Learning Structured Signals Using GANs with Applications in Denoising and Demixing

Mohammadreza Soltani
Iowa State University

Swayambhoo Jain
Technicolor AI Labs

Abhinav V. Sambasivan
University of Minnesota

Chinmay Hegde
Iowa State University

***Abstract*—Recently, Generative Adversarial Networks (GANs) have emerged as a popular alternative for modeling complex high dimensional distributions. Most of the existing works implicitly assume that the clean samples from the target distribution are easily available, which is not true in real-world applications. In this paper, we consider the observation setting when the samples from target distribution are given by the superposition of two structured components and leverage GANs for learning the structure of the components. We propose two novel frameworks: denoising-GAN and demixing-GAN. The denoising-GAN assumes access to clean samples from the second component and try to learn the other distribution, whereas demixing-GAN learns the distribution of the components at the same time. Through extensive numerical experiments, we demonstrate that proposed frameworks can generate clean samples from unknown distributions, and provide competitive performance in tasks such as denoising and demixing.**

I. Introduction

In this paper, we consider the classical problem of separating two structured signals observed under the following superposition model:

$$Y = X + N, \tag{1}$$

where $X \in \mathcal{X}$ and $N \in \mathcal{N}$ are the *constituent signals/components*, and $\mathcal{X}, \mathcal{N} \subseteq \mathbb{R}^p$ denote the two structured sets. In general the separation problem is inherently ill-posed; however, with enough structural assumption on $\mathcal{X}$ and $\mathcal{N}$, it has been established that separation is possible. Depending on the application, one might be interested in estimating only X (in this case, N is considered as the corruption), which is referred to as *denoising*, or in recovering both X and N which is referred to as *demixing*. Both demixing and denoising arise in a variety of important practical applications in the areas of signal/image processing, computer vision, machine learning, and statistics [1]–[4]. Most of the existing techniques assume some prior knowledge on the structures of $\mathcal{X}$ and $\mathcal{N}$ in order to recover the desired components. However, this is a limitted assumption in the real applications. In this paper, we consider the problem of separating constituent signals from superposed observations. In particular, we are given a set of superposed observations $\{Y_i = X_i + N_i\}_{i=1}^K$, where $X_i \in \mathcal{X}$ and $N_i \in \mathcal{N}$ are i.i.d samples from their respective (unknowns) distributions. In this setup, we explore two questions: First, *How can one learn prior knowledge about the individual components from superposition samples?* Second, *Can we leverage the implicitly learned constituent distributions for tasks such as denoising and demixing?*

Part of this work has been accpted in Deep Generative Models for Highly Structured Data workshop in ICLR 2019, New Orleans, LA.

A. Setup and Our Contribution

Motivated by the recent success of generative models in high-dimensional statistical inference tasks such as compressed sensing in [5], [6], in this paper, we focus on Generative Adversarial Network (GAN) based generative models to implicitly learn the distributions, i.e., generate samples from their distributions. Most of the existing works on GANs typically assume access to clean samples from the underlying signal distribution. However, this assumption breaks down in the superposition model considered in our setup, where the structured superposition makes training generative models very challenging.

In this context, we investigate the first question with varying degrees of assumption about the access to clean samples from two signal sources. We first focus on the setting when we have access to samples only from the constituent signal class $\mathcal{N}$ and observations, Y_i's. In this regard, we propose the *denoising*-GAN framework. However, assuming access to samples from one of the constituent signal class can be restrictive and is often not feasible in real-world applications. Hence, we further relax this assumption and consider the more challenging demixing problem, where samples from the second constituent component are not available. To solve the latter, we propose *demixing*-GAN framework. Finally, to answer the second question, we use our trained generator(s) from the proposed GAN frameworks for denoising and demixing tasks on unseen test samples (i.e., samples not used in the training process) by discovering the best hidden representation of the constituent components from the generative models. In addition to the denoising and demixing problems, we also consider a compressive sensing setting (whicb is deferred to a longer version of this paper due to the lack of space [7]) to test the trained generator(s).

II. Application and Prior Art

To overcome the inherent ambiguity issue in problem (1), many existing methods have assumed some prior knoeledge on the structures of components and also that the signals from $\mathcal{X}$ and $\mathcal{N}$ are "distinguishable" enough [8]–[13]. The assumption of having the prior knowledge is a big restriction in many real-world applications. While some recent attempts such as structured sparsity [14], dictionary learning [8], and

in general manifold learning have tried to automate this *hard-coding* approach, they still cannot fully address the need for the prior structure. Over the last decade, deep neural networks have been demonstrated to learn useful representations of real-world signals such as natural images, and thus have helped us understand the structure of the high dimensional signals, for e.g., using deep generative models [15]. Our approach in this paper is based on implicitly learning the distribution of constituent components using Generative Adversarial Networks (GANs) [16]. GANs have been established as a very successful tool for generating structured high-dimensional signals [17], [18] from distributions lying on a low-dimensional manifolds.

In most of the existing works on GANs with few notable exceptions [6], [19]–[22], it is implicitly assumed that one has access to clean samples of the desired signal. However, in many practical scenarios, the desired signal is often accompanied by unnecessary components. Recently, GANs have also been used for capturing of the structure of high-dimensional signals specifically for solving inverse problems such as sparse recovery, compressive sensing, and phase retrieval [5], [20], [21]. Specifically, [5] have shown that generative models provide a good prior to structured signals, for e.g., natural images, under compressive sensing settings over sparsity-based recovery methods. Furthermore, authors in [23] have proposed a projected gradient descent algorithm for solving the recovery problem directly in the ambient space (space of the desired signal) based on the prior knowledge about the structured signals provided by GAN. As mentioned, most of the prior art in GANs need direct access to the clean samples from the unknown distribution, which is not the case in many real applications. AmbientGAN framework [6] partially addresses this problem by studying a various measurement models. It showed that the GAN can find samples of clean signals from corrupted observations. However, AmbientGAN assumes that the measurement model and parameters are known, which is a very strong and limiting assumption in real applications. One of our main contributions is addressing this limitation by studying the demixing problem. In addition, the approach in AmbientGAN just learns the distribution of the clean images; however, it has not been used for the task of image denoisng. Our framework addresses this issue as well.

III. Background and the Proposed Idea

A. Background

Generative Adversarial Networks (GANs) have been initially introduced by [16] for generating samples from an unknown target distribution. This is done through a zero-sum game between two players, *generator*, G and *discriminator*, D through the following *min-max* optimization problem:

$$\min_{\theta_g} \max_{\theta_d} \; \mathbb{E}_{x\sim\mathcal{D}_x}[log(D_{\theta_d}(x))]\mathbb{E}_{z\sim\mathcal{D}_z}[log(1-D_{\theta_d}(G_{\theta_g}(z)))],$$

where θ_g and θ_d are the parameters of generator networks and discriminator network, respectively. Also, $\mathcal{D}_x$ and $\mathcal{D}_z$ represents the target probability distributionthe and distribution of the hidden variables $z \in \mathbb{R}^{h1}$. It has been shown that if G and D have enough capacity, then solving the above optimization problem by alternative stochastic gradient descent algorithm guarantees the distribution $\mathcal{D}_g$ at the output of the generator converges to $\mathcal{D}_x$. Next we present the proposed modifications to the basic GAN setup that allows for usage of GANs as a generative model for denoising and demixing structured signals.

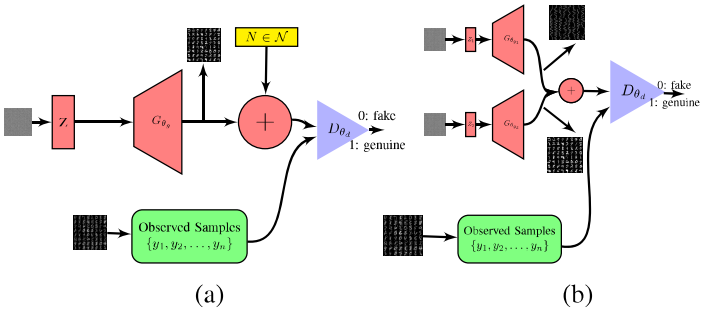

Fig. 1: The architecture of proposed GANs. (a) denoising-GAN. (b) demixing-GAN.

B. denoising-GAN

Our idea is inspired by AmbientGAN due to [6] in which they used a regular GAN architecture to solve some inverse problems such as inpainting and denoising from unstructured noise by assumption that the corruption model is known. The idea is to feed discriminator with observed samples, y_i's (distributed as Y) together with the output of generator G which is corrupted by the known corruption model. Our denoising-GAN framework is illustrated in Figure 1(a). This framework is similar to one proposed in AmbientGAN paper; however, denoising-GAN is used to denoise from a structured corruption (not just arbitrary noise). To this end, we use our assumption that the components have some structure and the representation of this structure is given by the last layer of the trained generator, i.e., $X \in G_{\widehat{\theta}_g}$[2]. This observation together with this fact that in GANs, the low-dimension random vector z is representing the hidden variables, leads us to this point: in order to denoise a new test image, we have to find a hidden representation, giving the smallest distance to the corrupted image in the space of $G_{\widehat{\theta}_g}$ [5], [23]. In other words, we have to solve the following optimization problem:

$$\widehat{z} = \arg\min_{z} \|u - G_{\widehat{\theta}_g}(z)\|_2^2 + \lambda\|z\|_2^2, \tag{2}$$

where u denotes the corrupted test image. The solution of this optimization problem provides the (best) hidden representation for an unseen image. Thus, the clean image can be reconstructed by evaluating $G_{\widehat{\theta}_g}(\widehat{z})$. While optimization problem (2) is non-convex, we can still solve it by running gradient descent algorithm in order to get a stationary point[3].

[1]One can also use identity function instead of $log(.)$ function in the above expression. The resulting formulation is called WGAN [24].

[2]$G_{\widehat{\theta}_g}(.)$ denotes the trained generator network with parameter $\widehat{\theta}_g$.

[3]While we cannot guarantee the stationary point is a local minimum, but the empirical experiments show that gradient descent (implemented by backpropagation) can provide a good quality result.

C. demixing-GAN

Now, we go through our main contribution, demixing. Figure 1(b) shows the GAN architecture, we are using for the purpose of separating or demixing of two structured signals form their superposition. As illustrated, we have used two generators and have fed them with two random noise vectors $z_1 \in \mathbb{R}^{h_1}$ and $z_2 \in \mathbb{R}^{h_2}$ according to a uniform distribution defined on a hyper-cube, where h_1, h_2 are less than the dimension of the input images. We also assume that they are independent of each other. Next, the output of generators are summed up and the result is fed to the discriminator along with the superposition samples, $y_i's$. Somewhat surprisingly, this architecture based on two generators can produce samples from the distribution of each component after enough number of training iterations. We note that this approach is fully unsupervised as we only have access to the mixed samples and nothing from the samples of constituent components is known. As mentioned above, this is in sharp contrast with AmbientGAN and our previous structured denoising approach. As a result, the demixing-GAN framework can generate samples from the second components, which further can be used in the task of denoising where the corruption components are sampled from highly structured signal. So, we d not need to know the corruption model. Now similar to the denoising-GAN framework, we can use the trained generators in Figure 1(b), for demixing of the constituent components for a given test mixed image by solving the following (demixing) optimization problem and obtaining $\widehat{z_1}, \widehat{z_2}$:

$$\underset{z_1,z_2}{\arg\min} \|y - G_{\widehat{\theta}_{g_1}}(z_1) - G_{\widehat{\theta}_{g_2}}(z_2)\|_2^2 + \lambda_1\|z_1\|_2^2 + +\lambda_2\|z_2\|_2^2,$$

where u denotes the test mixed image. Now, each component can be estimated by evaluating $G_{\widehat{\theta}_{g_1}}(\widehat{z_1})$ and $G_{\widehat{\theta}_{g_2}}(\widehat{z_2})$ (denoting the parameters of the first and second trained generators). While this optimization problem is non-convex, we can still solve it through block coordinate gradient descent algorithm, or in a alternative minimization fashion. Finally, we have provided some theoretical intuitions for the denoising-GAN and demixing-GAN in the arXiv version of this paper [7].

IV. Numerical Experiments

In this section, we present various experiments showing the efficacy of the proposed frameworks (depicted in Figure 1(a) and Figure 1(b)) in different setups. First, we will focus on the denoising from structured corruption in the testing scenario (please see [7] for the training part). Next, we focus on demixing signals from structured distributions. Here, we present our results on two sample datasets, MNIST [25] and F-MNIST [26]. Due to lack of space, we defer the details of experiments setup, the complementary experiments on the compressive sensing scenario, experiments on the other datasets (SVHN [27], and Quick-Draw [28]), and emrically investigating the failiure of demixing-GAN to [7].

A. Structured Corruption Models

For all the experiments on MNIST dataset, we have used the network architectures for discriminator and generator(s) similar to the one proposed in DCGAN [29]. For the corruption part, we have used two structured noise models similar to [30]. In the first one, we generate random vertical and horizontal lines and add them to the dataset. The second structured noise is constructed based on random sinusoidal waves in which the amplitude, frequency, and phase are random numbers. We define the level of corruption (lc) as the number of sinusoidal or lines added to the original image.

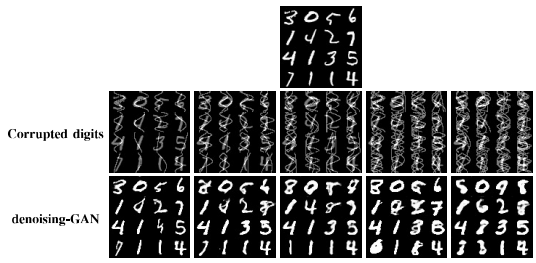

Fig. 2: The performance of trained generator for various *level of corruption (lc)* in denoising of unseen images. Top row: Ground truth digits. Middle row: corrupted digits with random sinusoidal, with a level of corruption from 1 to 5. Bottom row: the reconstructed digits.

B. Denoising from Structured Corruption

In this section, we use GAN architecture illustrated in Figure 1(a) for removing of the structured noise. The setup of the experiment is as follows: we use 55000 MNIST images with size 28×28 corrupted by either of the above corruption models[4]. The resulting images, y_i's are fed to the discriminator. We also use hidden random vector $z \in \mathbb{R}^{100}$ drawn from a uniform distribution in $[-1, 1]^{100}$ for the input of the generator. Once the denoising-GAN is trained (i.e., obtaining $\widehat{\theta}_g$), it can be used to denosie unseen test images. For reconstructing of the clean images, we solve the optimization problem in (2) to obtain solution $\widehat{z}$. Then we find the reconstructed clean images by evaluating $G_{\widehat{\theta}_g}(\widehat{z})$. In Figures 2, we have used the different level of corruptions for sinusoidal corruption model. Also, we vary the level of corruption from 1 to 5 with random sines. The result of $G_{\widehat{\theta}_g}(\widehat{z})$ has been shown below of each level of corruption.

As we can see, even with heavily corrupted images (level corruption equals to 5), GAN is able to remove the corruption from unseen images and reconstruct the clean digits. Furthermore, we evaluate the quality of reconstructed images compared to the corrupted ones through a classification task. Please see [7] for more details.

C. Demixing of Structured Signals – Training

In this section, we present the results of our experiments for demixing of the structured components. To do this, we use the proposed architecture in Figure 1(b). We first present our experiment with MNIST dataset, and then we show the similar set of experiments with Fashion-MNIST dataset (F-MNIST) which includes 60000 training 28×28 gray-scale

[4]Here, we just present the random sinusoidal corruption. Please see [7] for the other experiment. Also, we set level of corruption 1 for sinusoidal waves.

images with 10 labels. The different labels denote objects, including T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boot.

1) Experiments on MNIST Dataset: We start the experiments with considering two sets of constituent components. In the first one, we use random sinusoidal waves as the second constituent component. Compared to the denosing case, here we are interested in generating of the samples from both parts. In Figure 3, we show the training evolution of two fixed random vectors, z_1 and z_2 in $\mathbb{R}^{100}$ in the output of two generators. As we can see, our proposed GAN architecture can learn two distributions and generate samples from both digits and random sines.

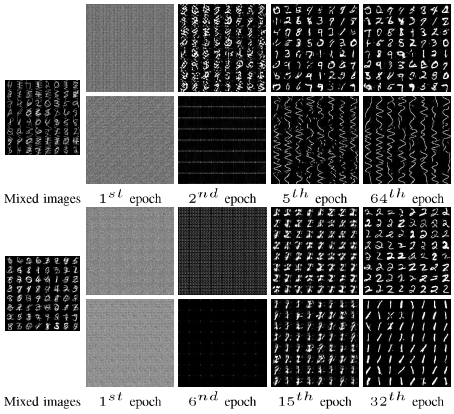

Fig. 3: Evolution of output samples by two generators for fixed z_1 and z_2. The top panel shows the evolution of the two generators in different epochs where the mixed images comprise of digits and sinusoidal. The first generator learns the distribution of MNIST digits, while the second one is learning the random sinusoidal waves. The bottom panel shows the same experiment withe the mixed images comprise only digits 1 and 2.

In the second scenario, we consider the mixed images as the superposition of digits 1 and 2. In the training set of MNIST dataset, there are around 6000 samples from each digits of 1 and 2. In this case, we are interested in learning the distributions from which digits 1 and 2 are drawn. Thus, we have used these digits to form the set of superposition images. The bottom panel of Figure 3 shows the output of two generators, which can learn the distribution of the two digits. The interesting point is that these experiments show that each GAN can learn the existing digit variety in MNIST training dataset, and we typically do not see mode collapse, which is a major problem in the training of GANs [31].

2) Experiments on F-MNIST Dataset: In this section, we illustrate the performance of the proposed demixing-GAN for F-MNIST dataset. Similar to the experiment with MNIST digits experiment, we train the demixing-GAN, where we have used InfoGAN [32] architecture for the generators. The dimension of input noise to the generators is set to 62. Figure 4 shows the output of two generators (mixed images comprise of dress and bag images from the training set), which can learn the distribution of dress and bag images during 21 epochs.

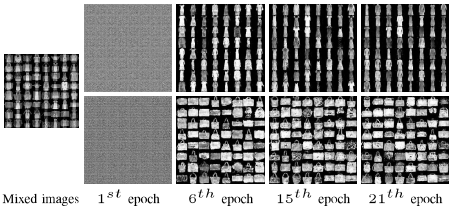

Fig. 4: Evolution of output samples by two generators for fixed z_1 and z_2. The mixed images comprise only two objects, dress, and bag in training F-MNIST dataset. One generator produces the samples from dress distribution, while the other one outputs the samples from the bag distribution.

D. Demixing of Structured Signals – Testing

Similar to the test part of denoising, in this section, we evaluate the performance of two trained generators in a demixing scenario for the mixed images, which have not been seen in the training time. Figure 5 shows the demixing on three different mixed images. Here, we have compared the performance of demixing-GAN with *Independent component analysis (ICA)* method [33]. In the top and middle rows of Figure 5, we consider the mixed images generated by adding a digit (drawn from MNIST test dataset) and a random sinusoidal. To demix these components from each other, we use trained GAN for learning the distribution of digits and sinusoidal waves (the top panel of Figure 3) and solve the demixing optimization problem through an alternative minimization fashion. The corresponding constituent components are then obtained by evaluating $G_{\widehat{\theta}_{g_1}}(\widehat{z_1})$ and $G_{\widehat{\theta}_{g_2}}(\widehat{z_2})$. As we can see, our proposed GAN can separate two digits; however, ICA method fails in demixing of two components.

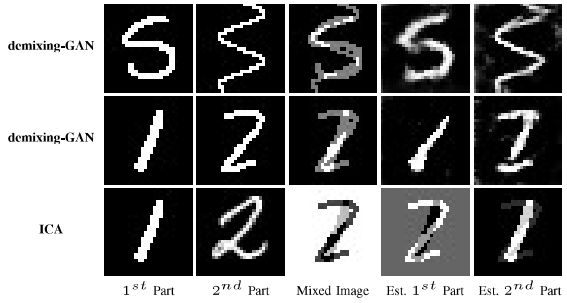

Fig. 5: The performance of trained generators for demixing of two constituent components. The first two columns are the ground-truth components. The third column is the ground-truth mixed image and the last two columns denote the recovered components. The first row uses the same generator trained for only one digit (drawn from MNIST test dataset) and a random sinusoidal. The second row uses the generator trained only for digits 1 and 2. The last row shows the result of demixing with ICA method.

Nest, we evaluate the performance of the trained demixing-GAN on the F-MNIST dataset. In Figure 6, we have illustrated a similar experiment to the previous case. The first row uses the generator trained for only two objects for 20 epochs. The second row uses the same generator trained for only two objects for 30 epochs. The last row shows the result of demixing with ICA

method. As we can see, ICA fails to separate the components (images of F-MNIST) from each other, while the proposed demixing-GAN can separate the mixed images from each other.

Finally, as an attempt to understand the condition under which the demixing-GAN is failed, we empirically investigated the role of two aspects of demixing-GAN. First, it seems that the z-space of the generators for characterizing the distribution of the constituent components play an essential role in the success/failure of the demixing-GAN. Second the incoherence of the hidden structures in the generator space determines to what extent demixing is feasible. We investigate these observations through some numerical experiments in [7]..

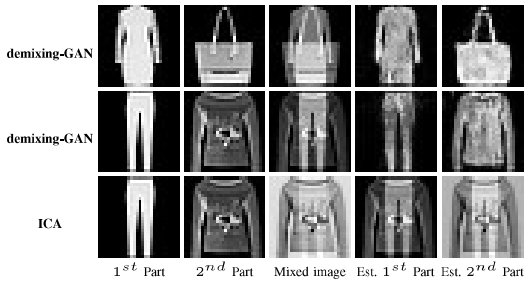

Fig. 6: The performance of trained generators for demixing of two constituent components. The first two columns are the ground-truth components. The third column is the ground-truth mixed image and the last two columns denote the recovered components. The first row uses the generator trained for only two objects for 20 epochs. The third row uses the generator trained for only two objects for 30 epochs. The last row shows the result of demixing with ICA method.

V. Conclusion

We studied the GANs framework for learning the structured signals in denoising and demixing applications. We empirically showed that it is possible to implicitly learn the underlying distribution of the structured components even in the superposition model. We also used these learned structures for the downstream task such as denoising and demixing of unseen images. Further theoretical investigation regarding the more involved demixng scenario is a future research direction.

References

[1] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[2] M. Elad, J. Starck, P. Querre, and D. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)," *Appl. Comput. Harmonic Analysis*, vol. 19, no. 3, pp. 340–358, 2005.

[3] J. Bobin, J. Starck, J. Fadili, Y. Moudden, and D. Donoho, "Morphological component analysis: An adaptive thresholding strategy," *IEEE Trans. Image Proc.*, vol. 16, no. 11, pp. 2675–2681, 2007.

[4] E. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM*, vol. 58, no. 3, p. 11, 2011.

[5] A. Bora, A. Jalal, E. Price, and A. Dimakis, "Compressed sensing using generative models," *Proc. Int. Conf. Machine Learning*, 2017.

[6] A. Bora, E. Price, and A. Dimakis, "Ambientgan: Generative models from lossy measurements," in *Int. Conf. on Learning Rep. (ICLR)*, 2018.

[7] M. Soltani, S. Jain, and A. Sambasivan, "Learning generative models of structured signals from their superposition using gans with application to denoising and demixing," *arXiv preprint arXiv:1902.04664*, 2019.

[8] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Proc.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[9] M. Soltani and C. Hegde, "Fast algorithms for demixing sparse signals from nonlinear observations," *IEEE Transactions on Signal Processing*, vol. 65, no. 16, pp. 4209–4222, 2017.

[10] ——, "Iterative thresholding for demixing structured superpositions in high dimensions," *NIPS Workshop on Learning. High Dimen. Structure (LHDS)*, 2016.

[11] J. Druce, S. Gonella, M. Kadkhodaie, S. Jain, and J. Haupt, "Defect triangulation via demixing algorithms based on dictionaries with different morphological complexity," in *Proc. European Workshop on Struc. Health Monitoring (IWSHM)*, 2016.

[12] M. Elyaderani, S. Jain, J. Druce, S. Gonella, and J. Haupt, "Group-level support recovery guarantees for group lasso estimator," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP)*, 2017, pp. 4366–4370.

[13] S. Jain, U. Oswal, K. Xu, B. Eriksson, and J. Haupt, "A compressed sensing based decomposition of electrodermal activity signals," *IEEE Trans. Biom. Eng.*, vol. 64, no. 9, pp. 2142–2151, 2017.

[14] C. Hegde, P. Indyk, and L. Schmidt, "Approximation algorithms for model-based compressive sensing," *IEEE Trans. Inform. Theory*, vol. 61, no. 9, pp. 5129–5147, 2015.

[15] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," *arXiv preprint arXiv:1711.10925*, 2017.

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2014, pp. 2672–2680.

[17] D. Berthelot, T. Schumm, and L. Metz, "Began: boundary equilibrium generative adversarial networks," *arXiv preprint arXiv:1703.10717*, 2017.

[18] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2016, pp. 613–621.

[19] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2016, pp. 82–90.

[20] M. Kabkab, P. Samangouei, and R. Chellappa, "Task-aware compressed sensing with generative adversarial networks," *Proc. Assoc. Adv. Artificial Intelligence (AAAI)*, 2018.

[21] P. Hand, O. Leong, and V. Voroninski, "Phase retrieval under a generative prior," *arXiv preprint arXiv:1807.04261*, 2018.

[22] J. Zhu, P. Krähenbühl, E. Shechtman, and A. Efros, "Generative visual manipulation on the natural image manifold," in *Proc. European Conf. Comp. Vision (ECCV)*, 2016, pp. 597–613.

[23] V. Shah and C. Hegde, "Solving linear inverse problems using gan priors: An algorithm with provable guarantees," *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP)*, 2018.

[24] M. A., S. C., and L. B., "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Machine Learning*, 2017, pp. 214–223.

[25] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[26] X. Han, R. Kashif, and V. Roland, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint cs.LG/1708.07747*, 2017.

[27] T. Netzer, Y.and Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, vol. 2011, no. 2, 2011, p. 5.

[28] "The quick, draw! dataset," https://github.com/googlecreativelab/quickdraw-dataset.

[29] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[30] G. Chen and S. Srihari, "Removing structural noise in handwriting images using deep learning," in *Proc. Indian Conf.e on Comp. Vision Graph. Image Proc.* ACM, 2014.

[31] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.

[32] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2016.

[33] P. Hoyer *et al.*, "Independent component analysis in image denoising," *Master degree dissertation, Helsinki University of Technology*, 1999.