthesis filters. In this pipeline, the down-sampling rate \( N \) should be no more than the number of channels for perfect reconstruction.

According to Eqs. (2) and (3), we can explicitly describe intermediate \( k^{th} \) sub-band signals in the Z-domain:

\[
Y_k(Z) = \frac{1}{N} \sum_{p=0}^{N-1} G_k(e^{-\frac{2\pi j p}{N}} Z) X(e^{-\frac{2\pi j p}{N}} Z), \tag{6}
\]
with \( Y_k(Z), k = 0,1, ..., M-1 \), and we have the final output in the Z-domain:

\[
\hat{X}(Z) = \frac{1}{N} \sum_{p=0}^{N-1} \sum_{k=0}^{M-1} F_k(Z) G_k(Z e^{-\frac{2\pi j p}{N}} Z) X(Z e^{-\frac{2\pi j p}{N}} Z) \tag{7}
\]

To represent the requisites for perfect signal recovery, we suppose:

\[
G(Z) = \begin{bmatrix}
G_0(Z) & G_0(Z e^{-\frac{2\pi j}{N}}) & \cdots & G_0(Z e^{-\frac{2\pi j (N-1)}{N}}) \\
G_1(Z) & G_1(Z e^{-\frac{2\pi j}{N}}) & \cdots & G_1(Z e^{-\frac{2\pi j (N-1)}{N}}) \\
\vdots & \vdots & \ddots & \vdots \\
G_{M-1}(Z) & G_{M-1}(Z e^{-\frac{2\pi j}{N}}) & \cdots & G_{M-1}(Z e^{-\frac{2\pi j (N-1)}{N}})
\end{bmatrix},
\]

\[
F(Z) = [F_0(Z), F_1(Z), ..., F_{M-1}(Z)],
\]
and

\[
\hat{X}(Z) = [X(Z), X(Z e^{-\frac{2\pi j}{N}}), ..., X(Z e^{-\frac{2\pi j (N-1)}{N}})].
\]

Then, Eq. (9) can be simplified as

\[
\hat{X}(Z) = F(Z) G(Z) X^*(Z)^T \tag{8}
\]

For perfect recovery, the filters should be chosen to eliminate aliasing terms: \( X(Z e^{-\frac{2\pi j}{N}}), ..., X(Z e^{-\frac{2\pi j (N-1)}{N}}) \). By causality, the best situation for signals in the time domain is to make the output signal equal to the time-delayed input signal. In other words, the perfect reconstruction is achieved when

\[
\frac{1}{N} F(Z) G(Z) = Z^{-q}[1,0,0,...,0] \tag{9}
\]

Comments: Although the above derivation is in the single-input-single-output case, in principle it can be extended to multi-input-multi-output cases which are structurally similar to a convolutional autoencoder. Down-sampling and up-sampling are conceptually the linear counterparts of pooling and un-pooling in a deep neural network. As we know, although pooling and un-pooling operations can reduce redundant information, they are often not included in the convolutional autoencoders, in hope for recovering structural details of authentic images. ReLU has been the most popular nonlinear activation function in deep learning because it has engineering strengths of preventing the searching gradient from vanishing or exploding. The system in Figure 2 implies the use of linear activations. Suppose ReLU activations are employed, then this filter bank system turns into a shallow five-layer convolutional autoencoder. Thus, the key difference between a filter bank system and a convolutional autoencoder lies in the use of the activation function.

### B. Wavelet Shrinkage Denoising

Donoho and Johnstone [27] proposed the wavelet shrinkage algorithm, which was theoretically proved with optimal denoising properties. Basically, the wavelet shrinkage algorithm contains the following three steps: (1) perform the wavelet transform to derive empirical wavelet coefficients; (2) apply a soft-thresholding operation to the wavelet coefficients; and (3) perform the inverse wavelet transform. Mathematically, suppose that we have the following additive noise model with one variation: \( Y(t) = S(t) + \epsilon(t) \). Then, the above three steps will correspond to the following three formulas: (1) \( \tilde{Y} = W[Y] \), (2) \( Z = sgn(\tilde{Y})(|\tilde{Y}| - \lambda)^+ \), (3) \( \tilde{S} = W^{-1}(Z) \). In solving real-world problems, wavelet shrinkage denoising algorithms produced excellent results. A theoretical rationale is given in the following theorems [27]:

**Theorem 1:** Let \( \{\tilde{f}_n^*(t)\} \) be the estimated vector of functional values using the above wavelet shrinkage denoising algorithm. There exists a smooth interpolation \( f_n^*(t) \) reproducing these values such that there are universal constants \( \pi_n \) with \( \pi_n \to 1 \) as \( n = 2^j \to \infty \), and constant \( C_1(F, \Psi) \) depending on the function space \( F[0,1] \in S \) and wavelet basis \( \Psi \), so that:

\[
\Pr \left\{ \| f_n^* \|_F \leq C_1 \| f \|_F, \forall F \right\} \geq \pi_n. \tag{10}
\]

Simply speaking, \( f_n^*(t) \) is as smooth as \( f \) with very high probability. This theorem is a strong claim on the smoothness of the solution, and indicates the noise-free property of the wavelet shrinkage system. A straightforward example is that if
we suppose \( f \) is a zero function, then with a high probability \( \pi_n \), \( f^*_n \) is also zero function. On the contrary, some other estimators exhibit annoying bumps even when reconstructing very smooth functions. This immune property from noise is significant in content-sensitive fields, such as nodules early detection, satellite remote sensing, and so on.

**Theorem 2:** Let \( F[0,1] \) denote a function space and let \( F_C \) be the ball of functions: \( \{ f : \|f\|_F \leq C \} \). For each ball \( F_C \) arising from an \( F \in S \), there is a constant \( C_2(F_C, \mathcal{F}) \), such that for all \( n = 2^l \),

\[
\sup_{f \in F_C} E \left\| \hat{f}_n^* - f \right\|^2_{L^2} \leq C_2 \cdot \log(n) \cdot \inf_{f \in F_C} E \left\| \hat{f} - f \right\|^2_{L^2} \quad (11)
\]

Eq. (11) demonstrates that \( \hat{f}_n^* \) realizes near minimaxity up to a constant proportional to \( \log(n) \) over a wide range of smoothness classes. Combined with Theorem 1, \( \hat{f}_n^* \) does not exhibit blips and ripples when achieving minimaxity.

Without further theoretical justification, here we heuristically illustrate why soft-thresholding works so well. As shown in Figure 2, the wavelet coefficients of a corrupted signal are full of glitches with small amplitudes over the whole spectrum. Apparently, linear estimators are not adequate to remove noise parts from wavelet coefficients, because noise is uneven and everywhere. In comparison, soft-thresholding on these wavelet coefficients will help threshold them to a proper level that noise is effectively removed. What is more favorable is that in the context of deep learning with big data, parameters in soft-thresholding are adaptively learned through backpropagation, and noise will be smartly removed, thereby leading to a more robust and powerful noise suppression system.

![Wavelet Transform](image)

**Figure 3.** Power of soft-thresholding in the wavelet domain.

### III. MAIN CONTRIBUTIONS

The efficacy of ReLU is now widely and successfully tested in the field of neural networks. However, ReLU could be too aggressive, and Zhao et al. proposed SWWAE to overcome such a risk somehow, in which the location information of survived variables is incorporated into the signal recovery/reconstruction. In this study, we look at ReLU from the perspective of interpretability. ReLU is a double-edge sword. Its nonlinearity leads to difficulties of explainability, while linear systems are completely interpretable but not more useful as solutions for nonlinear problems. Inspired by the idea of soft-thresholding, here we propose to enable Soft-thresholding with two ReLU units, termed as Soft-ReLU, so as to facilitate nonlinear power and interpretability simultaneously in the context of convolutional neural networks, turning a black-box convolutional autoencoder into an interpretable Soft-Autoencoder. In other words, the Soft-Autoencoder can be viewed as a trainable cascaded wavelet shrinkage system. The wavelet transform and soft-thresholding are sequentially conducted in the encoding layers, and then decoding layers invert the wavelet filtering accordingly. With big data, the Soft-Autoencoder can learn the adaptive wavelet kernels and thresholds. Mathematically, given \( x \in R \), the Soft-ReLU operation is defined as

\[
\text{Soft-ReLU}(x) =: \begin{cases} 
\text{ReLU}(x - b) - \text{ReLU}(-x - b) & \text{if} \ x > 0 \\
\alpha x & \text{if} \ x < 0
\end{cases}
\]

In parallel to our approach, there are other groups investigating on this topic as well. Coates et al. [32] used soft-thresholding to generate more independent features for linear SVM classifiers. Another way to keep negative signal information is to employ Leaky ReLU and its variants [33-34]. Instead of dropping the negative part completely, Leaky ReLU assigns a small slope in the negative domain. In contrast to the existing results, with Soft-ReLU here we make the Soft-Autoencoder an interpretable model, and incorporate the wavelet shrinkage property in the neural network.

### IV. EXPERIMENTS

To evaluate the performance of our Soft-Autoencoder, we compared to competing networks. Specifically, we selected the convolutional autoencoder with ReLU, Leaky-ReLU and Concatenated ReLU respectively. For convenience, we denote them by ReLU-AE, Leaky-AE, Conc-AE respectively. Mathematically,

\[
\text{Leaky-ReLU}(x) =: \begin{cases} 
x & \text{if} \ x > 0 \\
\alpha x & \text{if} \ x < 0
\end{cases}
\]

In Tensorflow, \( \alpha \) is set to 0.2.

Concatenated-ReLU basically concatenates two ReLU outputs in opposite phases:

\[
\text{Concatenate} \{ \text{ReLU}(x), \text{ReLU}(-x) \}
\]

One point for attention is that the dimension of inputs is altered after inputs are processed by Concatenated ReLU. That is, Conc-AE is different from that of Soft-AE, Leaky-AE and ReLU-AE in terms of output dimensionality.

Since our Soft-AE is a general interpretable autoencoder, it is worthwhile accessing its performance with skip connections. The addition of symmetric shortcuts is consistent to deep learning practice, and would not compromise its interpretability. Indeed, residual structures have remarkable advantages, as demonstrated for several autoencoders.
First, we systematically compared Soft-AE with ReLU-AE, Leaky-AE and Conc-AE in the setting of plain and residual structures, on the task of image denoising, with the benchmark datasets CIFAR-10 and BSD-300 respectively. We tested these autoencoders in their plain structures on CIFAR-10, and the enhanced autoencoders with residual structure on BSD-300, which is more complicated. Finally, we compared the performance metrics of these algorithms in clinical applications.

For structure preserving, neither pooling nor un-pooling operations were used in both plain and residual network configurations. Overall, the loss function is defined as

\[ L(\Theta) = \frac{1}{N} \sum_{i=1}^{N} ||F(\tilde{x}_i^{\text{noised}}, \Theta) - \bar{x}_i^{\text{denoised}}||^2, \]

where \( \Theta \) denotes hyper-parameters, \( \tilde{x}_i^{\text{noised}}, \bar{x}_i^{\text{denoised}} \) are the input and output vectors respectively.

### A. CIFAR-10

CIFAR-10 [35] is a classic benchmark dataset in machine learning comprised of 50,000 training images and 10,000 test images. Each image is of 32*32 in RGB channels. Considering that CIFAR-10 is a simple dataset, we did not select very deep networks and residual structures. In this study, three typical network structures were evaluated, as shown in Figure 4: (1) Four convolutional layers with 8 channels in every hidden layer, (2) Four convolutional layers with 16 channels in each hidden layer, And (3) Six convolutional layers with 16 channels per hidden layer. Convolutional kernel size was set to 3*3. The padding was used for convolution not to change the size of an image. In the case of Conc-AE, the activation function for the output layers were configured as ReLU, since the output images in three channels cannot be made positive and negative pairs. To keep symmetry, we used ReLU in the first layer as well. Concatenated ReLU activations were employed for the rest layers. For ReLU-AE and Leaky-AE, all the activations utilized ReLU and Leaky-ReLU respectively.

All the images were normalized by dividing 255. Noisy images were synthesized by adding additive Gaussian noise with zero mean and standard deviation \( \sigma = 0.1, 0.15, 0.2 \) respectively. Negative pixel values were truncated as 0. In the training process, noisy images \( x_i^{\text{noised}} \) were fed into the network, and denoised images \( x_i^{\text{denoised}} \) were accordingly compared. We used the stochastic gradient descent method to learn the mapping from noise-polluted images to clean counterparts. For ReLU-AE, Leaky-AE, Conc-AE and Soft-AE of three structures (1)-(3), the same configurations were used: the batch of 50 training samples were trained every iteration, the number of epochs was 20, the learning rate was set to \( 10^{-3} \), the network was randomly initialized. To quantitatively evaluate the denoising performance, each network was tested five times to compute mean SSIM and PSNR values. SSIM refers to structure similarity measure, and PSNR is peak signal to noise ratio. Both of them have been broadly used in imaging evaluation in the past decades.

The results are listed in TABLE I. Notes (1)-(3) in the table correspond to the three architectures in Figure 4. The best and second-best performers are marked in red and blue in the table. Generally speaking, at different noise levels, four autoencoders shared the same trend. Structures (2) and (3) yielded higher PSNR and SSIM measures than the relatively shallower structure (1). It is underlined that Soft-AE kept the best or second-best positions in most cases, and the improvements are considerable at all noise levels. However, there are also counterexamples: Plain-Soft-AE (3) performed the third in PSNR and SSIM metrics at the noise levels of 0.15 and 0.2. Overall, plain Soft-AE has competitive performance in denoising tasks over existing state-of-the-arts.

![Figure 4. Three autoencoder architectures labeled as (1), (2), and (3) respectively and evaluated on CIFAR-10.](image)

### B. BSD-300

BSD-300 consists of 300 high-quality images with different sizes, where 200 images serve as training data and 100 images for testing. We randomly selected 30,000 patches of 50*50 from these 300 images to make a new benchmark. 20,000 batches were prepared for training, and the remaining for evaluation. Similarly, we utilized the networks of three symmetric structures to perform comparison as shown in Figure 5: (1) seven hidden layers with 8 channels of 3*3 convolutional kernels in each layer, (2) seven hidden layers with 12 channels of 3*3 kernels in each layer, and (3) nine hidden layers with 8 channels of 3*3 kernels per layer. The positions of skip-connections are visualized in Figure 5. Not all paired encoder layers/decoder layers are bridged by shortcuts for less computational overhead. Notably, the topology of structure (3) is exactly the same as that of RED-CNN [35] but with less number of convolutional channels. As demonstrated in the literature, the addition of skip-connections will improve the learning process.

![Figure 5. Three convolutional autoencoder architecture with skip connections for comparison on BSD-300 data.](image)
The same protocols with CIFAR-10 were followed: The results are in Table II. The best and second-best performers are again marked in red and blue. With residual linkages, Soft-AE performed even better. In structures (1) and (3), Soft-AE performed the best in terms of both SSIM and PSNR. Particularly, the SSIM and PSNR improvements by Res-Soft-AE are significantly over Res-Conc-AE and Res-ReLU-AE. The counterexample was only seen once in the setting of Res-Sof-AE (2) at $\sigma = 0.1$. Clearly, the results on BSD-300 show that Soft-AE is superior.

**TABLE I: DENOISING PERFORMANCE COMPARISON AMONG LEAKY-AE, CONC-AE, RELU-AE AND SOFT-AE ON CIFAR-10**

| Plain-Leaky-AE (1) | Plain-Conc-AE (1) | Plain-ReLU-AE (1) | Plain-Soft-AE (1) | Plain-Leaky-AE (2) | Plain-Conc-AE (2) | Plain-ReLU-AE (2) | Plain-Soft-AE (2) | Plain-Leaky-AE (3) | Plain-Conc-AE (3) | Plain-ReLU-AE (3) | Plain-Soft-AE (3) |
|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|
| $\sigma = 0.1$    |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
| 27.043            | 26.961            | 27.150            | 27.469            | 27.936            | 27.640            | 27.919            | 27.944            | 27.898            | 27.815            | 27.974            | 28.039            |
| $\sigma = 0.15$   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
| 25.058            | 24.957            | 25.186            | 25.370            | 25.752            | 25.676            | 25.783            | 25.786            | 25.914            | 25.837            | 26.036            | 25.774            |
| $\sigma = 0.2$    |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |                   |
| 23.845            | 23.606            | 23.913            | 23.952            | 24.393            | 24.320            | 24.403            | 24.355            | 24.572            | 24.385            | 24.537            | 25.535            |

**TABLE II: DENOISING PERFORMANCE COMPARISON OF RESIDUAL LEAKY-AE, CONC-AE, RELU-AE AND SOFT-AE ON BSD-300**

| Res-Leaky-AE (1) | Res-Conc-AE (1) | Res-ReLU-AE (1) | Res-Soft-AE (1) | Res-Leaky-AE (2) | Res-Conc-AE (2) | Res-ReLU-AE (2) | Res-Soft-AE (2) | Res-Leaky-AE (3) | Res-Conc-AE (3) | Res-ReLU-AE (3) | Res-Soft-AE (3) |
|------------------|-----------------|-----------------|-----------------|------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| $\sigma = 0.1$   |                 |                 |                 |                  |                 |                 |                 |                 |                 |                 |                 |
| 29.252           | 28.507          | 28.789          | 29.543          | 29.437           | 29.367          | 29.336          | 29.363          | 29.545          | 29.037          | 29.425          | 29.700          |
| $\sigma = 0.15$  |                 |                 |                 |                  |                 |                 |                 |                 |                 |                 |                 |
| 26.999           | 26.470          | 26.786          | 27.486          | 27.424           | 27.121          | 27.287          | 27.432          | 27.349          | 26.875          | 27.275          | 28.109          |
| $\sigma = 0.2$   |                 |                 |                 |                  |                 |                 |                 |                 |                 |                 |                 |
| 25.589           | 24.462          | 25.406          | 26.153          | 26.064           | 25.829          | 25.955          | 26.094          | 25.797          | 25.237          | 25.739          | 26.267          |

**C. Mayo Clinical Data**

Low-dose CT imaging has gained a considerable traction over the past decade due to its potentials to decrease the x-ray induced risk to a patient. One effective way to reduce the X-ray dose is to use a weaker X-ray flux. However, a reduced X-ray flux will elevate image noise and compromise image quality. Currently, algorithms dedicated to image denoising can be roughly put into three categories [35]: (a) sinogram
domain filtering, (b) iterative reconstruction, (c) image post-processing.

Sinogram filtering methods [37-39] can be used when the data format is available and noise character is known. Albeit this, sinogram filtering tends to reduce spatial resolution, since edges in the sinogram are not clear. On the other hand, image-domain iterative methods were intensively investigated, such as compressed sensing methods [40-44] and model based iterative reconstruction [45]. Although iterative algorithms produced encouraging results, their computational costs are high, and residual image artifacts are often significant. Image post-processing methods, such as dictionary learning [46] and block-matching 3D [47-48], are directly applied to low-dose CT images without any direct access to raw data. The barrier for post-processing methods is that the noise distribution cannot be perfectly pre-determined, leading to structural blurring or distortion.

Recently, deep learning methods were successfully applied to low-dose CT denoising, such as RED-CNN [35] and transfer learning based networks [5], which achieved state-of-the-art denoising results. Hence, here we tested the denoising performance of our Soft-AE on low-dose CT denoising task with a real clinical dataset, which was prepared by Mayo Clinics for “the 2016 NIH-AAPM-Mayo Clinic Low Dose CT Grand Challenge”. This dataset has 2,378 full dose and corresponding quarter dose 512*512 CT images of 10 patients. Considering data scarcity and expensive networks, we randomly extracted 64,000 64*64 patches from these images for training. In this dataset, we continued resorting to residual structures in the RED-CNN topology in Figure 5 and employed residual ReLU-AE, Leaky-AE and Conc-AE. It is observed that the over-smoothing problem appeared with the published RED-CNN method [35]. As far as Soft-AE is concerned, we utilized 34 layers with 8 convolutional kernels in each layer. The hyper-parameters for training include 50 batches in each iteration, the learning rate for Adam optimization $1.5e^{-3}$ in the first 20 epochs and $1.0e^{-3}$ in the final 10 epochs.

Table III: RESULTS ASSOCIATED WITH SELECTED SLICES.

| Learning Rate$e^{-3}$ | Res-Soft-AE (1) | Res-Soft-AE (1)-sf | Res-Soft-AE (1) | Res-Soft-AE (1)-sf |
|------------------------|----------------|-------------------|----------------|-------------------|
| 0.05                   | 0.8578         | 0.8552            | 0.8579         | 0.8403            |
| 0.1                    | 0.8806         | 0.8565            | 0.8940         | 0.8521            |
| 0.5                    | 0.8963         | 0.8861            | 0.9067         | 0.8905            |
| 1.5                    | 0.9005         | 0.9013            | 0.9064         | 0.8945            |
| 2.5                    | 0.8995         | 0.9056            | 0.9043         | 0.9014            |
| 6.5                    | 0.8989         | N/A               | 0.9059         | 0.3869            |
| 8.5                    | N/A            | N/A               | N/A            | N/A               |
| 9.0                    | N/A            | N/A               | N/A            | N/A               |

Two representative abdominal CT slices were selected to evaluate the performance of Soft-AE. Table III tabulates the results from these images. The best SSIM results are from Soft-AE in Figure 6. In Figure 8, both PSNR and SSIM are the highest with Soft-AE, although the difference was not large. Visually speaking, it is clearly seen in Figures 6 and 8 that Soft-AE achieved a good balance between noise suppression and structural fidelity. In the zoomed parts in Figures 7 and 9, as the regions highlighted in the red circles, Soft-AE gave more desirable results in a best contrast.

V. DISCUSSIONS

In this section, several important network issues, including the learning rate, batch size, and layer depth, that affect the performance of Soft-AE are discussed for the purpose of casting light on the optimization of our proposed Soft-AE.

A. Model Configurations.

1) Learning rate: Learning rate is an important hyperparameter that determines how rapidly weights and biases are compensated for in each iteration. A high learning rate may make models diverge, albeit it accelerates training. In contrary, a low learning rate has a desirable property for assuring convergence, but the model may not converge fast and may be trapped to local minima. Configuring a proper learning rate usually relies on intuition, experience and, more importantly, experiments. In this study, with typical models, we used SSIM to evaluate the performance of plain Soft-AE and residual Soft-AE subject to different learning rates. The chosen models are Res-Soft-AE (2) & (3) that were used in the aforementioned experiments and their shortcut-free variants. A shortcut-free version is denoted as (sf). The noise level was set to 0.1. As Table IV suggested, the effective range of the learning rate for the structures with shortcuts is larger than that for the corresponding shortcut-free networks, which means that the trainability is indeed improved by skip-connections.

Table IV: QUANTITATIVE SSIM RESULTS ON THE EFFECTIVE RANGE OF THE LEARNING RATE (N/A means the model cannot converge at that learning rate).

Two representative abdominal CT slices were selected to evaluate the performance of Soft-AE. Table III tabulates the results from these images. The best SSIM results are from

2) Batch Size: Due to the computational burden, it is usually prohibitive to pass the entire dataset at once for network training. Generally, there are two approaches to train neural networks: online learning [49] and batch learning [50]. In batch learning, batch size is a key factor that determines how many samples flow into the network each time to update the hyper-parameters. A small batch size is computationally easier to handle and may help the training trajectory jump out of local minima. However, if the batch size is too small, it may not reduce the loss function effectively, jeopardizing the overall network performance. To explore the best batch size for our Soft, we performed a series of tests based on the aforementioned two typical networks.
As indicated by Figure 10, Soft-AE with or without residual links tends to favor relatively small batch sizes. When
the batch size surpassed 200, the performance dropped sharply. We conclude that 5-50 is a sweet region for our Soft-AE.

![Graph showing batch size significantly influencing the network performance.](image)

Figure 10. Batch size significantly influencing the network performance.

3) Layer Depth: One consensus regarding deep learning is that the performance becomes better as a network goes deep. To check whether Soft-AE fulfills such an expectation or not, we investigated the relationship between the performance and depth of Soft-AE. When deep networks contain no shortcuts they are rather difficult to train. Hence, our investigation was based on residual Soft-AE. We tested the use of 15, 23, 31 numbers of layers with eight channels of 3*3 convolution kernels per layer. The results are in Table V. Our data suggest that the network performs better when it becomes deeper, especially the improvement is considerable from 15 layers to 23 layers and from 23 layers to 31 layers. However, additional gains are marginal after the number of layers is beyond 39.

Table V: QUANTITATIVE RESULTS ASSOCIATED WITH NETWORK DEPTH

| Number of Layers | 15 | 23 | 31 | 39 |
|------------------|----|----|----|----|
| SSIM             | 91.090% | 91.317% | 91.618% | 91.692% |

B. Learned Wavelets

In [20], it is argued that deep networks perform a kind of multi-resolution wavelet analysis in a nonlinear way. Our Soft-AE actually moves forward along this direction. Replacing ReLU activation with soft-thresholding and linear activation in convolutional autoencoders reflects a flavor of “linearity” in the context of networks, expressing Soft-AE as a learned and twisted wavelet shrinkage system. To perceive the nature of wavelets from Soft-AE, we visualize convolutional kernels of the second hidden layer, in total 64 kernels of 5*5, from a trained network.

Soft-AE (3) in Figure 11. Taking kernels marked by the rectangular boxes as an example, the the red boxes look like low-pass wavelets, while the yellow boxes are similar to high-pass filters of different orientations.

![Gallery of some convolutional kernels of Soft-AE (3) reminding us the appearance of classic wavelets.](image)

Figure 11. Gallery of some convolutional kernels of Soft-AE (3) reminding us the appearance of classic wavelets.

VI. CONCLUSION

In conclusion, we have investigated the ReLU activation in the setting of convolutional autoencoders and introduced a pair of ReLU units emulating soft-thresholding, thereby offering the network interpretability while enhancing the network performance as well. As a result, we propose to interpret our Soft-AE as a deeply learned nonlinear wavelet shrinkage system. Our experiments on representative datasets and clinical benchmark have demonstrated the utilities of our Soft-AE. In the future, tasks such as image inpainting and image super-resolution can be revisited in our framework.

REFERENCES

[1] Y. LeCun, Y. Bengio, G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436, 2015.
[2] F. Fan, W. Cong, & G. Wang, “A new type of neurons for machine learning,” IJNMBE, vol. 34, no. 2, e2920, 2018.
[3] F. Fan, W. Cong, & G. Wang, “Generalized backpropagation algorithm for training second-order neural networks,” IJNMBE, vol. 34, no. 5, e2956, 2018.
[4] K. He, et al., “Deep residual learning for image recognition,” in CVPR, 2016.
[5] H. Shan, et al., “3-D Convolutional Encoder-Decoder Network for Low-Dose CT via Transfer Learning From a 2-D Trained Network,” IEEE transactions on medical imaging, vol. 37, no. 6, pp. 1522-1534, 2018.
[6] H. Shan, et al., “Can Deep Learning Outperform Modern Commercial CT Image Reconstruction Methods?” arXiv preprint arXiv:1811.03691, 2018.
[7] H. Chen, et al., “LEARN: Learned experts’ assessment-based reconstruction network for sparse-data CT,” IEEE transactions on medical imaging, 2018.
[8] P. Vincent, et al. “Extracting and composing robust features with denoising autoencoders,” In ICML, 2018.
[9] S. Rifai, et al., “Higher order contractive auto-encoder,” In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 645-660). Springer, Berlin, Heidelberg, 2011.
[10] A. Makhzani and B. Frey, “K-sparse autoencoders,” arXiv preprint arXiv:1312.5663, 2013.
[11] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” arXiv preprint arXiv:1312.6114, 2013.
[12] J. Masci, U. Meier, D. Cireşan & J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction. In ICANN, 2011.

[13] L. Chu, X. Hu, J. Hu, L. Wang, & J. Pei, “Exact and Consistent Interpretation for Piecewise Linear Neural Networks: A Closed Form Solution,” in KDD, 2018.

[14] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” In CVPR, 2015.

[15] M. Wu, M. C. Hughes, S. Parbhoo, et al. “Beyond Sparsity: Tree Regularization of Deep Models for Interpretability,” arXiv preprint arXiv:1711.06178, 2017.

[16] L. Fan, “Revisit Fuzzy Neural Network: Demystifying Batch Normalization and ReLU with Generalized Hamming Network,” In NIPS, 2017.

[17] P. W. Koh, P. Liang, “Understanding black-box predictions via influence functions,” in ICML, 2017.

[18] B. Jorgesh, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” In CVPR, 2016.

[19] N. Lei, K. Su, L. Cui, S. T. Yau, & D. X. Gu, “A Geometric View of Optimal Transportation and Generative Model,” arXiv preprint arXiv:1710.05488, 2017.

[20] G. Wang, “A perspective on deep imaging,” IEEE Access, vol. 4, pp. 8914-8924, 2016.

[21] F. Fan and G. Wang, “Fuzzy Logic Interpretation of Artificial Neural Networks,” arXiv preprint arXiv:1807.03215, 2018.

[22] H. Yu and G. Wang, “Compressed sensing based interior tomography,” Physics in medicine & biology, vol. 54, no. 9, pp. 2791-2009.

[23] W. Wu, et al., “Low-dose spectral CT reconstruction using image gradient 10-norm and tensor dictionary.” Applied Mathematical Modelling vol. 63, pp. 538-557, 2018.

[24] W. Shang, K. Sohn, D. Almeida and H. Lee, “Understanding and improving convolutional neural networks via concatenated rectified linear units,” In ICMIL, 2016.

[25] M. Blot, M. Cord, and N. Thome, “Max-min convolutional neural networks for image classification,” In ICIP, 2016.

[26] J. Kim, S. Kim and M. Lee, “Convolutional neural network with biologically inspired on/off relu.” In NIPS, 2015.

[27] D. L. Donoho, “De-noising by soft-thresholding,” IEEE transactions on information theory. vol. 41, no. 3, pp. 613-27, 1995.

[28] J. Zhao, M. Mathieu, R. Goroshin and Y. Lecun, “Stacked what-where auto-encoders,” arXiv preprint arXiv:1506.02351, 2015.

[29] X. Mao, C. Shen and Y. B. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections.” In NIPS, 2016.

[30] V. Turchenko, E. Chalmers and A. Luczak, “A Deep Convolutional Auto-Encoder with Pooling-Unpooling Layers in Caffe.” arXiv preprint arXiv:1701.04949, 2017.

[31] J. C. Ye, Y. Han and E. Cha, “Deep convolutional framelets: A general deep learning framework for inverse problems,” SIAM Journal on Imaging Sciences, vol. 11, no. 2, pp.991-1048, 2018.

[32] A. Coates and A. Y. Ng, “The importance of encoding versus training with sparse coding and vector quantization,” In ICML, 2011.

[33] A. L. Maas, A.Y. Hannun, and A.Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” In ICML, 2013.

[34] K. He, X. Zhang, S. Ren and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” In 2015.

[35] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Master’s thesis, Department of Computer Science, University of Toronto, 2009.

[36] H. Chen, et al., “Low-dose CT with a residual encoder-decoder convolutional neural network,” IEEE transactions on medical imaging, vol. 36, no. 12, pp. 2524-2535, 2017.

[37] M. Balda, J. Hornegger, and B. Heisemann, “Ray contribution masks for structure adaptive sinogram filtering,” IEEE Trans. Med. Imaging, vol. 30, no. 5, pp. 1116–1128, 2011.

[38] A. Manduca, L. Yu, J. D. Trzasko, N. Khayllova, J. M. Kofler, C. M. McCollough, and J. G. Fletcher, “Projection space denoising with bilateral filtering and CT noise modeling for dose reduction in CT,” Med. Phys, vol. 36, no. 11, pp. 4911–4919, 2009.

[39] J. Wang, T. Li, H. Lu, and Z. Liang, “ Penalized weighted least-squares approach to sinogram noise reduction and image reconstruction for low dose X-ray computed tomography,” IEEE Trans. Med. Imaging, vol. 25, no. 10, pp. 1272–1283, 2006.

[40] E. Y. Sidky and X. Pan, “Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization,” Phys. Med. Biol. vol. 53, no. 17, pp. 4777–4807, 2008.

[41] Y. Zhang, W. Zhang, Y. Lei, and J. Zhou, “Few-view image reconstruction with fractional-order total variation,” J. Opt. Soc. Am. A, vol. 31, no. 5, pp. 981–995, 2014.

[42] Y. Zhang, Y. Wang, W. Zhang, F. Lin, Y. Pu, and J. Zhou, “Statistical iterative reconstruction using adaptive fractional order regularization,” Biomed. Opt. Express, vol. 7, no. 3, pp. 1015–1029, 2016.

[43] Q. Xu, H. Yu, X. Mou, L. Zhang, J. Hsieh, and G. Wang, “Low-dose xray CT reconstruction via dictionary learning,” IEEE Trans. Med. Imaging, vol. 31, no. 9, pp. 1682–1697, 2012.

[44] J.-F. Cai, X. Jia, H. Gao, S. B. Jiang, Z. Shen, and H. Zhao, “Cine cone beam CT reconstruction using low-rank matrix factorization: algorithm and a proof-of-principle study,” IEEE Trans. Med. Imaging, vol. 33, no. 8, pp. 1581–1591, 2014.

[45] M. Katsura, M. Matsuda, M. Akahane, et al., “Model-based iterative reconstruction technique for radiation dose reduction in chest CT: comparison with the adaptive statistical iterative reconstruction techniques,” Eur. Radiol, vol. 22, no. 8, pp. 1613–1623, 2012.

[46] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” IEEE Trans. Signal Process., vol. 54, no. 11, pp. 4311–322, 2006.

[47] P. F. Feruglio, C. Vinegoni, J. Gros, A. Sbarbati, and R. Weissleder, “Block matching 3D random noise filtering for absorption optical projection tomography,” Phys. Med. Biol., vol. 55, no. 18, pp. 5401–5415, 2010.

[48] D. Kang, P. Slomka, R. Nakazato, J. Woo, D. S. Berman, C.-C. J. Kuo and D. Dey, “Image denoising of low-radiation dose coronary CT angiography by an adaptive block-matching 3D algorithm,” Proc. SPIE 8669, 86692G, 2013.

[49] J. Duchi, E. Hazan & Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” Journal of Machine Learning Research, vol. 12, pp. 2121-2159, 2011.

[50] J. Duchi and Y. Singer, “Efficient online and batch learning using forward-backward splitting,” Journal of Machine Learning Research, vol. 10, pp. 2899-2934, 2009.

[51] S. Zagoruyko and N. Komodakis. “Wide residual networks,” arXiv preprint arXiv:1605.07146, 2016.