Simple Sparsification Improves Sparse Denoising Autoencoders in Denoising Highly Noisy Images

KyungHyun Cho KyungHyun.Cho@aalto.fi
Department of Information and Computer Science, Aalto University School of Science, Finland

Abstract

Recently Burger et al. (2012) and Xie et al. (2012) proposed to use a denoising autoencoder (DAE) for denoising noisy images. They showed that a plain, deep DAE can denoise noisy images as well as the conventional methods such as BM3D and KSVD. Both of them approached image denoising by denoising small, image patches of a larger image and combining them to form a clean image. In this setting, it is usual to use the encoder of the DAE to obtain the latent representation and subsequently apply the decoder to get the clean patch. We propose that a simple sparsification of the latent representation found by the encoder improves denoising performance, both when the DAE was trained with and without sparsity regularization. The experiments confirm that the proposed sparsification indeed helps both denoising a small image patch and denoising a larger image consisting of those patches. Furthermore, it is found out that the proposed method improves even classification performance when test samples are corrupted with noise.

1. Introduction

Many latent variable models can be cast as a model that learns an encoder and a decoder, either explicitly or implicitly (Ranzato et al., 2007). The encoder maps a given data sample to a latent space, and the decoder decodes the latent representation into the data space. For instance, principal component analysis (PCA) learns a linear encoder and decoder so that the L_2 error between a given sample and the sample reconstructed by applying the encoder and decoder sequentially is minimal (see, e.g., Bishop, 2006).

Often these models are trained to minimize the L_2 reconstruction error together with a sparsity regularization. The sparsity regularization ensures that the number of non-zero components in the latent representation given a sample is small. Sparse coding (see, e.g., Olshausen & Field, 1996) is one example that aims to minimize the reconstruction error while regularizing the sparsity of (overcomplete) latent representations.

One popular application of an encoder-decoder model with sparsity regularization has been image denoising. Elad & Aharon (2006) showed that a clean image can be constructed by denoising and combining small image patches of the noisy, original image, where sparse coding is used for denoising. Hyvärinen et al. (1999) denoised a large image by explicitly sparsifying the latent representation of each small image patch extracted from the larger image with a shrinkage nonlinearity.

More recently, Xie et al. (2012) proposed to use, yet, another encoder-decoder model called sparse denoising autoencoders (spDAE) for image denoising. An spDAE is a variant of a denoising autoencoder (DAE) proposed by Vincent et al. (2010) that uses a sparsity regularization during training. It was shown that the spDAE is also effective in denoising noisy images, especially when the number of hidden layers is larger than one.

These two approaches, sparse coding and denoising autoencoder, have two important differences. Firstly, DAEs, including an spDAE, have a parameterized encoder while sparse coding relies on optimization to obtain a latent representation. Secondly, the latent representation encoded by a DAE is not necessarily sparse in a strict sense due to the usual use of smooth, saturating nonlinearity functions. Sparse coding, on the other hand, finds a truly sparse latent representation.

Based on these observations, we claim that faster and

Proceedings of the 30^{th} International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

improved denoising can be done by explicitly sparsifying the latent representation found by spDAEs. We explain an intuition behind the claim by considering an spDAE as a previously mentioned encoder-decoder model with, potentially multi-layered, nonlinear mappings between data space and (sparse) latent space. We, then, propose a *simple sparsification* that explicitly sparsifies the latent representation of an spDAE.

We empirically confirm that the proposed simple sparsification leads to better performance by denoising various types of images corrupted with noise using spDAEs having a number of different structures. Furthermore, we show that the proposed sparsification also improves the discriminative properties of the latent representations obtained by spDAEs.

2. Sparse Denoising Autoencoders and Simple Sparsification

2.1. Sparse Denoising Autoencoder

We begin with a single-layer spDAE which is a special form of multi-layer perceptron network with a single hidden layer (Vincent et al., 2010). An spDAE tries to learn a network that reconstructs an input vector optimally by minimizing the following cost function:

$$\sum_{n=1}^{N} \left\| g \circ f \left(\eta(\mathbf{x}^{(n)}) \right) - \mathbf{x}^{(n)} \right\|^2 + \lambda \Omega(\mathbf{W}, \{\mathbf{x}^{(n)}\}), \quad (1)$$

where $\Omega(\mathbf{W}, \{\mathbf{x}^{(n)}\})$ is a sparsity regularizer, and

$$f(\mathbf{x}) = \phi(\mathbf{W}^\top \mathbf{x}) \text{ and } g(\mathbf{h}) = \mathbf{W}\mathbf{h}$$

are, respectively, an encoder and decoder with a component-wise nonlinearity function ϕ. η explicitly adds noise to an input sample $\mathbf{x}^{(n)}$. $\mathbf{W}$ is a matrix of the weights between the input layer and the hidden layer and is shared by the encoder and decoder. For notational simplicity, we omit biases to all units.

2.2. Non-linear Mapping with Constrained Range

If we assume $[0, 1]$ hidden units with a sigmoid activation function $\phi(x) = \frac{1}{1+\exp(-x)}$, the encoder f is a non-linear function that maps a sample $\mathbf{x}$ in a data space[1] $\mathbb{P} \subseteq \mathbb{R}^p$ to a potentially higher-dimensional, latent space $\mathbb{Q} \subseteq [0, 1]^q$, where p and q are the number of visible and hidden units, respectively. The decoder does exactly the opposite with $\mathbb{Q}$ and $\mathbb{P}$ as its domain and range, respectively[2]. In this setting, the sparsity regularizer Ω defines, or restricts, the range $\mathbb{Q}$ of the learned encoder f.

We should note that, for any sample $\mathbf{x} \in \mathbb{R}^p$, the sample does not belong to $\mathbb{P}$, if $f(\mathbf{x}) \notin \mathbb{Q}$. It is further expected that any sample that was corrupted by error smaller than that injected by η is still encoded to a hidden code in $\mathbb{Q}$, but any highly noisy or corrupted sample $\mathbf{x}$ maps through the encoder f to a region outside $\mathbb{Q}$, albeit not necessarily.

Let us consider a specific case of regularizing the average hidden activation to a predefined sparsity hyperparameter ρ (Lee et al., 2008). In this case,

$$\Omega(\mathbf{W}, \{\mathbf{x}^{(n)}\}) = \frac{1}{2} \left\| \frac{1}{N} \sum_{n=1}^{N} f(\mathbf{x}^{(n)}) - \rho \mathbf{1} \right\|_2^2. \quad (2)$$

Assuming that the model consists of stochastic binary hidden units with their probabilities given by the encoder f, we can see that $f(\mathbf{x})$ will have, on average, $\rho \times q$ components active while all others are inactive. If ρ is set close to 0, a sparse latent representation will be produced by the encoder f.

This leads to an encoder f with a (approximate) range, conditioned on the data set $\mathbb{P}$,

$$\mathbb{Q} \approx \left\{ \mathbf{h} = f(\mathbf{x}) \,\middle|\, \mathbf{x} \in \mathbb{P}, \|\mathbb{E}_{\mathbf{x} \in \mathbb{P}} [h_j] - \rho\|_2^2 = 0 \right\} \quad (3)$$

with the amount of error controlled by the regularization constant λ, where h_j is the j-th component of $\mathbf{h}$.

In this case, $f(\mathbf{x})$ of any sample $\mathbf{x}$ that is close to one of training samples will fall in $\mathbb{Q}$. In other words, the average activation of $f(\mathbf{x})$ will be around the predefined ρ. If the average activation of $f(\mathbf{x})$ is either too smaller or too larger than ρ, it can be suspected that the sample $\mathbf{x}$ is either not of the same type as training samples or corrupted with high level of noise.

2.3. Simple Sparsification

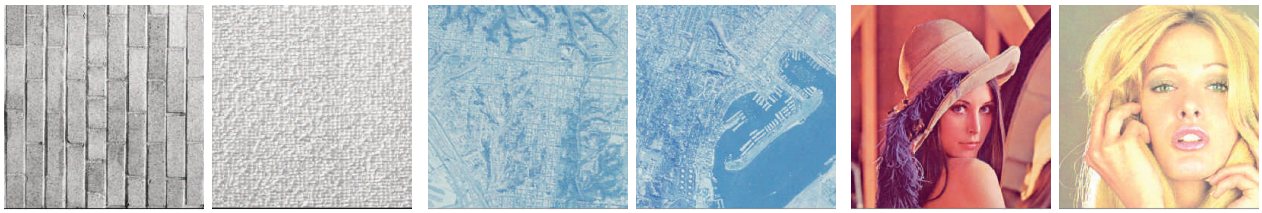
Obviously, when the latent representation $f(\tilde{\mathbf{x}})$ of a corrupted sample $\tilde{\mathbf{x}}$ is found to be outside $\mathbb{Q}$, we cannot

[1] For an spDAE that explicitly adds white Gaussian noise via η when learning the parameters, the data space $\mathbb{P}$ is defined as

$$\mathbb{P} = \left\{ \mathbf{x} \in \mathbb{R}^p \,\middle|\, \exists \mathbf{x}^{(n)} \in D, \|\mathbf{x} - \mathbf{x}^{(n)}\|_2^2 \leq \epsilon \right\},$$

where $D = \left\{ \mathbf{x}^{(n)} \right\}_{n=1}^{N}$ is a training set and ϵ is decided by the variance of white Gaussian noise explicitly added by $\eta(\cdot)$.

[2] Note that the decoder g is a *linear* mapping in the case of an spDAE with a single hidden layer. However, when an spDAE has more than one hidden layer, g becomes nonlinear as described in Section 2.4.



(a) Textures (b) Aerials (c) Miscellaneous

Figure 1. Sample images from the test image sets

expect the decoder g to correctly reconstruct the clean sample, since g was trained to map from only $\mathbb{Q}$ to $\mathbb{P}$. It is, hence, not desirable to simply apply the encoder and decoder sequentially to denoise an input sample.

Instead, it must be checked whether $\mathbf{h} = f(\tilde{\mathbf{x}})$ belongs to $\mathbb{Q}$ before the decoder g is applied. If $\mathbf{h} \notin \mathbb{Q}$, one must project it onto $\mathbb{Q}$ such that the decoder will correctly map $\mathbf{h}$ to a clean, denoised sample. As $\mathbb{Q}$ is defined by the sparsity of its data points, another way to put it is that $\mathbf{h}$ needs to be *sparsified*.

We define a sparsification R by

$$R(\mathbf{h}) = \arg\min_{\mathbf{q} \in \mathbb{Q}} d(\mathbf{h} - \mathbf{q}), \tag{4}$$

where $d(\cdot, \cdot)$ is a suitable distance metric.

Simply put, R projects $\mathbf{h} \in [0, 1]^q$ onto $\mathbb{Q}$. Depending on a type of a sparsity regularizer and a target application, one must choose a suitable d, and the choice may have impact on the denoising performance.

In the case of the previously described sparsity regularizer (2), we can, for instance, use a variant of orthogonal matching pursuit which stops when the number of zero hidden units reaches the target sparsity $\bar{\rho}$ or the average activation reaches $1 - \bar{\rho}$. However, this type of approaches using optimization will be prohibitively expensive, especially for image denoising task which requires evaluating the encoder tens and hundreds of thousands times per image.

Alternatively, we can define $R(\mathbf{h})$ to decrease each component of $\mathbf{h}$ so that the average activation is closer to $1 - \bar{\rho}$. We call this approach a *simple sparsification*, and this effectively sets small components to zero by

$$\mathbf{h} \leftarrow \max\left(\mathbf{h} - \max\left(\frac{1}{q}\|\mathbf{h}\|_1 - (1 - \bar{\rho}), 0\right), 0\right), \tag{5}$$

where max applies to each component.

Note that it does not attempt to increase the components of $\mathbf{h}$ even if the average activation of $\mathbf{h}$ is smaller than $1 - \bar{\rho}$. This is justified by the fact that noise is likely to encourage more hidden units to respond meaninglessly, assuming white Gaussian additive noise. Fig. 2(c) shows that the average activation of hidden units increases as noise does.

It might seem obvious to choose the target sparsity $\bar{\rho}$ to be $1 - \rho$ of which ρ was used when training the spDAE. Another possibility is to estimate $\bar{\rho}$ to minimize the reconstruction error of noisy training samples corrupted with a predefined level and type of noise. The latter approach can be useful when the spDAE was *not* trained with the sparsity regularization.

2.4. Deep Denoising Autoencoders

Unfortunately, applying the sparsity regularizer, for instance, the one in (2), is not computationally efficient in DAEs with multiple hidden layers. Hence, it has been common to use the sparsity regularizer only during pretraining (see, e.g., Xie et al., 2012).

Once the weights of a deep DAE are initialized by pretraining, this works as regularization that controls the sparsity of the hidden activations. After pretraining, the whole deep DAE can be further finetuned by stochastic backpropagation algorithm (Rumelhart et al., 1986).

Considering a deep DAE with $2L-1$ hidden layers, we have an encoder

$$f(\mathbf{x}) = \phi\left(\mathbf{W}^{(L-1)}\phi\left(\mathbf{W}^{(L-2)}\cdots\phi\left(\mathbf{W}^{(1)}\mathbf{x}\right)\cdots\right)\right) \tag{6}$$

that maps from $\mathbb{P}$ to $\mathbb{Q}$, and a decoder

$$g(\mathbf{h}) = \mathbf{W}^{(1)^\top}\phi\left(\mathbf{W}^{(2)^\top}\cdots\phi\left(\mathbf{W}^{(L)^\top}f(\mathbf{x})\right)\cdots\right) \tag{7}$$

that reversely maps from $\mathbb{Q}$ to $\mathbb{P}$.

In this case, it is not obvious at which stage the proposed sparsification should apply. For instance, the simple sparsification can be used at each layer of the encoder, the decoder or both of them. On the other hands, if all hidden layers in the encoder are considered as a single non-linear mapping function in whole, the simple sparsification should be applied at the bottleneck layer.

In this paper, we obtain the denoised sample by only applying the simple sparsification at the bottleneck

layer:

$$\hat{\mathbf{x}} = g(R(f(\mathbf{x}))),$$

where f and g are defined by Eq. (6) and Eq. (7), respectively. It is, however, left for future to investigate other possibilities.

2.5. Related Approaches

2.5.1. Sparse Coding

Assuming a linear generative model, sparse coding (see, e.g., Olshausen & Field, 1996) tries to find a (overcomplete) dictionary of features $\mathbf{W}$ and a set of *sparse* codes $\{\mathbf{h}^{(n)}\}_{n=1}^N$ given a set of training samples $\{\mathbf{x}^{(n)}\}_{n=1}^N$. This is achieved by minimizing the following cost:

$$\sum_{n=1}^{N} \left\| \mathbf{x}^{(n)} - \mathbf{W}\mathbf{h}^{(n)} \right\|_2^2 + \lambda\Omega(\mathbf{W}, \{\mathbf{h}^{(n)}\}),$$

where Ω is, again, a sparsity regularizer. Although this is closely related to the cost function of spDAEs in Eq. (1), there is no explicit encoder in sparse coding, and encoding has to be done via optimization.

If one makes the model more precise by requiring the sparsity of the sparse code $\mathbf{h}$ in a form $\|\mathbf{h}\|_0 \leq L$ for some integer $L \ll q$, then, given a sample $\mathbf{x}$ and the weights $\mathbf{W}$, encoding by optimization finds the latent representation $\mathbf{h}$ approximately inside $\mathbb{Q} = \{\mathbf{h} \mid \|\mathbf{h}\|_0 \leq L\}$ (Elad & Aharon, 2006). This is contrast to the spDAEs which do not guarantee that the encoded representation, without any explicit sparsification, resides in $\mathbb{Q}$.

This optimization-based approach of sparse coding and the proposed sparsification combined with an spDAE are two opposite approaches in finding a sparse latent representation. The former, for instance, based on the pursuit algorithms (see, e.g., (Chen et al., 2001)), starts from the center of $\mathbb{Q}$ (all zero hidden units) and sequentially finds the non-zero hidden units that decrease the reconstruction error, until a stopping criterion is met. The latter, proposed method, first encodes a given sample to a superset $[0, 1]^q$ of $\mathbb{Q}$, and then, projects the found latent representation onto $\mathbb{Q}$.

2.5.2. Shrinkage Nonlinearity

In a similar context of sparse coding, Hyvärinen (1999) proposed to find a dictionary by independent component analysis (ICA) and use a shrinkage nonlinearity function to find a sparse code. This approach makes obtaining sparse code computationally less demanding compared to the optimization-based encoding in the conventional sparse coding.

This approach is closely related to the proposed simple sparsifiation. Hyvärinen (1999), for instance, suggested the following shrinkage nonlinear function in the case of each latent component following a super-Gaussian distribution:

$$s(h) = \frac{1}{1 + \sigma^2 a} \text{sign}(h) \max(0, |h| - b\sigma^2),$$

where a and b are parameters to be estimated, σ^2 is a noise variance, and h is a latent component. If we assume a unit noise variance ($\sigma^2 = 1$) and $a = 0$, it becomes

$$s(h) = \text{sign}(h) \max(0, |h| - b).$$

If we set b to $\|\mathbf{h}\|_1 - (1 - \bar{\rho})$ from Eq. (5), it is easy to see that applying the shrinkage nonlinearity s to each latent component is equivalent to the proposed simple sparsification. Both of them reduce the absolute value of each latent component.

Compared to these approaches based on sparse coding, an spDAE, using the simple sparsification, has an advantage that it is natural to extend the model into a deeper model. This allows us to build a *deep* sparse generative model with a fast inference procedure, unlike the conventional sparse coding models.

3. Image Denoising

A noisy large image can be denoised by denoising small patches of the image and combining them together (see, e.g., Hyvärinen, 1999; Elad & Aharon, 2006). Let us define a set of N binary matrices $\mathbf{D}_n \in \mathbb{R}^{p \times d}$ that extract a set of small image patches given a large, whole image $\mathbf{x} \in \mathbb{R}^d$, where $d = wh$ is the product of the width w and the height h p is the size of image patches (e.g., $p = 64$ if the size of an image patch is 8×8). Then, the denoised image is constructed by

$$\tilde{\mathbf{x}} = \left(\sum_{n=1}^{N} \mathbf{D}_n^\top r_{\boldsymbol{\theta}}(\mathbf{D}_n \mathbf{x}) \right) \oslash \left(\sum_{n=1}^{N} \mathbf{D}_n^\top \mathbf{D}_n \mathbf{1} \right), \quad (8)$$

where $\oslash$ is a element-wise division and $\mathbf{1}$ is a vector of ones. $r_{\boldsymbol{\theta}}(\cdot)$ is an image denoising function, parameterized by $\boldsymbol{\theta}$, that denoises N image patches extracted from the input image $\mathbf{x}$.

Eq. (8) essentially extracts and denoises image patches from the input image. Then, it combines them by taking an average of those overlapping pixels. For a computational reason, it is usual to use overlapping image patches separated by a few pixels rather than all possible patches.

Recently, Burger et al. (2012), Xie et al. (2012) and Cho (2013) proposed to utilize a denoising autoen-

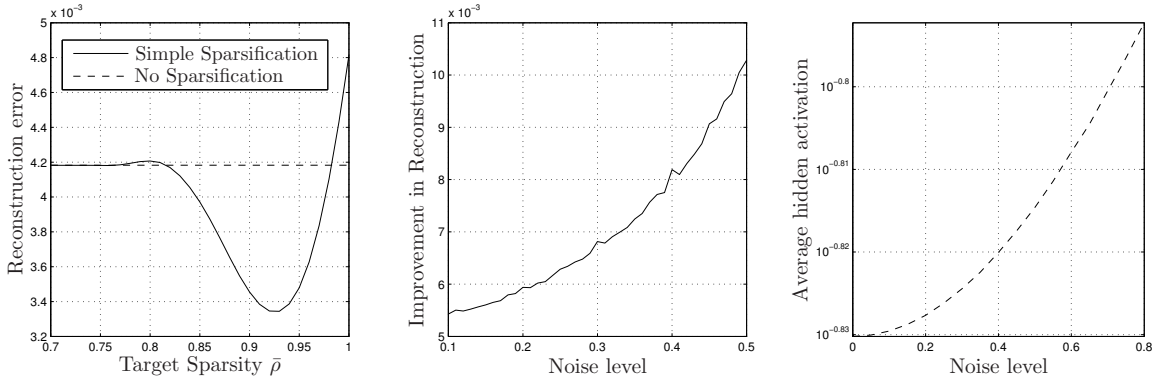


Figure 2. (a) Reconstruction error of image patches with varying target sparsity $\bar{\rho}$. (b) Improvement of reconstruction error of image patches achieved by using the simple sparsification with fixed $\bar{\rho} = 0.9$. (c) Average hidden activations given a set of noisy image patches with varying noise levels. A solid line and dashed line denote reconstruction errors obtained by the spDAE with and without the explicit sparsification, respectively. Image patches were randomly collected from the test sets and corrupted with white Gaussian additive noise of standard deviation 0.1. All errors were obtained using a single-layer DAE trained on 8×8 image patches with and without the simple sparsification.

coder (DAE) in place of $r_{\boldsymbol{\theta}}(\cdot)$ to perform image denoising. It is straightforward to denoise an image patch with a DAE, or in our case, spDAE. Given a noisy image patch, we first obtain a latent representation by applying the encoder f. Then, the decoder g will reconstruct a clean patch from the hidden representation.

The proposed sparsification R can be plugged in before the decoder is applied, that is, the denoised patch $\hat{\mathbf{x}} = g\left(R\left(f\left(\mathbf{x}\right)\right)\right)$. This applies to spDAEs with any number of hidden layers.

4. Experiments

Following the approach used by Cho (2013), we used the images from three separate sets, *textures*, *aerials* and *miscellaneous*, from the USC-SIPI Image Database[3] as test images. Fig. 1 presents six sample images from the three image sets.

All images were converted to grayscale by averaging three color channels into a single grayscale pixel. Also, each pixel of the images was normalized into $[0, 1]$ instead of the original $[0, 255]$.

Although we mainly focused on a single-layer spDAE in this paper, we trained spDAEs with one, two and four hidden layers on images patches randomly collected from CIFAR-10 dataset (Krizhevsky, 2009) to see the effect of the simple sparsification on deeper models. The size of each hidden layer was fixed to a constant multiple of the size of a visible layer[4]. We use the shorthand notations DAE, DAE(2) and DAE(4) for denoting the trained DAEs with one, two and four hidden layers, respectively.

A single-layer spDAE was trained with the sparsity regularizer given in Eq. (2) with $\rho = 0.1$. Each layer of all deep spDAEs with more than one hidden layers were pretrained as a single-layer spDAE with, again, ρ set to 0.1. We used $1 - \rho$ for the target sparsity $\bar{\rho}$ of the simple sparsification.

Unlike in (Burger et al., 2012) and (Xie et al., 2012), all the models were trained in a completely *blind* way. In other words, no prior knowledge about the level or type of noise in the test images was used. Regardless of the types or levels of noise injected in the test images, each model was trained by adding white Gaussian noise with 0.1 standard deviation and dropping 10% of input pixels at each stochastic gradient update.

4.1. Image Patch Denoising

We have extracted from each image of the test set 50 random image patches. White Gaussian additive noise of standard deviation 0.1 was added to each pixel, and it was denoised by the trained single-layer spDAE with the simple sparsification using the heuristic introduced in Section 2.3. To see the effect of the simple sparsification, we varied $\bar{\rho}$ from 0.7 to 1.

Fig. 2 (a) shows the reconstruction errors obtained by

[3] http://sipi.usc.edu/database/

[4] We used 5 as suggested by Xie et al. (2012). For instance, the models trained on 8×8 pathces have $8 \times 8 \times 5 = 320$ units per hidden layer.

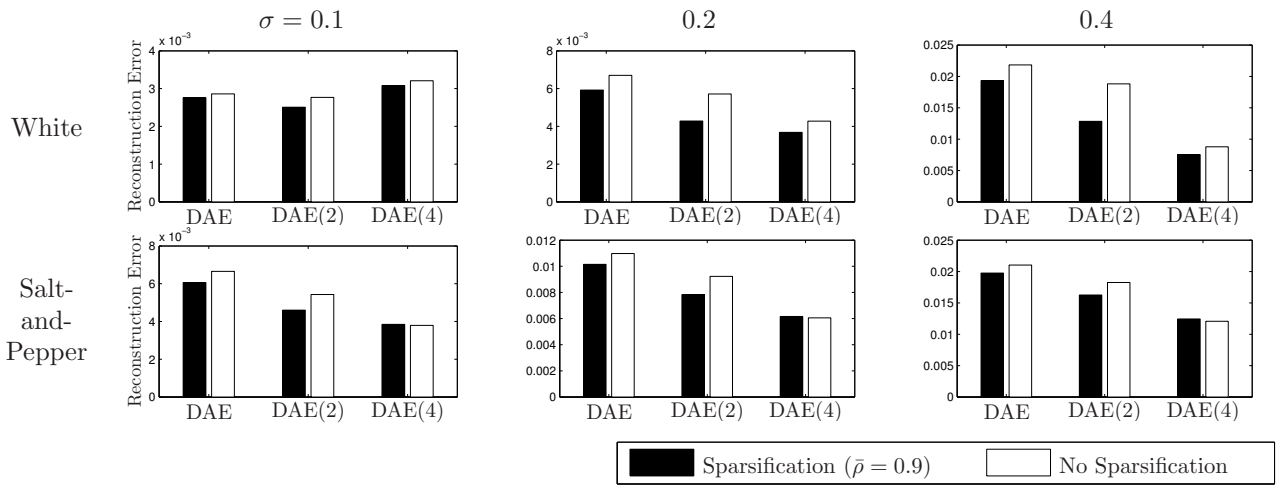


Figure 3. MSEs obtained with and without the explicit sparsification. The top and bottom rows show the denoising performance on the test images corrupted with white Gaussian additive noise and salt-and-pepper noise, respectively, of different noise levels σ. All images were denoised with the models trained on 8×8 patches. Lower is better.

the trained single-layer spDAE using the simple sparsification with varying target sparsities $\bar{\rho}$. The dashed-line shows the reconstruction error obtained without the simple sparsification. It is clear that lower reconstruction error could be achieved with the simple sparsification around $1 - \rho = 0.9$. We could also observe similar trend with the deeper spDAEs.

Furthermore, in Fig. 2 (b), we can see that the performance improvement by the simple sparsification grows as the level of noise increases. This suggests that the simple sparsification may help denoising an image especially when the image is highly corrupted.

4.2. Large Image Denoising

We tested two types of noise which were white Gaussian additive noise and salt-and-pepper noise. Three levels of noise were tested; 0.1, 0.2 and 0.4. In the case of white Gaussian noise, those levels correspond to standard deviations, while they correspond to noisy pixel probabilities with salt-and-pepper noise. The denoising performance was measured by the reconstruction error between the denoised image from the original, clean image. L_2-norm of the difference between them was used as the reconstruction error.

For the large, noisy test images, we first applied Wiener filtering with 3×3 neighborhood, as was done by Cho (2013). Subsequently, we denoised the images with the trained spDAEs following Eq. (8). We used overlapping image patches extracted every second pixel.

In Fig. 3, we can see the effect of the simple sparsification. It is clear that the proposed sparsification improves the performance of the spDAEs regardless of the level of noise. However, the improvement is more visible when the amount of noise is high (0.2 or 0.4).

This result was expected, as each image patch from those images corrupted with low level of noise are likely to be encoded into a latent representation that (approximately) resides in $\mathbb{Q}$. Hence, the simple sparsification will not alter it much, giving a performance similar to the one obtained without the simple sparsification. On the other hand, when the level of noise is high, the latent representation is likely to be outside $\mathbb{Q}$, and the simple sparsification correctly projects it onto $\mathbb{Q}$ to make the decoder behave better, resulting in superior performance.

The performance improvement was especially apparent with the spDAEs with the smaller number of hidden layers (one or two). The improvement decreased as the number of hidden layers increased. This might be due to the fact that the deep spDAEs were not fine-tuned with the sparsity regularizer, which made them encode a given sample into a *less* sparse latent representation.

4.3. Non-Regularized Denoising Autoencoders

If we consider the case where $\bar{\rho}$ is chosen to minimize the reconstruction error of noisy training samples, it is possible to consider the case where the DAEs were not trained with the sparsity regularization. The non-regularized DAEs are further interesting, as Burger et al. (2012) showed that the non-regularized ones perform comparably to, or better than, the conventional state-of-the-art denoising methods.

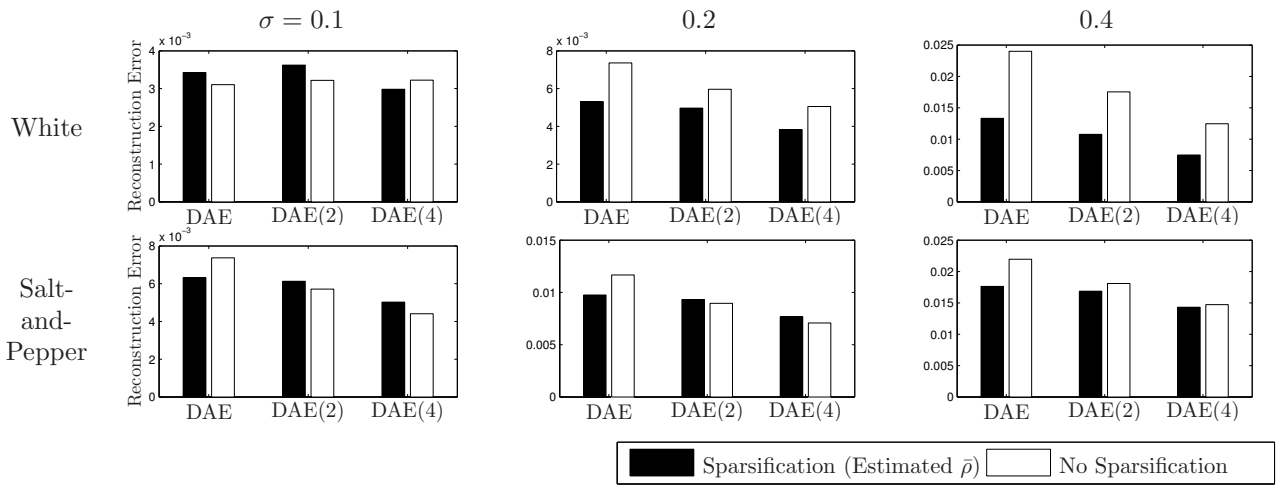


Figure 4. MSEs obtained with and without the explicit sparsification using the non-regularized DAEs. The top and bottom rows show the denoising performance on the test images corrupted with white Gaussian additive noise and salt-and-pepper noise, respectively, of different noise levels σ. All images were denoised with the models trained on 8×8 patches. Lower is better.

	4	8	16
DAE	0.83	0.84	0.83
DAE(2)	0.75	0.73	0.74
DAE(4)	0.81	0.75	0.78

Table 1. The estimated $\bar{\rho}$ for each non-regularized denoising autoencoders trained on square image patches of width 4, 8 and 16.

Hence, we have run the same set of experiments using the non-regularized DAEs. The test images were denoised with the simple sparsification using the estimated $\bar{\rho}$'s in Tab. 1.

In Fig. 4, we can see that the non-regularized DAEs also benefit from using the proposed simple sparsification when the level of noise is high. However, when only small amount of noise was injected, in some cases, the performance degraded with the sparsification.

Interestingly, we observed that the sparse DAEs outperformed the non-regularized DAEs when no sparsification was used. When the proposed sparsification was applied, however, the performance gap between them decreased. This suggests that the existing denoising approach based on non-regularized DAEs, for instance, used by Burger et al. (2012), may also benefit by simply plugging in the simple sparsification.

4.4. Discriminative Performance

Although we focus mainly on image denoising in this paper, we have also briefly investigated the potential improvement in the discriminative performance.

We used MNIST handwritten digits dataset (LeCun et al., 1998) and trained an spDAE with two hidden layers. Each layer had 4000 units and was pretrained with the sparsity regularization.

We trained a support vector machine (SVM) using libsvm (Chang & Lin, 2011) with a radial-basis function (RBF) kernel on the raw pixels of MNIST. As our interest is in a case where there are only noisy test samples available, we tried classifying, with the traind SVM, the noisy test set of MNIST after denoising them with the trained spDAE. The test set was corrupted with salt-and-pepper noise.

Additionally, we checked the effect of the simple sparsification on the discriminative properties of the latent representation. Two SVMs, again with the RBF kernel, were separately trained on the original latent representations and explicitly sparsified ones, respectively.

The robustness of the classification performance to the level of noise was measured by

$$m_p = \frac{\mathcal{E}_0}{\mathcal{E}_p},$$

where $\mathcal{E}_p$ is the classification error with the noise level p. The classifier with m_p dropping more slowly can be considered more robust to noise in the test samples.

In Table 2, we can clearly see that the proposed sparsification makes the classifier more robust to noise in the test samples. This is especially apparent as the level of noise increased. This experiment, albeit brief and simple, suggests that the proposed simple sparsification improves even the discriminative property of the latent representations.

Noise Level p		0.000	0.100	0.200	0.300	0.400
No Sparsification	m_p	1	0.861	0.565	0.217	0.106
	$\mathcal{E}_p$	0.032	0.037	0.057	0.148	0.304
Sparsification	m_p	1	0.873	0.615	0.252	0.117
	$\mathcal{E}_p$	0.035	0.039	0.056	0.137	0.295

(a)

p	0.000	0.100	0.200	0.300	0.400
m_p	1	0.393	0.071	0.035	0.025
$\mathcal{E}_p$	0.015	0.039	0.216	0.438	0.623
m_p	1	0.443	0.083	0.039	0.026
$\mathcal{E}_p$	0.016	0.035	0.189	0.399	0.591

(b)

Table 2. The robustness of the classification performance, measured by m_p, and the classification error $\mathcal{E}_p$, with varying noise levels. (a) The SVM was applied to the raw pixels denoised by the spDAE with and without the simple sparsification. (b) The SVMs were trained on the latent representations obtained by the spDAE with and without the simple sparsification.

This is interesting to notice that the classifier benefited from the feature extraction by the spDAE only when low level of noise (0 or 0.1) was injected to the test samples. When, the amount of noise was larger, we were able to see that the classification using the denoised raw pixels, especially with the proposed sparsification, outperformed that using the latent representations.

5. Discussion and Conclusion

A sparse denoising autoencoder (spDAE) learns an encoder that maps from a data space, defined by training samples corrupted with explicit noise, to a sparse latent space, defined by the sparsity regularization. A decoder of the spDAE, then, maps back from the sparse latent space to the data space. As its name suggests, an spDAE encodes a given noisy sample into a *sparse* latent representation and decodes the found representation into a *denoised* sample.

However, we noticed that the learned non-linear mappings, both encoder and decoder, are not well defined when a given sample is highly corrupted. In other words, if a given sample does not belong to the same data space, then the spDAE is not expected to denoise it well. Under this observation, we have proposed that explicit sparsification is necessary after the encoder is applied. For an spDAE trained with a sparsity regularization given in Eq. (2), a *simple sparsification* which makes the average latent activation closer to the target sparsity $\bar{\rho}$ was proposed.

Only very recently have *deep* neural networks, including denoising autoencoders, been applied to image denoising by Burger et al. (2012) and Xie et al. (2012). They all showed that these deep neural networks perform comparably to, or better than, conventional denoising methods. In this context, we have empirically shown that the proposed simple sparsification further improves denoising performance of deep neural networks. In addition to image denoising, we were able to see that the proposed sparsification could be used to improve even the discriminative performance when test samples were noisy.

In future, different sparsification methods should be sought for other available sparsity regularizers. Also, in the context of denoising, another type of datasets, such as speech, should be investigated in future.

References

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

Burger, H., Schuler, C., and Harmeling, S. Image denoising: Can plain neural networks compete with bm3d? In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 2392 –2399, june 2012.

Chang, C.-C. and Lin, C.-J. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011.

Chen, S. S., Donoho, D. L., and Saunders, M. A. Atomic decomposition by basis pursuit. *SIAM Rev.*, 43(1):129–159, January 2001.

Cho, K. Boltzmann machines and denoising autoencoders for image denoising. *ArXiv e-prints*, January 2013.

Elad, M. and Aharon, M. Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15 (12):3736–3745, December 2006.

Hyvärinen, A. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–34, 1999.

Hyvärinen, A., Hoyer, P., and Oja, E. Image denoising by sparse code shrinkage. In *Intelligent Signal Processing*. IEEE Press, 1999.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, Computer Science Department, University of Toronto, 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pp. 2278–2324, 1998.

Lee, H., Ekanadham, C., and Ng, A. Sparse deep belief net model for visual area v2. In Platt, J., Koller, D., Singer, Y., and Roweis, S. (eds.), *Advances in Neural Information Processing Systems 20*, pp. 873–880. MIT Press, Cambridge, MA, 2008.

Olshausen, B. A. and Field, D. J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, June 1996.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. Efficient learning of sparse representations with an energy-based model. In Schölkopf, B., Platt, J., and Hoffman, T. (eds.), *Advances in Neural Information Processing Systems 19*, pp. 1137–1144. MIT Press, Cambridge, MA, 2007.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Nature*, 323(Oct):533–536+, 1986.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, December 2010.

Xie, J., Xu, L., and Chen, E. Image denoising and inpainting with deep neural networks. In Bartlett, P., Pereira, F., Burges, C., Bottou, L., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 350–358. 2012.