Compressive learning of deep regularization for denoising

Hui Shi, Yann Traonmilin, Jean François Aujol

Compressive learning of deep regularization for denoising ⋆

Hui Shi[1*], Yann Traonmilin[1], and Jean-François Aujol[1]

[1]Univ. Bordeaux, Bordeaux INP, CNRS, IMB, UMR 5251,F-33400 Talence, France.
[*]hui.shi@u-bordeaux.fr

Abstract. Solving ill-posed inverse problems can be done accurately if a regularizer well adapted to the nature of the data is available. Such regularizer can be systematically linked with the distribution of the data itself through the maximum a posteriori Bayesian framework. Recently, regularizers designed with the help of deep neural networks (DNN) received impressive success. Such regularizers are typically learned from large datasets. To reduce the computational burden of this task, we propose to adapt the compressive learning framework to the learning of regularizers parametrized by DNN. Our work shows the feasibility of batchless learning of regularizers from a compressed dataset. In order to achieve this, we propose an approximation of the compression operator that can be calculated explicitly for the task of learning a regularizer by DNN. We show that the proposed regularizer is capable of modeling complex regularity prior and can be used for denoising.

Keywords: Regularization · Compressive learning · Denoising.

1 Introduction

We consider the denoising problem, i.e. finding an accurate estimate $\hat{x}$ of the original signal $x \in \mathbb{R}^d$ from the observed noisy signal $y \in \mathbb{R}^d$:

$$y = x + \epsilon, \tag{1}$$

where the noise ϵ (assumed to be additive white Gaussian noise of standard deviation σ) is independent of x. Recovering x from its degraded version y is an ill-posed problem and we needs to use additional (prior) information about the unknown signal x to obtain meaningful solutions. Common strategies [2] for solving inverse problems often define an estimator which minimizes

$$\hat{x} \in \arg\min_x F(x) + \lambda R(x), \tag{2}$$

where F is the data fidelity term making the solution consistent with the observation y and R is the regularization term weighted by $\lambda > 0$ that incorporates the prior information. The choice of R depends on the statistics of the signal of interest which is not always available in real-life applications.

⋆ This work was partly funded by ANR project EFFIREG - ANR-20-CE40-0001

The maximum a posteriori (MAP) Bayesian framework provides a useful tool to interpret such methods. The MAP estimator is given by:

$$\hat{x}_{\text{MAP}} \propto \arg\min_{x} \|y - x\|_2^2 - \lambda log(\mu(x)) \tag{3}$$

where μ denotes a prior probability law (of density $\mu(\cdot)$) of the unknown data x. In this context, the regularizer is related to the prior distribution of the data, i.e., $R(x) = -log(\mu(x))$.

It is not an easy task to accurately estimate a prior model, especially in high-dimensional spaces. Classical Bayesian approaches, e.g. in image processing, rely on explicit priors such as total variation or Gaussian mixture models (GMM) [22] trained on a database of image patches. Recently, researchers propose to use DNN to design the regularizer. Methods such as the total deep variation [11], adversarial regularizers [12,15], as well as the Plug & Play approach and its extensions [21,9] deliver remarkably accurate results. However, such models are typically learned from large datasets. Estimating their parameters from such a large-scale dataset is a serious computational challenge.

Compressive learning One possibility to reduce the computational resources of learning consists in using the compressive learning (CL) framework [4,5,7]. The main idea of CL, coined as sketching, is to compress the whole data collection into a fixed-size representation, a so-called *sketch* of data, such that enough information relevant to the considered learning task is captured. Then the learned parameters are estimated by minimizing a non-linear least-square problem built with the sketch. The size of sketch m is chosen proportional to the intrinsic complexity of the learning task. Meanwhile, the cost of inferring the parameters of interest from the sketch does not depend on the number of data in the initial collection but on the number of parameters we want to estimate. Hence, it is possible to exploit arbitrarily large datasets in the sketching framework without demanding more computational resources.

During the sketching phase, a huge collection of n d-dimensional data vectors $X = \{x_i\}_{i=1}^n$ is summarized into a single m-dimensional ($m \ll n$) vector $\hat{z}$ with:

$$\hat{z} = \frac{1}{n} \sum_{i=1}^{n} \Phi(x_i) = \mathcal{S}(\hat{\mu}_n), \tag{4}$$

where $\hat{\mu}_n := \frac{1}{n} \sum_{i=1}^n \delta_{x_i}$ the empirical probability distribution of the data, δ_{x_i} is the Dirac measure at x_i and the function $\Phi : \mathbb{R}^d \to \mathbb{R}^m$ is called the feature map (typically random Fourier moments). The operator $\mathcal{S}$ is a linear operator on measures μ defined by $\mathcal{S}\mu := \mathbb{E}_{X \sim \mu} \Phi(X)$. An estimate of a distribution μ (or of distributional parameters θ of interest) is computed by solving:

$$\mu_\theta^* = \arg\min_{\mu_\theta} \|\hat{z} - \mathcal{S}\mu_\theta\|_2^2. \tag{5}$$

In practice, this "sketch matching" problem can be solved by greedy compressive learning Orthogonal Matching Pursuit (OMP) algorithm and its extension

Compressive Learning-OMP with replacement [10]. When the distribution μ is a GMM in high-dimension with flat tail covariances, the problem can also be solved by the Low-Rank OMP algorithm (LR-OMP). It was shown that the prior model learned with LR-OMP can be used to perform image denoising [20].

These greedy algorithms are suitable for any sketching operator $\mathcal{S}$ and any distribution density μ, as long as the sketch $\mathcal{S}\mu$ and its gradient $\nabla_\theta \mathcal{S}\mu$ with respect to the distributional parameters θ of interest have a closed-form expression: the core of these OMP-based algorithms is computing the expression of $\mathcal{S}\mu$ and $\nabla_\theta \mathcal{S}\mu$. However, real-life data needs to be modeled with more complex distributions. In this case, the sketching feature map may not have a closed form. This limits the possible use of the sketching framework in practice.

In this paper, our goal is to recover a good approximation of the probability distribution of any unknown data from its sketch (i.e. beyond GMM). As neural networks (NN) have great expressive power [8,14], we propose to tackle the problems by adapting the sketching to NN. More precisely, we propose to define the regularizer R_θ parametrized by a DNN f_θ (precisely a ReLU network) as

$$R_\theta(\cdot) = \|f_\theta(\cdot)\|_2^2. \tag{6}$$

Such a regularization corresponds to the parametric distribution density $\mu_\theta \propto e^{-\|f_\theta(\cdot)\|_2^2}$. Thus it can be viewed as a generalized Gaussian distribution, where the bilinear form induced by the covariance matrix is replaced by a network. Due to the fact that NN have good generalization properties, the proposed regularization should be capable of encoding complex probability distributions. Unfortunately, a direct practical application of existing tools is not possible as closed-form expressions of $\mathcal{S}\mu$ are not available for sketching operator $\mathcal{S}$ based on random Fourier features.

Contributions and outline In this work, we show the feasibility of learning regularizers parametrized by a DNN from a compressed database. Once the network is trained, the regularizer can be used for denoising. To do so, we propose to approximate the sketching operator $\mathcal{S}$ by a discrete version $\mathcal{S}_d$ whose feature map can be calculated with closed-form expressions, and such that the approximation still permits to apply the sketch matching estimation method. The approximation is performed on a grid of the domain where the data is located.

To find an estimate of the distribution μ_θ, we adapt the sketch matching problem with our approximate sketching operator in the following way:

$$\theta^* \in \arg\min_{\theta \in \Theta} \|\mathcal{S}_d \mu_\theta(p) - \hat{z}\|_2^2 \,, \tag{7}$$

where Θ is a set were the DNN inducing the regularizer $R_\theta(\cdot) = \|f_\theta(\cdot)\|_2^2$ can be parametrized (i.e. weights and bias). This problem can be solved practically with gradient descent based methods.

As we do not need the original dataset during the training process, the training procedure does not need to build batches of data. As a consequence, *each gradient descent iteration in the training incorporates information from the whole*

original database. Once the empirical sketch has been computed (in a single pass, possibly in parallel), the dataset can be removed from memory. This reduces the memory complexity of the learning task. Moreover, the Jacobian $\nabla \mathcal{S}\mu_\theta$ can be computed efficiently with back-propagation.

Our approach overcomes the limits of greedy learning algorithms of the original sketching framework: regardless of the complexity of the data distribution, the proposed sketching operator allows us to always have a closed-form expression of $\mathcal{S}_d\mu_\theta$. Thus, the sketching is no longer limited to the distribution densities for which the Fourier transform is explicit.

As a result, the learned regularizer can be used to solve inverse problems. The effectiveness of the proposed scheme is tested on synthetic examples and real dataset. Due to the limitation of our approximation of the sketching operator (the dependence on training points), the feasibility is illustrated on 2-D and 3-D data with possibly complex distributions. Our work thus opens the broader open question of designing closed-form sketching operators in high dimension.

The rest of this article is organized as follows. We start by introducing the sketching framework, ReLU networks and some related works in section 2. In section 3, we describe the proposed framework: the adaptation of the compressive learning framework to the learning of regularizers parameterized by ReLU networks. Section 4 illustrates the performance of the proposed methods on both synthetic data and real-life data. Finally, conclusions are drawn in section 5.

2 Background, related works

We suppose that data samples x_i are modeled as independent and identically distributed random vectors having an unknown probability distribution with density $\mu \in \mathcal{D}$ ($\mathcal{D}$ is the set of probability measures over $\mathbb{R}^d$). We define the linear sketching operator $\mathcal{S}$ that maps μ to the m-dimensional sketch vector z:

$$\begin{aligned} \mathcal{S} : \mathcal{D} &\to \mathbb{C}^m \\ z = \mathcal{S}\mu &:= \int_{\mathbb{R}^d} \mu(x)\Phi(X)dx. \end{aligned} \tag{8}$$

When the transformation (sketching feature map) $\Phi(\cdot)$ is built with random frequencies of the Fourier transform, the l-th component of the sketch is

$$z_l = \int_{\mathbb{R}^d} e^{-j<\omega_l, x>}\mu(x)dx, \quad \text{for} \quad l = 1, \dots, m, \tag{9}$$

where $\{\omega_l\}_{l=1}^m \in \mathbb{R}^d$ are frequencies drawn at random. Taking a statistical perspective, the components z_l can be seen as samples of the characteristic function of $\mu_\mathcal{X}$. Accordingly, given a dataset $X = \{x_i\}_{i=1}^n$, the empirical sketch $\hat{z}$ can be computed from the samples of the database as

$$\hat{z}_l = \frac{1}{n}\sum_{i=1}^{n} e^{-j<\omega_l, x_i>}, \quad \text{for} \quad l = 1, \dots, m. \tag{10}$$

The compression ratio r is m/nd. It was shown [10,6,5] that when the probability distribution μ has a low dimension structure, e.g. a GMM, one can recover it (with high probability) from enough randomly chosen samples of its Fourier transform. The required size of the sketch is typically of the order of the number of parameters we need to estimate.

ReLU network A ReLU network, denoted by f_θ, is defined as a fully connected, feed-forward network (multi-layer perceptrons) with rectified linear unit (ReLU) activations. This activation has grown in popularity in feed-forward networks due to the success of first-order gradient based heuristic algorithms and the improvement in convergence to the approximated function for training [13].

Related works The sketching framework has been successfully applied to parametric models including GMMs [10,5,20], K-means clustering [10,5] and classification [17]. These methods are limited to the models for which the sketch function has a closed form. In our work, we apply the sketching to neural networks to encode more complex probability distributions. The sketching has been used for neural networks once [18]. In their work, the authors combine the sketching with generative networks to generate data samples, while in our work, we aim at learning a deep regularizer for solving the inverse problem. In addition, the authors of [18] proposed to approximate the sketching map by Monte-Carlo sampling. In our approach, we propose to do the approximation with a discrete sketching operator. The sketching framework mentioned above focus on data-independent approximation, i.e. the sketches are obtained by averaging random features. In [1], the authors propose to perform the sketching based on a Nyström approximation, the latter is data-dependent and shows empirically better performances for K-means clustering and Gaussian modeling.

3 Proposed method

In this Section, we explain how we adapt the sketching framework to estimate regularizations by DNN. We start by explaining why there are no explicit closed-form expressions of the sketching function available in the context of prior parametrized by DNN. Intuitively, since ReLU networks define piecewise affine functions, we can indeed express a ReLU network f_θ as:

$$f_\theta(x) = \sum_{\gamma=1}^{N_R} \mathbf{1}_{R_\gamma}(x)(W_\gamma x + b_\gamma), \tag{11}$$

where $\mathbf{1}_{R_\gamma}$ is the indicator function of each of the N_R affine regions R_γ, with parameters (W_γ, b_γ).

Given a dataset X, we aim at learning, from only the sketch z, an approximation μ_θ for the probability distribution μ generating X. We consider a regularizer of the form $R_\theta(\cdot) = \|f_\theta(\cdot)\|_2^2$ which corresponds to parametric densities of the

form $\mu_\theta(\cdot) \propto e^{-R_\theta(\cdot)}$. Ideally, with the definition in (9), the sketch would have to be calculated as

$$\begin{aligned} z_l &= \int_{\mathbb{R}^d} e^{-j<\omega,x>} e^{-\|f_\theta(x)\|_2^2} dx \\ &= \int_{x_d} \cdots \int_{x_1} e^{-j\sum_{p=1}^d \omega_p x_p} e^{-\sum_{p=1}^d (\sum_i^{N_R} \mathbf{1}_{R_i}(x)((W_i x)_p + b_{ip}))^2} dx_1 \cdots dx_d. \end{aligned} \tag{12}$$

However, to the best of our knowledge, there is no analytic expression of such Fourier transform (Fourier transform on polygons). To tackle this issue, we consider approximating the continuous Fourier transform on a set of discrete points.

To be specific, we define an approximation $\mathcal{S}_d : \mathbb{R}^d \to \mathbb{C}^m$ of the sketching operator $\mathcal{S}$ such that $\mathcal{S}_d\mu_\theta(\omega) \approx \mathcal{S}\mu(\omega)$ for a given frequency ω. The approximated sketch $\tilde{z}$ then has components:

$$\tilde{z}_l = |\Delta\Omega| \sum_{p_i \in \Omega} e^{-j<\omega_l, p_i>} e^{-\|f_\theta(p_i)\|_2^2}, \tag{13}$$

where p_i is a point in the d-dimensional cell Ω with volume $|\Delta\Omega|$. Of course, the major pitfall of this approximation is the limitation for applications in high dimension as the number of points is exponential with respect to the dimension d. The required boundedness (or approximate boundedness such as in the Gaussian case) of the data is a valid assumption in many practical applications in signal and image processing.

As a consequence, given N points $\{p_i\}_{i=1}^N$ on the grid where the dataset X lives and the empirical sketch defined as (4), we consider a ReLU network sketch matching problem as finding the network parameters θ^* in the set Θ of possible parametrizations, such that

$$\theta^* = \arg\min_{\theta \in \Theta} \|\mathcal{S}_d\mu_\theta - \hat{z}\|_2^2 . \tag{14}$$

With the discretization, if μ_θ is differentiable at point p_i, the gradient of $\mathcal{S}_d\mu_\theta$ with respect to the parameters θ can be computed easily by using the automatic differentiation. Note that the discretization is used only in the estimation of the regularizer from the sketch. It thus only impacts the calculation time and memory requirement of the estimation of the regularizer and not the size of the compressed dataset itself.

Denoising with prior With the learned regularization term, we solve the variational problem (3) which yields the minimization of:

$$G(x) = \|x - y\|_2^2 + \lambda \|f_\theta(x)\|_2^2. \tag{15}$$

The optimization problem can be solved by gradient descent based methods. Similarly, we can compute the gradient by using automatic differentiation. Also, note that this denoising method can easily be extended to other linear inverse problems, such as interpolation and deconvolution by including the corresponding forward measurement operator.

4 Experiments

Synthetic data We first test our framework with 2-D and 3-D synthetic data. To illustrate the advantage of the compressive learning framework in terms of (computational) learning times, the used training datasets are made of $n = 10^6$ samples which are generated from: a spiral with parameters: the radius of circular curve $R = 0.3$ to 1, spiral length $L = 2\pi$; and a zero-mean GMM of 2 Gaussians. The source code is available at [19] which contains parts of code taken from the Python Compressive Learning toolbox [16]. To train the network, we use the Adam optimizer with a learning rate of value 10^{-6}. The number of points on the grid is set to $N = 20^d$ where d denotes the data dimension. For comparison, we propose to learn the regularizer on the non compressed dataset using the same network with the following learning objective function:

$$\theta' = \arg\min_{\theta} \sum_{i=1}^{N} \left\| \|f_\theta(p_i)\|_2^2 - dist_i \right\|_2^2, \tag{16}$$

where $dist_i = \min_j \|x_j - p_i\|_2^2$ is the distance between the data x_j and its nearest grid point. This objective function imposes a regularizer that approximates the function "distance to the model". Note that for non compressive learning, as we do not go through a (implicit) model of the density, we explicitly give the distance value, which is not necessary when using our proposed sketched method.

For the 2-D experiments, the network f_θ is designed as a ReLU network with 3 fully connected hidden layers with 64, 128, and 256 neurons in each layer respectively. Figure 1 shows the experimental results for 2-D synthetic data. The first column shows the training spiral and GMM samples. The results shown in the $2nd$ column are models learned from 4000-fold compressed dataset while producing comparable results to those learned from a non-compressed dataset ($3rd$ column), indicating that our approach achieves its goal: efficiently learn prior probability densites from compressed datasets (which will be evaluated when used as a regularization on real audio data). In fact, we match the distribution density of data directly in the compressive method which is not trivial for the non-compressed method.

Table 1 shows the needed learning times (in hours) with respect to the different sketch size $m = 50, 100, 1000, 5000$, i.e. with compression ratio $r = 40000, 20000, 2000, 400$ respectively. We see that training the same dataset, the non compressed learning takes much longer (20 times) than the compressive learning. Meanwhile, the proposed compressive learning approach is capable of recovering good approximations of the probability distribution of sample data.

Denoising results The learned regularizers are evaluated on the denoising of white Gaussian noise. The noisy dataset is made of 500 samples generated with noise level σ^2. We choose the optimal hyper-parameter values (the learning rate η and the regularizer parameter λ) for each model. Figure 4 visually illustrates the 2-D denoising results using regularizers learned from the compressed dataset

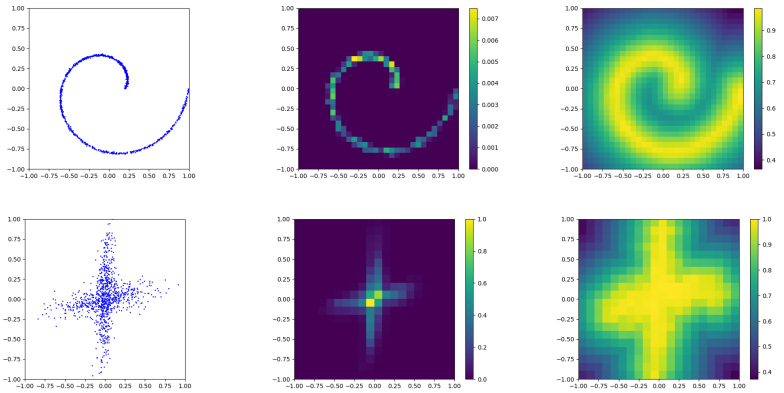

Fig. 1. The distribution densities of sample data (left) learned from compressed dataset with sketching (compression ratio $r = 4000$)(middle) and original dataset (right).

Table 1. Table of leaning times with respect to the compression ratio: sketch used for training is r times smaller than the original dataset (results in bold).

		sketching				non
	r	40000	20000	2000	400	compressed
Time	Spiral	0.19h	0.23h	0.28h	0.72h	**28.7h**
	GMM	0.23h	0.18h	0.24h	0.73h	**28.5h**

4000 times smaller (left) and the original dataset (right). Figure 3 shows the 3-D denoising results with different noise levels.

We assess the effect compression ratio, *i.e.* the sketch size, in Table 2 and use the signal-to-noise ratio (SNR) to evaluate the effectiveness of our method. We generate 10 different noisy datasets of 500 samples for each data type with noise level $\sigma^2 = 0.15$. Table 2 shows the average gain in SNR with respect to the priors learned with different compression ratios. It is shown that the proposed approach is capable to learn distributions from databases with high compression ratios, and that regularizers trained from the compressed datasets have similar denoising performance compared to regularizers trained on their original non

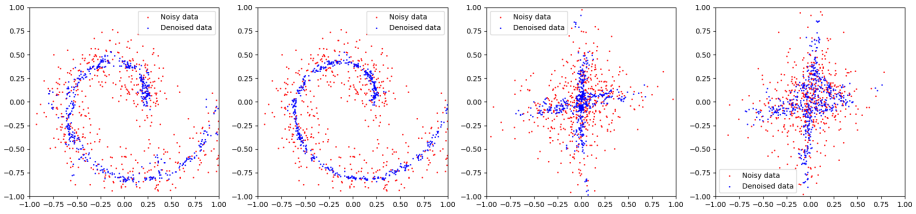

Fig. 2. Denoising results ($\sigma^2 = 0.15$) with regularizers learned with the compressed dataset with compression ratio $r = 4000$ (**a**, **c**) and the original dataset (**b**, **d**).

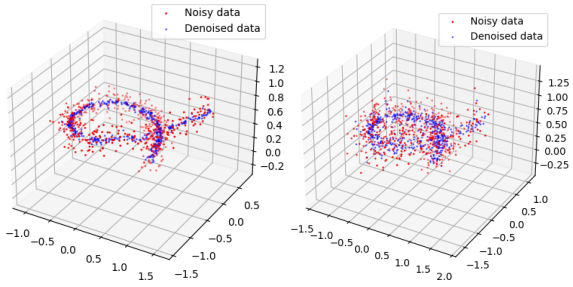

Fig. 3. 3-D Denoising results ($\sigma^2 = 0.1$ (left) and $\sigma^2 = 0.2$ (right)) with densities learned from the proposed method with a compression ratio $r = 3000$.

compressed datasets. When we reduce the sketch size, the denoising performance drops slightly.

Table 2. Table of reconstruction loss with respect to the compression ratio: sketch used for training is r times smaller than the original dataset.

		sketching				non
	r	40000	20000	2000	400	compressed
SNR Gain	Spiral	1.37	1.89	1.92	2.31	2.61
	GMM	0.80	0.86	1.57	1.82	1.77

The number of frequencies used to compute sketches affects memory storage, while the number of grid points used during training process affects the learning time. The number of grid points should be chosen well if we want to control the learning time well, since the number of grid points N grows exponentially with respect to the dimension d. Experiments from figure 4 also show that overparameterized networks (with more layers or more neurons per layer) can achieve better results while using less learning time and fewer necessary grid points.

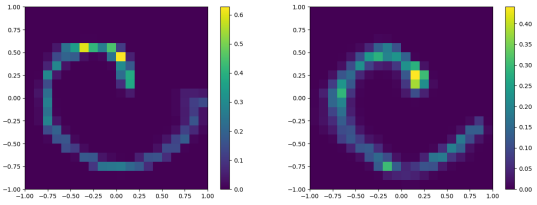

Fig. 4. Regularizers learned via the proposed method with different learning parameters. Using the same number of grid points, we have better result when the network has more neurons. (Left) 3 hidden layers with 64, 128, 256 neurons in each layer. (Right) 3 hidden layers with 64, 128, 192 neurons in each layer.

Robustness to noise during training We train the regularizer on a compressed dataset of data samples generated with noise level $\sigma^2_{train} = 0.15$ using the same

network and training procedure as described above. Figure 5 shows the learned distribution density and the denoising result. The result shows that the regularizer trained with a compressed noisy dataset has good denoising performance. This illustrates that our approach is robust to low noise level. This is easy to understand due to the fact that adding Gaussian noise corresponds to a convolution of the density with a Gaussian kernel, which does not change the shape of the distribution if small enough. It is even possible to add a deconvolution term to the distribution parameter estimation if the noise level in the training dataset is known.

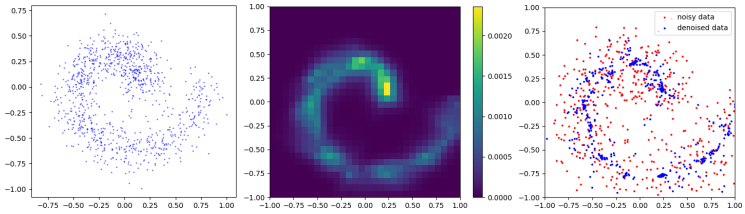

Fig. 5. Denoising result (right, $\sigma^2 = 0.2$) with density (middle) learned from the compressived noisy dataset (left, $\sigma^2_{train} = 0.15$).

Application to audio denoising We perform experiments on recorded musical notes (monophonic 16kHz audio snippets) from the NSynth dataset [3]. The training data is an extracted 0.125s audio recorded from an acoustic guitar. After filtering the normalized audio data s by two 4th-order Butterworth low-pass filters h_1 and h_2 with a cutoff frequency of 1.5kHz and 3.75kHz, three frequency responses are constructed with $s_1 = h_1 * s$, $s_2 = h2 * (s - s_1)$, and $s_3 = s - s1 - s2$. Then the frequency responses are concatenated, hence the training set is of dimension 2000×3; i.e. 2000 samples in dimension 3. The regularizer is learned from a sketch of size $m = 200$, *i.e.* the dataset is compressed by a factor of 30. Once the regularizer is learned, it is used to denoise the audio corrupted by Gaussian white noise of noise levels $\sigma^2 = 0.1$ and $\sigma^2 = 0.2$.

Figure 6 and 7 show the audio denoising results with different noise levels. In the two cases, we gain more than 1dB on SNR in the case of small noise and more than 2.5dB in case of large noise. Similar denoising results (gain of 1dB in the small noise and 1.94dB in the case of large noise) are obtained from the priors learned from the non compressed approach with 3 times slower training time. The results show that, in addition to the original low-pass filtering effect, denoising can be achieved in low dimensions even when temporal consistency between individual samples is not guaranteed. These first feasibility results for solving inverse problem using regularizers learned from sketch are promising, especially if it is possible to extend this frramework to high dimension.

5 Conclusions

In this work, we illustrate the feasibility of adapting the compressive learning framework to the learning of a regularizer parameterized by a DNN. We achieve

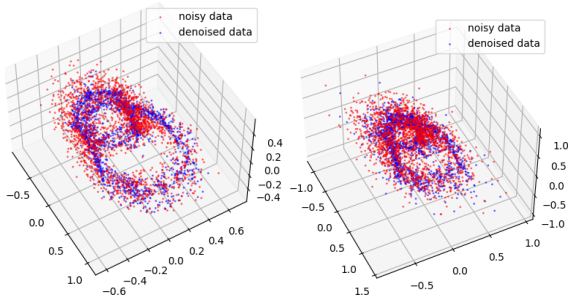

Fig. 6. Audio denoising performances for different noise levels: (left) $\sigma^2 = 0.1$, SNR is 10.03 for noisy data and 11.39 for denoised data; (right) $\sigma^2 = 0.2$, SNR is 4.01 for noisy data and 6.64 for denoised data.

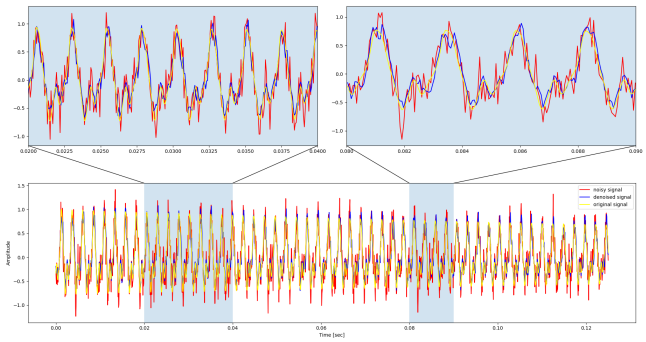

Fig. 7. Audio denoising performance for noise level $\sigma^2 = 0.2$.

this by approximating the original sketching operator with a discrete one. With the proposed approximated sketching operator, the "sketch matching" problem can be solved with a gradient based algorithm. In addition, we define a new parametrization of the regularizer to solve the inverse problem. The regularizer is defined as the squared ℓ^2-norm of a ReLU network and learned with a compressive dataset instead of the original dataset. It gathers the advantages of sketching which reduces the learning cost and of NN which have great expressive power. Experiment results on 2-D/3-D synthetic data and audio data show that our method accomplishes the objective of compressive learning, illustrating the potential of sketched NN. However, our method relies on a discretization of the domain on which the data resides, which limits its use in high-dimensional domains (*e.g.*, for image denoising). Future works will be needed to overcome this limitation. We want to design a fast sketching operator that avoids such discretization. This leads to a major open question: can we find a sketching operator such that any distribution parametrized by a DNN can be estimated from the sketch? Does such a sketching operator exist? A positive answer to these questions could drastically change the way deep priors are trained.

References

1. Chatalic, A., Carratino, L., De Vito, E., Rosasco, L.: Mean Nyström embeddings for adaptive compressive learning. In: Int. Conf. on Artificial Intelligence and Statistics. pp. 9869–9889. PMLR (2022)
2. Demoment, G.: Image reconstruction and restoration: Overview of common estimation structures and problems. IEEE Transactions on Acoustics, Speech, and Signal Processing **37**(12), 2024–2036 (1989)
3. Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D., Simonyan, K.: Neural audio synthesis of musical notes with wavenet autoencoders. In: Int. Conf. on Machine Learning. pp. 1068–1077. PMLR (2017)
4. Gribonval, R., Blanchard, G., Keriven, N., Traonmilin, Y.: Compressive statistical learning with random feature moments. Mathematical Statistics and Learning **3**(2), 113–164 (2021)
5. Gribonval, R., Blanchard, G., Keriven, N., Traonmilin, Y.: Statistical learning guarantees for compressive clustering and compressive mixture modeling. Mathematical Statistics and Learning **3**(2), 165–257 (2021)
6. Gribonval, R., Chatalic, A., Keriven, N., Schellekens, V., Jacques, L., Schniter, P.: Sketching datasets for large-scale learning (long version). arXiv preprint arXiv:2008.01839 (2020)
7. Gribonval, R., Chatalic, A., Keriven, N., Schellekens, V., Jacques, L., Schniter, P.: Sketching data sets for large-scale learning: Keeping only what you need. IEEE Signal Processing Magazine **38**(5), 12–36 (2021)
8. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks **2**(5), 359–366 (1989)
9. Hurault, S., Leclaire, A., Papadakis, N.: Gradient step denoiser for convergent plug-and-play. arXiv preprint arXiv:2110.03220 (2021)
10. Keriven, N., Bourrier, A., Gribonval, R., Pérez, P.: Sketching for large-scale learning of mixture models. Information and Inference **7**(3), 447–508 (2018)
11. Kobler, E., Effland, A., Kunisch, K., Pock, T.: Total deep variation: A stable regularization method for inverse problems. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–1 (2021)
12. Lunz, S., Öktem, O., Schönlieb, C.B.: Adversarial regularizers in inverse problems. Advances in Neural Information Processing systems **31** (2018)
13. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Int. Conf. on Machine Learning (2010)
14. Pan, X., Srikumar, V.: Expressiveness of rectifier networks. In: Int. Conf. on Machine Learning. pp. 2427–2435. PMLR (2016)
15. Prost, J., Houdard, A., Almansa, A., Papadakis, N.: Learning local regularization for variational image restoration. In: Int. Conf. on Scale Space and Variational Methods in Computer Vision. pp. 358–370. Springer (2021)
16. Schellekens, V.: Pycle: a python compressive learning toolbox. (may 2020), https://doi.org/10.5281/zenodo.3855114
17. Schellekens, V., Jacques, L.: Compressive classification (machine learning without learning). arXiv preprint arXiv:1812.01410 (2018)
18. Schellekens, V., Jacques, L.: Compressive learning of generative networks. arXiv preprint arXiv:2002.05095 (2020)
19. Shi, H.: https://github.com/shihui1224/sketching-deep-regu-for-denoising
20. Shi, H., Traonmilin, Y., Aujol, J.F.: Compressive learning for patch-based image denoising. SIAM Journal on Imaging Sciences **15**(3), 1184–1212 (2022)

21. Venkatakrishnan, S.V., Bouman, C.A., Wohlberg, B.: Plug-and-play priors for model based reconstruction. In: IEEE Global Conf. on Signal and Information Processing. pp. 945–948. IEEE (2013)
22. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: Int. Conf. on Computer Vision. pp. 479–486. IEEE (2011)