Manuscript submitted to
AIMS' Journals
Volume **X**, Number **0X**, XX **200X**

doi:10.3934/xx.xxxxxxx

pp. **X–XX**

GENERATIVE IMAGING AND IMAGE PROCESSING VIA GENERATIVE ENCODER

Yong Zheng Ong

National University of Singapore
21 Lower Kent Ridge Rd
Singapore, Singapore 119077

Haizhao Yang

Purdue University
610 Purdue Mall
West Lafayette, IN 47907, USA

(Communicated by the associate editor name)

Abstract. This paper introduces a novel generative encoder (GE) framework for generative imaging and image processing tasks like image reconstruction, compression, denoising, inpainting, deblurring, and super-resolution. GE unifies the generative capacity of GANs and the stability of AEs in an optimization framework instead of stacking GANs and AEs into a single network or combining their loss functions as in existing literature. GE provides a novel approach to visualizing relationships between latent spaces and the data space. The GE framework is made up of a pre-training phase and a solving phase. In the former, a GAN with generator G capturing the data distribution of a given image set, and an AE network with encoder E that compresses images following the estimated distribution by G are trained separately, resulting in two latent representations of the data, denoted as the generative and encoding latent space respectively. In the solving phase, given noisy image $x = \mathcal{P}(x^*)$, where x^* is the target unknown image, $\mathcal{P}$ is an operator adding an addictive, or multiplicative, or convolutional noise, or equivalently given such an image x in the compressed domain, i.e., given $m = E(x)$, the two latent spaces are unified via solving the optimization problem

$$z^* = \underset{z}{\operatorname{argmin}} \|E(G(z)) - m\|_2^2 + \lambda \|z\|_2^2$$

and the image x^* is recovered in a generative way via $\hat{x} := G(z^*) \approx x^*$, where $\lambda > 0$ is a hyperparameter. The unification of the two spaces allows improved performance against corresponding GAN and AE networks while visualizing interesting properties in each latent space.

1. **Introduction.** Deep learning-based structures have become an effective tool for image processing. A stream of such studies is an end-to-end training with an Autoencoder (AE) [1, 2] mapping source images to reconstructed images with desired properties. Due to the powerful representation capacity of AEs, AEs can approximate the desired imaging or image processing procedure well as long as

2020 *Mathematics Subject Classification.* Primary: 68U10; Secondary: 68T07.

Key words and phrases. Generative Adversarial Network, Autoencoder, Inverse Problem, Latent Space, Image Reconstruction, Denoising, Deblurring, Super Resolution, Inpainting.

The second author is supported by NSF grant DMS-1945029.

training data are sufficiently good. In particular, the dimension reduction property in AEs play a key role in enhancing its performance, similar in importance to sparsity in traditional image processing algorithms.

AEs are designed with 2 Neural Networks (NN). The encoder, $E(x;\theta_E)$ with parameters θ_E, adaptively captures the low-dimensional structure of a source image x through repeated applications of convolution, pooling, and nonlinear activations, to output a small feature vector z; whilst the decoder, $DC(z;\theta_{DC})$ with parameters θ_{DC}, efficiently reconstruct x via repeated deconvolution, up-sampling, and nonlinear activations acting on z. Parameters in E and DC are jointly tuned such that the input and output of the AE match via the below optimization problem

$$\min_{\theta_E,\theta_{DC}} \mathbb{E}_{x\sim p_{data}(x)} \left[\|DC(E(x;\theta_E);\theta_{DC}) - x\|_2^2\right], \tag{1}$$

where p_{data} is the image data distribution. Armed with a powerful representation capacity, AEs are capable of learning nonlinear transforms which create highly sparse and low-dimensional representations of images, while maintaining the accuracy of image reconstruction. However, due to the least square nature of (1), AEs penalize pixel-wise error and hence tends to generate smooth images, instead of learning to generate fine details. A recent paper [3] explores this limitation in AEs, and this paper will also demonstrate such phenomena experimentally.

Another stream of deep learning approaches [4, 5, 6, 7, 8, 9] are based on generative adversarial networks (GANs) [10] and its variants [11, 12, 13, 14]. GANs consists of a generator G and discriminator D pair which are trained adversarially. G, with parameters θ_G, takes random vector z from a given distribution p_z and output a synthetic sample $G(z;\theta_G)$. D, with parameters θ_D, takes as input x and outputs a value $D(x;\theta_D) \in [0,1]$ denoting the probability that the input x follows the data distribution p_{data}. In simple terms, G is trained to generate samples to fool D into thinking that the generated sample is real, while D learns to distinguish between real samples from the data distribution versus synthetic fake data from G via the below adversarial game

$$\begin{aligned}\min_{\theta_G}\max_{\theta_D} V(\theta_D,\theta_G) &= \mathbb{E}_{x\sim p_{data}(x)}[\log D(x;\theta_D)] \\ &+ \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z;\theta_G);\theta_D))]\end{aligned} \tag{2}$$

D in the adversarial learning above can be applied to enhance image quality as in [4]. G can also be applied as a tool for data augmentation [15, 16] or as an inverse operator [5, 17, 18, 19, 20, 21] that returns the desired image from applications with compressed measurements like compressed sensing. However, solving (2) or its variants is challenging and the solution might not be stable, e.g. GANs are prone to mode collapse or vanishing gradient problems, although generated content are better in quality of fine details than those generated by AEs, which tends towards smooth images.

A common point in both types of generative network lies in a mapping between a lower-dimensional latent/feature space and the original data space. In particular, the difference in objective of both leads to a different representation of data in the latent spaces. This leads to a particular question of whether the latent spaces can be combined, to gain the advantages of both. Present architectures do so either using a single latent space to represent the encoding and the generating space, like in VAEGAN [22, 23] and [24, 25], or train the losses jointly [26, 27]. This paper introduces a novel generative encoder (GE) framework that takes advantage of both AEs and GANs separately for generative imaging tasks. Instead of current methods

which merge GAN and AE training, an alternative method to link both latent spaces is proposed through solving an invert GAN problem, in which recent works have begun exploring [17, 18, 20]. Empirical evidence shows improved performance in applications like image reconstruction, denoising, inpainting, deblurring, and super-resolution. The framework also introduces a novel approach towards the study and understanding of latent spaces, which recent works [18] has investigated.

The GE model consists of a pre-training and a solving phase. In the former, a GAN and an AE are separately trained. As the G in GAN captures the data distribution of a given image set, generated images in a GAN can be used as training data to augment the training of the AE component. Hence, the GAN is trained first in the pre-training phase, followed by augmenting generated data with training data to train the AE. Depending on the application, the AE can be further augmented with noisy data, and trained to filter such noise [28]. In the solving phase, given a noisy image $x^\dagger = \mathcal{P}(x^*)$, where x^* is the target unknown image, $\mathcal{P}$ is an operator adding an additive, multiplicative, or convolutional noise, the below optimization problem is solved

$$z^* = \arg\min_z \|E(S(G(z))) - E(x^\dagger)\|_2^2 + \lambda\|z\|_2^2 \tag{3}$$

and x^* is recovered via $\hat{x} := G(z^*) \approx x^*$. Here S denotes a down-sample or an up-sample operator balancing the dimension of the output of G and the input of E, or noise added to the generated image, if the structure of the noise is known, like in inpainting. Else, S is the identity map. Finally, $\lambda > 0$ is a hyper-parameter. GE unifies the generative capacity of the generative space in GANs and the stability and rough detail capturing properties of the encoding space in AEs via an optimization framework (3) instead of stacking GANs and AEs into a single network or training the two together. Instead of learning an encoding and a generating network into the same latent space, GE explores the benefits of using an optimization network to learn a relationship between the two spaces, while each maximizes its own advantages.

1.1. **Related Work.** This section briefly introduces related work in the field of invert GAN. Lipton & Tripathy [19] described a direct optimization method to invert a fixed generator G, which can be summarized to solving the below equation

$$z^* = \arg\min_z d(G(z), x) \tag{4}$$

Here d denotes some distance function, for example $l2$-norm. The image x is then recovered by $G(z^*)$. Similar work is done in [20] to solve the above invert GAN problem using different alternatives for d. Recent work by [17, 21] proposes alternative gradient/LP based optimization methods to invert compressed sensing or deep generative models using ReLU activations, but is not applicable when model architecture is much more complex, like progressive GAN (pGAN) [29]. [18] proposed the use of a third network to learn the inverse mapping, by training the third network, denoted as $G^{-1}(x, \theta_{G^{-1}})$ via the objective

$$\arg\min_{\theta_{G^{-1}}} d_1(G^{-1}(G(z)), z) + d_2(G(z), x) \tag{5}$$

Their method, while using an "encoder" like structure in terms of G^{-1}, differs from GE in terms of G^{-1} learning an inverse of G, while in GE, there is no direct relation between E and G. In fact, the dimensions of the two latent spaces in GE need not be the same, which allows for greater flexibility and generalization.

1.2. **Contributions.** This section summarizes the contributions of GE:

a. **Training of G and E in GE is done separately.** To the best of our knowledge, GE is the first framework that performs the training separately. The motivation in doing so is to reduce competition between the different objectives involved in training a GAN and AE. To illustrate this, the separation of the two models in a denoising problem distributes the generative component to the generative space and the dimension reduction and filtering of noise component to the encoding space, which differs from existing methods that combine both spaces together. This not only allows each model to maximise their performance on the corresponding objectives, but also allows the use of the same pre-trained GAN (since it is not trained to filter noise), which in general is harder to train, for multiple applications of imaging problems, through switching of the AE component. Numerical results are presented to demonstrate this capability and to show that GE outperforms end-to-end convolutional AEs and the original invert GAN proposed by [19] which uses only a generator. Empirical results also show the benefits of separating losses, by comparing with a network that merges both training losses, which will be denoted as AEGAN.

b. **E learns a different set of latent vector compared to G.** This is in contrast with [18] where the third network is trained to learn and reproduce the latent distribution of G, or in present GAN and AE frameworks like AE, VAEGAN, etc, which uses the same feature space between their G and E. GE is the first framework which utilizes a unification method to learn relationships between the two learnt latent spaces. Numerical results in Section 5.5 will demonstrate the importance of this idea. This section introduces the differing objectives of the GAN and AE objectives, in which GANs are shown to favour more prominent features for a realistic generation, while AE focuses on reconstructing as many features as possible. GE, with the separation of training, allows two different sets of latent vectors to be learnt, each maximising their corresponding objectives, while using a solving phase to draw upon the benefits of both.

c. **Training of AE is augmented by G**, thus the dimension reduction of AE adapts to the target data distribution instead of training data only, making the AE more compatible with data distribution and increasing the generalization flexibility of AE. This is in contrast with existing methods, where the training of the AE and GANs are done simultaneously, thus removing any possibility for such a style of data augmentation.

d. Instead of creating an end-to-end neural network, an optimization problem (3) is proposed to search for the best reconstruction that fits data measurements in the compressed domain. On one hand, **E stabilizes the reconstruction via reducing the search domain**, filtering out key features that should be generated by G, based on how E is trained to handle and compress information for the image processing task. On the other hand, **G that focuses on producing realistic images improves the reconstruction quality often lost in traditional AEs** that prefers smooth images. The latter property, in particular, is lost in the merging of AE and GAN training, due to the addition of the ℓ^2 type loss from AE, and G observing training data directly instead of only through DC's output.

e. A direct implication of using a solving phase allows the final output image to be generated from G rather than the decoder of an AE, DC. This is motivated by

the fact that AEs uses a distance type loss function which results in smoother outputs. In comparison, GANs have been known to generate more realistic images. Using a solving phase allows GE to produce realistic images, while using the AE to perform task specific filtering. This is supported with numerical results, which shows the distinct difference between using a GAN as output (models like InvertGAN and GE), against using a decoder from an AE (models labelled ConvAE and AEGAN). The importance of E during the solving phase is apparent in complex applications, like Inpainting, where without it, the model totally fails.

f. **GE is a general framework with various applications**. Numerical experiments show that the proposed model can produce competitive or better outcomes in these fields. Furthermore, GE is **flexible in the choice of GAN or AE used** and can be constantly improved using newer GAN or AE methods, according to the problem.

g. [18] introduced the idea that learning an inverse generator allows for an effective method in visualizing what a GAN cannot generate. In this regard, **GE presents a novel framework towards learning and understanding relationships between different latent spaces**, by visualizing the information gained and lost across two such spaces. This provides answers to certain interesting questions, which will be elaborated on shortly.

2. Applications.

This section briefly introduces the applications of GE covered in this paper.

2.1. Image Reconstruction/Compressed Sensing.

Given measurement $y = Ax^* + \epsilon$, where A is a sensing matrix satisfying the restricted isometry property (RIP), ϵ is a noise vector, the compressed sensing problem seeks to recover x^*. If x^* is sparse, the recovery via an ℓ^1-penalized least square problem is guaranteed by the compressed sensing theory in [30, 31]. The RIP condition is satisfied when A is a random Gaussian matrix, and natural images are generally sparse after an appropriate transform, e.g., wavelet transform. Therefore, compressed sensing has always been a successful tool in imaging science.

Pioneer works including [5, 17] have explored the application of generative models to improve traditional compressed sensing algorithms. The main idea is to apply G to generate a synthetic image $G(z)$ from z in a compressed space. In present image processing applications, the matrix A is often replaced with E, and the problem is solved as an AE.

2.2. Denoising and Inpainting.

Denoising and inpainting have the same problem statement in mathematics. Given a measurement $y = x^* + \epsilon$ or $y = x^* \circ \epsilon$, where ϵ is a certain random or structured noise and $\circ$ represents the Hadamard product, denoising and inpainting seek to recover x^* from y. In this paper, Gaussian random noise with a standard deviation of $\sigma = 0.5$ is used for denoising, while images are partially masked by a 64×64 block in randomly chosen positions in inpainting.

Traditional denoising or inpainting techniques generally rely on the sparsity of x^* after an approximate transform in a certain metric, e.g., KSVD [32], GSR [33], BM3D [34], NLM [35], and total variation (TV) regularization [36, 37]. Deep learning approaches have become more popular than traditional methods recently, e.g., AE methods [1, 38, 39], especially when hidden information of noisy or damaged

images is not visually obvious. Recently, invert GAN based methods [17] have also been suggested for inpainting.

2.3. **Deblurring.** Blurring an image is commonly modeled as the convolution of a point-spread function over an original sharp image. For example, this paper uses a rotationally symmetric Gaussian lowpass filter to produce blurred images. Deblurring aims to reverse this process. Mathematically speaking, given a measurement $y = x * h$, where h is an unknown convolution kernel function and $*$ represents the convolution operator, deblurring seeks to recover x with certain assumptions on h and x to ease the ill-posedness. Sparse coding [40] and kernel estimation [41] are effective methods for image deblurring. CNNs have also been applied to this problem recently [42], especially with the help of GAN [14]. Notice, that in this application, GAN which has higher detail generation are used in these existing works over AEs.

2.4. **Super-resolution.** Super-resolution aims to generate high-resolution images from low-resolution ones. For example, given a measurement $y = S(x^*)$, where S is a down-sampling operator or a convolution operator with a convolution kernel function decaying quickly in the Fourier domain, super-resolution seeks to recover x^* with certain assumptions on S and x^* to ease the ill-posedness. Traditionally, interpolation methods (Bicubic, Nearest Neighbours) and sparse-coding [40] are popular tools to increase image resolution. Recently, CNNs have also been applied to solve this problem with great success [43, 44].

For the experiments, original images are downsampled to $32 \times 32 \times 3$. Instead of inputting this image into GE directly, the image is first reconstructed by upsampling the image using the nearest neighbor interpolation as input $x^\dagger$ in (3), which is equivalent to using the upsampling interpolation as a preconditioner of the optimization problem.

3. **Generative Encoder Framework.** This section introduces the detailed implementation of GE. In this paper, GE is trained using pGAN as the GAN and a convolutional AE as the AE. In general, GE is broadly compatible with various GANs and AEs. The overall training procedure of the GE model is summarized in Algorithm 1 and visualized in Figure 1.

Algorithm 1 GE: generative encoder model

1: Pre-train a generator using any GAN.
2: Pre-train an AE using both real and fake images (generated by GAN in Step 1).
3: Take the generator G of GAN and the encoder E of AE to form the generative encoder.
4: Given a measurement $m = E(x^\dagger)$, find $z^* = \arg\min_z ||E(S(G(z))) - m||_2^2 + \lambda \|z\|_p^2$ and return $\hat{x} = G(z^*)$.

The key idea in GE is to combine G and E from a GAN and AE model respectively. This takes advantage of each model using a new optimization framework in (3) to unify latent spaces. The first term $d(m, z) = ||E(S(G(z))) - m||_2^2$ itself serves as the loss function in the solving phase. The ℓ^p term is introduced to regularize the highly nonconvex function $d(m, z)$. This choice could vary depending on how the latent space is represented in G (e.g. $N(0, 1)$). Besides, searching for a solution with a small ℓ^2-norm also agrees with most GANs that has $z \sim \mathcal{N}(0, I)$.

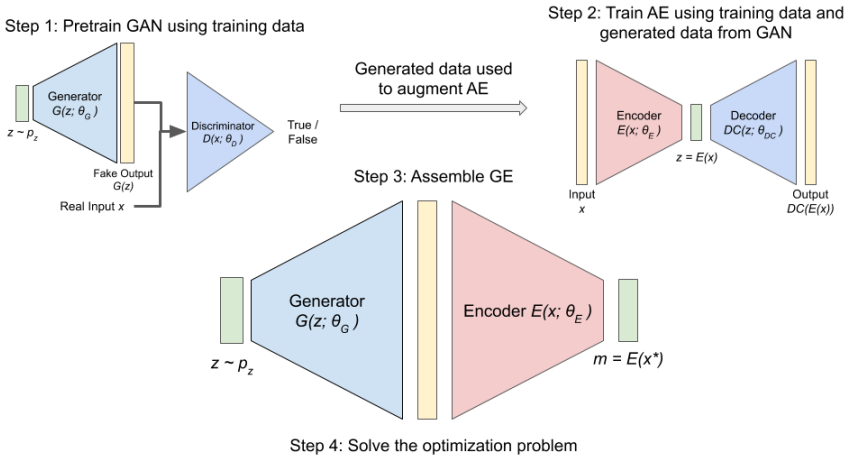

FIGURE 1. Flow of training process in GE. Step 1 and 2 forms the pre-training phase, while the remaining form the solving phase

Finally, $\lambda > 0$ is a hyperparameter denoting the weight of the regularization term. An interesting extension would be replacing the ℓ^2 regularization with a NN for a data-driven regularization. This is left as future work.

3.1. **Visualizing and understanding the relationship between latent spaces.** Understanding latent spaces play an important role in NNs, especially in GAN and AE which rely on a good representative space to perform. However, it is difficult to visualize what is represented in the latent spaces, as well as derive mappings between them. As a result, current applications in generative models like GAN, AE, VAEGAN, etc relies on a single latent space. For example, in AE, E maps an input to a feature representation, and DC reconstructs output from the same space. [18] explores this concept to learn what a generative space cannot represent. GE also seeks to address this objective, to provide answers to questions like

a. Are certain features avoided during generation, or are they just underrepresented?
b. Can a model be encouraged to produce specific features without skewing the training data? [18]
c. What information is prioritized in the different latent space? What gains/losses are there in learning a mapping between each other?

Under the same set of data, different models produce largely different representations of the data. Such differences are difficult to visualize. GE leverages the use of inverse GAN problem with an optimization framework to explore relationships between any pair of generative and encoding space, by providing a way to visualize this phenomenon, to answer questions like above.

Consider a scenario that in optimizing (4), generated samples $G(z^*)$ are noticed to remove certain details. A valid suspicion would be that G avoids producing such details, perhaps due to lack of appearance in training data, or to prioritize learning a higher frequency feature. Now consider optimizing (3). Two possible scenarios could be observed:

a. **The feature appears:** This suggests that G learned to produce the above detail, but is underrepresented in the generative space. The same detail is instead better represented in E. Optimizing G with the help of latent space E magnifies these underrepresented features being expressed, even without skewing the training data.
b. **The feature does not appear:** This suggests that both E and G does not prioritize in learning the above feature. In this case, these features are likely to be avoided by the models during the learning of each latent space.

On the other hand, suppose a feature disappears in GE as opposed to (4). This in turn suggests a loss of information from G to E, that is, the feature is underrepresented by E, while largely represented by G. GE's capability to visualize these phenomenon allows one to gather statistical evidence to answer questions like above. A method to visualize the results will be demonstrated in the experiments (via Truth Tables and Visual Inspection).

3.2. **Loss Functions.**

3.2.1. *Image Reconstruction.* Suppose the original image is x^* and a sensor E is given. Given the measurement $m = E(x^*)$, to find the reconstruction $\hat{x}$ that is as close to x^* as possible, GE finds z in the latent space such that the compressed measurement of its generated image, i.e., $E(S(G(z)))$, is as close to the measurement of the real image m as possible. Therefore, Algorithm 1 is solved with N as the identity map to identify $\hat{x} := G(z^*) \approx x^*$ as the reconstruction.

3.2.2. *Denoising, Deblurring, Super-Resolution and Inpainting.* The solution of the denoising, deblurring, super-resolution, and inpainting can be obtained by solving Algorithm 1 with their respective noise maps S, which is an identity for denoising and deblurring, a masking operator for inpainting, and a dimension adjustment operator for super-resolution. Let $x^\dagger$ be the given noisy image constructed from an unknown target image x^*. Then the reconstructed image is set as $\hat{x} = G(z^*) \approx x^*$.

The AE is also trained to filter the noise, like in denoising AEs [28]. Training of the AE in Step 2 of Algorithm 1 is modified to include noisy samples as data augmentation, where in addition to training on reconstructing images, an additional term $\|AE(x^\dagger) - x\|_2^2$ is added so that the AE matches noisy images against their corresponding clean image, thus acting as a filter for GE.

4. **Training Details.** This section describes training details for the experiments in this paper.

4.1. **Training Data.**

For the experiments, the following datasets are used. First, the CelebA dataset [45] is used and the following applications are explored on the dataset: Image Reconstruction, Denoising, Inpainting, Deblurring, and Super-resolution. This dataset contains more than $200,000$ celebrity images cropped to size $128 \times 128 \times 3$. A further split of the dataset to training and testing data is performed, so that the performance of GE are tested with data not used in training.

Next, a digital rock dataset containing micro computed tomography (μCT) images of a dry Bentheimer sandstone outcrop [46] is used and the following applications are explored on the dataset: Image Reconstruction. The choice of using digital rock images is due to the need for high quality results during the image processing process. In particular, such rock images contains sharp edges and sediments that does not favour the use of conventional AEs, which produces smooth outputs. The

3D μCT image is first partitioned into 2D slices along varying axis, and crops of size $256 \times 256 \times 1$ are obtained from the centre of the resulting slices.

The third dataset is obtained from the LSUN church dataset [47] and the following applications are explored on the dataset: Image Reconstruction. This dataset is preprocessed by resizing all images to size $256 \times 256 \times 3$. The train images are used for training, while the provided validation images are used in testing.

4.2. **GAN.** pGAN is trained with training data above and the corresponding G is adopted. pGAN model used is obtained from the official Github of the pGAN paper [29]. The same G and weights are used throughout the different experiments. This provides evidence to motivate the separation of training GAN and AE - in that the G focuses on maximizing the generation capacity, as the same pre-trained GAN model presents a similar generated quality of results for different problems, without a need to retrain the GAN to handle the different tasks.

4.3. **AE.** GE is tested using a deep convolutional AE. The structure of the AE is based on ensuring that the number of trainable parameters in the AE and pGAN is similar. This provides a fair comparison in terms of model size for the experiments. Table 1 describes the structure of E used in ConvAE, while DC is the reverse of E using conv_transpose and upsample, with a tanh output activation to match the output of pGAN [29]. The input shape is $[-1, 128, 128, 33]$. k, s, a and h describes the kernel, stride, choice of activation and the fully connected layer's output size respectively in the table. $f = 32$ in our experiments.

layer type	**layer**
conv2d	$k = [3, 3, 3, f], s = [1, 1], a = ReLU$
maxpool2d	$k = [1, 2, 2, 1], s = [2, 2]$
conv2d	$k = [3, 3, f, 2 * f], s = [1, 1], a = ReLU$
maxpool2d	$k = [1, 2, 2, 1], s = [2, 2]$
conv2d	$k = [3, 3, 2 * f, 4 * f], s = [1, 1], a = ReLU$
maxpool2d	$k = [1, 2, 2, 1], s = [2, 2]$
conv2d	$k = [3, 3, 4 * f, 8 * f], s = [1, 1], a = ReLU$
maxpool2d	$k = [1, 2, 2, 1], s = [2, 2]$
conv2d	$k = [3, 3, 8 * f, 16 * f], s = [1, 1], a = ReLU$
maxpool2d	$k = [1, 2, 2, 1], s = [2, 2]$
conv2d	$k = [3, 3, 16 * f, 32 * f], s = [1, 1], a = ReLU$
maxpool2d	$k = [1, 2, 2, 1], s = [2, 2]$
fullyconnected	$h = 256$

TABLE 1. Structure of E. The decoder DC is a mirror of E using conv_transpose and upsample.

Note that the size of the latent space dimension of the GAN and AE used in the experiments are not the same. E maps $128 \times 128 \times 3$ images to vectors of length 256, while G maps noise vectors of length 512 to images of size $128 \times 128 \times 3$. This design is intentional, to demonstrate how GE differs from similar structured models like [18] in which the choice of an additional neural network (NN), E, is added not to learn the inverse mapping of G, but rather as a feature extraction and filtering tool, based on the original function of E in traditional AEs.

Together, the combination of G of pGAN and E in the AE is denoted as GE.

4.4. **Optimization in GE.** In the solving phase, ADAM optimizer [48] is used with $lr = 0.1$ to solve (3). z is initialized using the random start of the same distribution as input distribution in pGAN, and training is conducted for 1400 iterations. For the experiments, the same random start and ADAM is used across models to ensure a fair comparison.

4.5. **Baseline Methods.** As GE is a framework which is able to use different GAN and AE, the main focus for comparison are the below objectives

a. The performance of GE is compared with existing methods similar in objective [19, 20] and model size, as well as an AE, AEGAN which merge AE and GAN training
b. Numerical evidence is demonstrated to motivate the discussion in Section 3.1

Should some existing SotA methods involving AE or GAN could outperform the choice of pGAN or AE, the corresponding components of that method could always be adopted to achieve similar improvement of results. This will be demonstrated by a side experiment, through building GE with an alternate GAN, BEGAN [14], instead of pGAN.

Baseline methods are chosen and modified to have the same G or E structure as used in the choice of pGAN and AE for a fair comparison. Below lists the baseline methods which will be used for comparison in this paper under the same conditions and training set:

a. *Convolutional AE.* The same AE architecture (Table 1) in GE is used. This allows for comparing between the benefits of using G from a GAN as a replacement to DC in AEs, which favors smooth images and lacks finer details. This also provides a means to compare between GE, which trains G and E separately, to AEs which trains both the generator/decoder and encoder simultaneously. This model is denoted as ConvAE.
b. *Invert GAN.* This benchmark uses the proposed optimization method by [19], by solving the optimization problem (4) using ℓ^2 norm as d. As mentioned, other works like [20, 17] are based upon this same problem with different optimization methods. Furthermore, as explained previously, this benchmark is required to investigate and show experimentally the claim that GE can be used to learn the links between latent spaces. The same G and weights from the GE generator is used in this benchmark. This allows for understanding the benefits of adding E as a filtering mechanism in GE. This model is denoted as invertGAN.
c. *AEGAN.* This benchmark is used to visualize the effects of combining the loss function of the AE and GAN in a typical AEGAN framework. The standard AEGAN objective given by

$$\min_{\theta_D,\theta_G,\theta_E} d_1(x, G(E(x))) + \mathbb{E}_{x\sim p_{data}(x)}[\log D(x;\theta_D)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z;\theta_G);\theta_D))] \tag{6}$$

is used. For the experiments, the same G and D structure from pGAN is used, while E adopts the same architecture (Table 1), with h in the final layer replaced to the same as the input vector size of pGAN, i.e. $h = 512$. The need to replace this final layer already highlights one of the advantage of GE, in which here, the same feature space is required, while in GE, this is not required. AEGAN is trained following (6), with similar loss functions to GE and the other baselines, without additional enhancements (e.g. Spectral Normalization). Thus d_1 is the

ℓ^2-norm. Additionally, a KL-divergence term [49] is included to ensure latent space follows normal distribution.

5. **Experimental Results.** This section presents empirical results for the experiments.

5.1. **Reconstruction Results.** In the reconstruction case, Equation (3) is solved with S as identity. The results are compared with ConvAE, AEGAN and invertGAN, which is presented in Figure 2 for the CelebA dataset, Figure 3 for the Digital Rock dataset and Figure 4 for the LSUN church dataset.

FIGURE 2. Reconstruction results on CelebA dataset.

5.2. **Denoising, Deblurring, Super-resolution and Inpainting Results.** For the other applications, Equation (3) is solved, and the AE is trained with the added loss term as described in the previous section, using their corresponding noise to generate noisy data. The results for denoising, deblurring, super-resolution, inpainting are presented in Figures 5, 6, 7 and 8 respectively.

5.3. **Quantitative Analysis.** Numerically, the following metrics: average mean squared error (MSE), structural similarity index measure (SSIM) [50], and Frchet inception distance (FID) [51] are used to evaluate the performance of GE in reconstructing images in the CelebA test set. MSE measures the overall accuracy of reconstruction, while SSIM and FID serves as a good metric for the evaluation of detail. SSIM is evaluated using the python package skimage, while FID score is computed using provided code from [51]. For the digital rock dataset, MSE and Peak Signal-to-Noise Ratio (PSNR) are used, as they are commonly used metrics on digital rock imaging. The results are presented in Table 2 and 3.

Figure 9 plots the log of average Mean Squared Error (MSE) over the test data during each iteration in the solving phase of GE. As comparison, the average MSE for ConvAE and AEGAN are also included. In general, convergence in invertGAN and GE is stabilized around 1000 iterations.

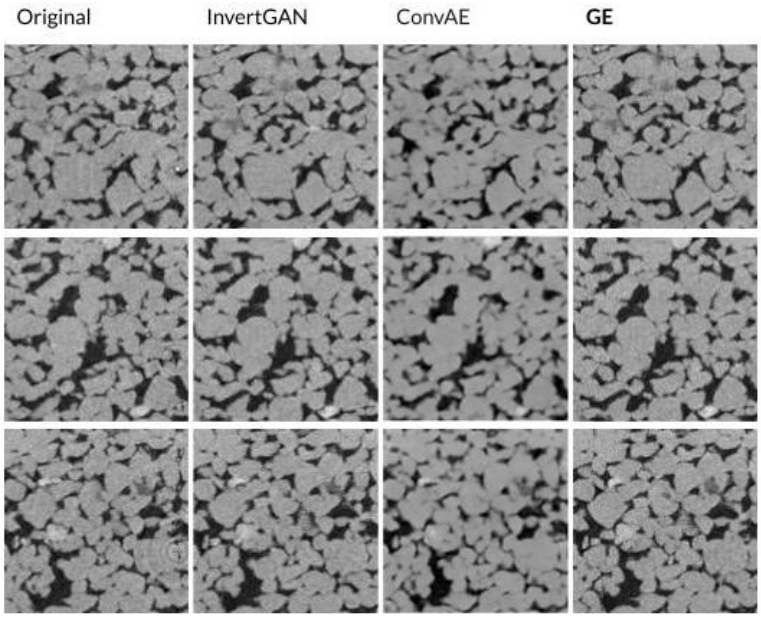

FIGURE 3. Reconstruction results on Digital Rock dataset.

FIGURE 4. Reconstruction results on LSUN church dataset.

Comparing GE against ConvAE, invertGAN, and AEGAN, GE outperforms all 3 baselines based on the metrics. Visually, Figures 2 and 3 demonstrates this phenomenon. The use of a generator to generate output images in the GE model provides image reconstructions of higher quality than the coarse images output by both the ConvAE and AEGAN in both figures. Both models, which is trained using ℓ^2-norm loss, while being able to remain competitive in overall accuracy scores in

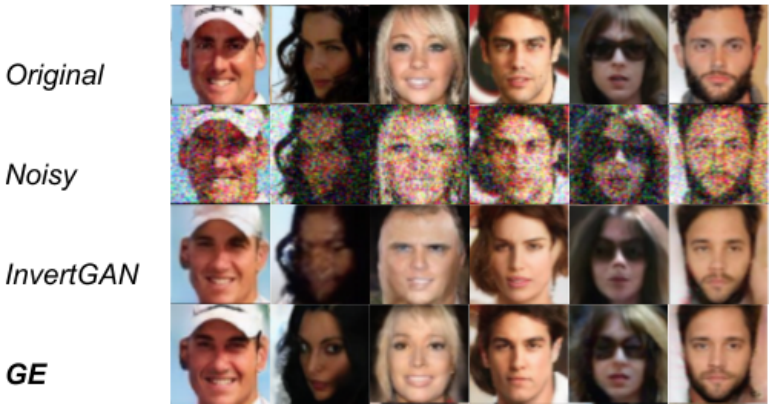

FIGURE 5. Denoising results on CelebA dataset.

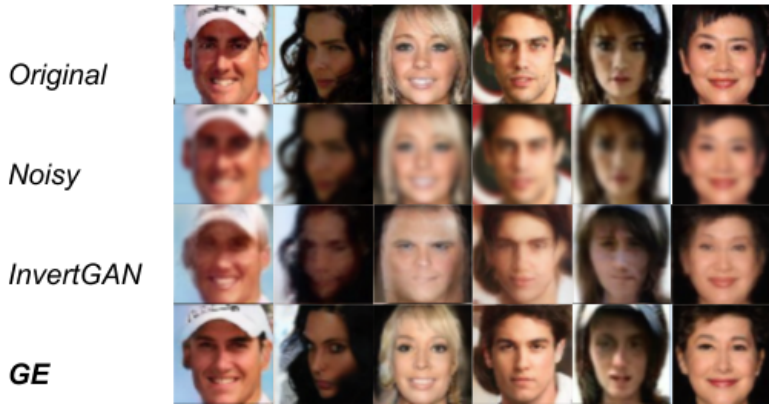

FIGURE 6. Deblurring results on CelebA dataset.

FIGURE 7. Super-resolution results on CelebA dataset.

FIGURE 8. Inpainting results on CelebA dataset.

Model	MSE	SSIM	FID
CRGAN*			16.97
SSGAN*			24.36
Our pGAN			22.13
ConvAE	0.03386	0.6823±0.051	87.71
AEGAN	0.03317	0.6907±0.050	34.53
invertGAN	0.03529	0.7203±0.038	19.19
GE	**0.03262**	**0.7329±0.025**	**17.42**

TABLE 2. Quantitative results comparing models for CelebA. Additionally, some FID scores reported by recent GAN papers that used CelebA $128\times128\times3$ images are also presented for comparison, labelled with *.

Model	MSE	PSNR
ConvAE	0.009271	20.32
invertGAN (512)	0.008185	20.86
GE (512)	**0.007470**	**21.26**
GE (256)	0.007741	21.11
GE (128)	0.007839	21.05
GE (64)	0.008499	20.70

TABLE 3. Quantitative results comparing models for digital rocks. The number in brackets show the size of the latent vector in pGAN that the model is trained on. Models with same latent sizes are solved with the same pGAN weights. The same AE is used for all models.

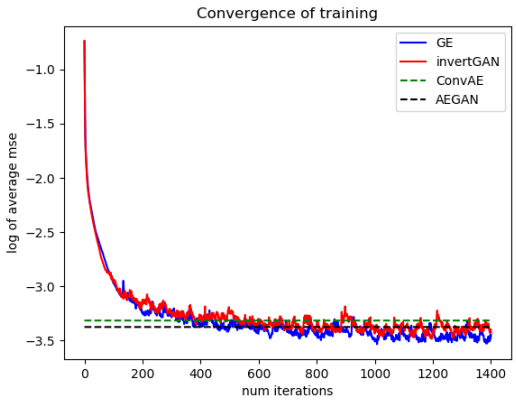

FIGURE 9. Plot of log of average MSE based on number of iterations in the solving phase.

MSE, loses out in details, where it can be seen that SSIM, PSNR and FID scores perform much worse than the GAN based counterparts (invertGAN and GE), whose generators are not trained with ℓ^2 norm, as seen in Tables 2 and 3. This highlights the importance in separating the GAN and AE training, where combining both introduces coarser outputs.

In comparison against recent GAN models like CRGAN [52], SSGAN [53], GE is still capable of presenting competitive performance in terms of FID scores. The flexibility of the GE framework allows us to easily adapt to newer models, by replacing the corresponding GAN or AE component and training in the same manner. This flexibility also shows itself in the digital rock experiments, where the same fixed AE is used across all variations of GE with differing parameters, and retraining of the models is only performed on the GAN component.

The differences between the scores for invertGAN and GE also highlights the importance of E in the solving phase. For GE, E plays two main roles. First, it plays a large role in reducing the search domain from the image domain to the compressed domain. Due to the optimization problem (3) being highly non-convex, this is crucial to ensure the stability and quality of the result. Numerically, the overall performance of GE outperforms invertGAN. SSIM scores are also much stabler in GE (lower SD values) as opposed to the other models. Visually, from examples like Sample 1 in Figure 2, the reduction of the search domain through E captures important features like hat patterns, which is missing in invertGAN. A closeup analysis of this phenomenon is presented in Figures 10 and 11. Additionally, the above phenomena is also observed in the LSUN church dataset, boxed in red in Figure 4. In the first image, the sign patterns is integrated into the background building in InvertGAN, while not in GE, while the second image shows the missing human, where in InvertGAN, this is integrated into the background building.

This is of crucial importance in digital rock images. The GE algorithm is applied to test images of 2D slices along an axis of a 3D digital rock sample from the dataset. The resulting 2D results are then combined to form a 3D reconstruction of the sample and a separate slice along another axis is obtained from the result. Figure 12 shows one such slice. In this application, the requirement of E in GE

FIGURE 10. Comparison of image reconstruction of detail region (red box) for original image (left). In order of comparison, from left to right, we have Original, GE, invertGAN, ConvAE.

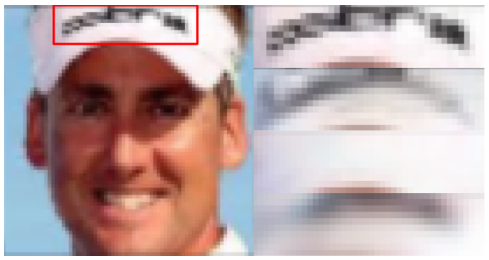

FIGURE 11. Comparison of image reconstruction of detail region (red box) for original image (left). In order of comparison, from top to bottom, we have Original, GE, invertGAN, ConvAE.

becomes apparent, where without them, the GAN only InvertGAN model is seen to introduce additional pores to the final results (highlighted by the red box), which is undesirable. The use of an encoder in GE highly stabilizes this, and reduces the likelihood to remove or add unnecessary details.

Secondly, E acts as a filter for noise. GE outperforms invertGAN in all of our applications, as seen in Figures 5, 6, 7 and 8. In particular, consider results in inpainting (Figure 8) and deblurring (Figure 6), in which InvertGAN fails totally. In comparison, GE performed well even when using the same G throughout the applications, while changing only the AEs which are trained to filter noise. A single G can be used to solve all the problems in GE, whilst other GAN methods may require training the entire model in some special manner to treat the noise, especially in complicated applications. In GE, only the much stabler to train AE portion needs retraining. GE presents an edge to other frameworks in this regard, as it is sufficient to train a strong GAN that focuses solely on detail generation while leaving filtering tasks to the AE.

5.4. **Efficiency Analysis.** The presence of a solving phase in GE results in a slower inference time during inference. As seen in Figure 9, performance of GE achieves the same amount of error as 1 step inference methods like ConvAE and AEGAN at about 400 iterations. This timing can be reduced using early stopping after the error reaches a predefined threshold, while the overall optimization time can be reduced by improving the initialization method for z in the solving phase. For example, instead of initializing z using one random start, multiple z can be sampled and the vector whose reconstruction $G(z)$ has the lowest loss to the target image x can be used for initialization. Alternatively, the initialization can be trained using

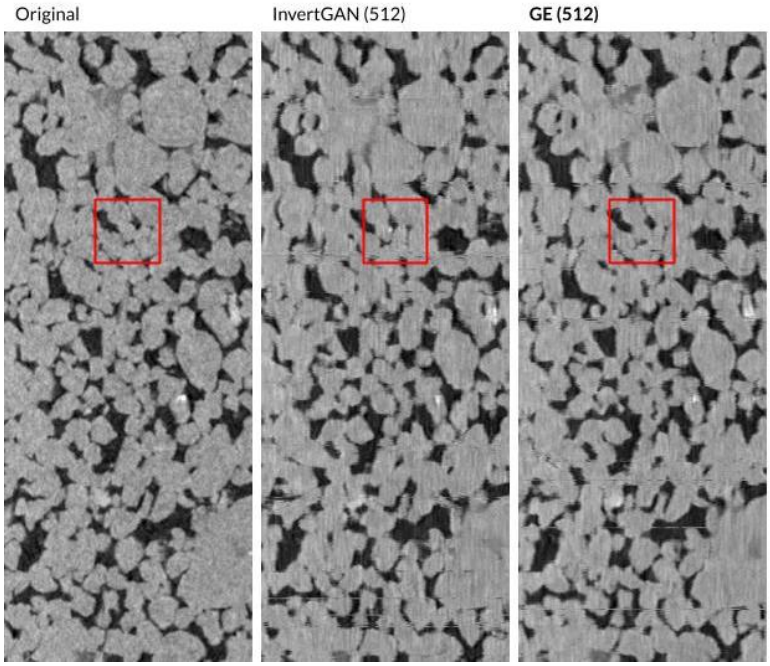

FIGURE 12. Additional pore sample result on Digital Rock dataset.

an encoder like structure which could even perform the solving phase in 1 step. This is left as future work.

5.5. **Visualizing feature spaces.** The differences in representations of both feature spaces are apparent in Figures 4, 10, 11 and 12. This motivates a need for the study of relationships between the feature spaces. This section demonstrates how the relationship between the two feature spaces can be investigated. For this, the CelebA dataset will be used and spectacle features are chosen as the feature of study. We investigate image reconstructions for 1000 randomly chosen eyeglass images in train data instead of test data, as the feature to investigate would be observed in the training of the model. Table 4 shows the truth table comparing both models, and Figure 13 shows some samples. Table 4 shows that 35% of samples which did not produce spectacles in invertGAN did so in GE. The last sample in Figure 13 showed a case where both model fails, and comparing with the original image, this is likely that the spectacles are difficult to detect.

	InvertGAN,F	**InvertGAN,T**
GE,F	289	32
GE,T	157	469

TABLE 4. Results of invertGAN, GE on spectacles. T refers to samples which produced spectacles, F refers to samples which did not. Remaining are invalid reconstructions.

The appearance of spectacle features in GE shows that they are not completely forgotten by G, and instead is likely to be just underrepresented in favor of other

FIGURE 13. Missing spectacles sample results on CelebA dataset.

features. Thus, introducing E magnified the observation of said feature in G. GE's optimization framework provides a method to encourage the same G to reproduce underrepresented features learned, without needing to skew training data. Figure 13 and Table 4 visualizes the relative importance of spectacles in the generative and encoding space. In the generative space, as the objective is to generate realistic images, G tends towards reproducing simple but detailed features, and less on generating complicated details. In the encoding space, however, as the objective is to reconstruct the image, such features like spectacles are important to remember. These observations coincide with the traditional understandings of each feature space.

GE presents an effective tool to investigate specific shortcomings in each component. In this example, a reasonable conclusion is that the introduction of E results in the underrepresented feature of spectacles in G to be enhanced, without any retraining of G. If one wishes to improve the overall result in reproducing said feature, from this visualization, a more effective approach would be to tackle improvements in GAN - for e.g. choosing alternative GAN which performs better in producing spectacles. In contrast to existing methods that use the same latent space, in order to improve their results, it is difficult to visualize which component requires improvement, and retraining has to be done on the whole model.

5.6. **Flexibility of GE.** In this final part, GE is retrained using the same ConvAE, but a different choice of GAN, BEGAN. The objective is to demonstrate the flexibility of GE, which can adapt to different choices of GAN. Figure 14 shows the results for GE using BEGAN. Note that since the final result is obtained using $G(z^*)$, the quality of the final reconstruction depends on the ability of G to generate details. This can also be observed in the constant quality of details in the results from applying GE to solve the different applications.

As GE solves the imaging problem during a solving phase instead of the training phase, modifying components of G or E with alternative models is still able to achieve good results. The final quality of result depends on the detail generating capability of G, and how E encodes the data and filters noise.

6. **Conclusion.** This paper introduced GE, a flexible framework that produces promising outcomes for generative imaging tasks. GE unifies GANs and AEs in an innovative manner to maximize the generative capacity of GANs and the compression ability of AEs to stabilize image reconstruction. GE also introduces a method to visualize latent space relationships, and in turn, understanding each latent space

FIGURE 14. Reconstruction results for $64 \times 64 \times 3$ images in CelebA with GE using BEGAN instead of pGAN.

in relation to the training data. This helps to answer important questions on the latent space of generative networks. In the future, we intend to explore other applications that rely on latent spaces, like semantic attribute manipulation [54, 55], as well as derive theoretical results similar to [17, 5], to further the understanding of latent spaces and generative networks. The code will be available on the author's github.

REFERENCES

[1] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, December 2010.

[2] Jo Schlemper, Jose Caballero, Joseph V. Hajnal, Anthony Price, and Daniel Rueckert. A deep cascade of convolutional neural networks for mr image reconstruction. In Marc Niethammer, Martin Styner, Stephen Aylward, Hongtu Zhu, Ipek Oguz, Pew-Thian Yap, and Dinggang Shen, editors, *Information Processing in Medical Imaging*, pages 647–658, Cham, 2017. Springer International Publishing.

[3] Sachit Menon, A. Damian, S. Hu, Nikhil Ravi, and C. Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2434–2442, 2020.

[4] David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

[5] A. Bora, A. Jalal, E. Price, and A. G. Dimakis. Compressed sensing using generative models. *ICML'17 Proceedings of the 34th International Conference on Machine Learning*, 70:537–546, Aug 2017.

[6] Raymond A. Yeh, Chen Chen, Teck-Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson, and Minh N. Do. Semantic image inpainting with deep generative models. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6882–6890, 2017.

[7] Q. Yan and W. Wang. DCGANs for image super-resolution, denoising and debluring. 2017.

[8] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8183–8192, June 2018.

[9] Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2014.

[11] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[12] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[13] Junbo Jake Zhao, Michaël Mathieu, and Yann LeCun. Energy-based generative adversarial networks. In *ICLR*, 2017.
[14] David Berthelot, Tom Schumm, and Luke Metz. BEGAN: boundary equilibrium generative adversarial networks. *CoRR*, abs/1703.10717, 2017.
[15] Christopher Bowles, Liang Jeff Chen, Ricardo Guerrero, Paul Bentley, Roger N. Gunn, Alexander Hammers, David Alexander Dickie, Maria del C. Valdés Hernández, Joanna M. Wardlaw, and Daniel Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks. *ArXiv*, abs/1810.10863, 2018.
[16] Sheng-Wei Huang, Che-Tsung Lin, Shu-Ping Chen, Yen-Yi Wu, Po-Hao Hsu, and Shang-Hong Lai. Auggan: Cross domain adaptation with gan-based data augmentation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 731–744, Cham, 2018. Springer International Publishing.
[17] Qi Lei, Ajil Jalal, Inderjit S. Dhillon, and Alexandros G. Dimakis. Inverting deep generative models, one layer at a time. *CoRR*, abs/1906.07437, 2019.
[18] David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. Seeing what a gan cannot generate. *ArXiv*, abs/1910.11626, 2019.
[19] Zachary C. Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. *CoRR*, abs/1702.04782, 2017.
[20] Antonia Creswell and Anil Anthony Bharath. Inverting the generator of A generative adversarial network. *CoRR*, abs/1611.05644, 2016.
[21] V. Shah and C. Hegde. Solving linear inverse problems using gan priors: An algorithm with provable guarantees. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4609–4613, 2018.
[22] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015.
[23] Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, and S. Mohamed. Variational approaches for auto-encoding generative adversarial networks. *ArXiv*, abs/1706.04987, 2017.
[24] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Adversarial generator-encoder networks. *CoRR*, abs/1704.02304, 2017.
[25] Anh Nguyen, Jason Yosinski, Yoshua Bengio, Alexey Dosovitskiy, and Jeff Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. *CoRR*, abs/1612.00005, 2016.
[26] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *CoRR*, abs/1605.09782, 2016.
[27] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville. Adversarially learned inference. *ArXiv*, abs/1606.00704, 2017.
[28] Pascal Vincent, H. Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML '08*, 2008.
[29] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.
[30] E. J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489509, Feb 2006.
[31] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289 – 1306, Apr 2006.
[32] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311 – 4322, Oct 2006.
[33] J. Zhang, D. Zhao, and W. Gao. Group-based sparse representation for image restoration. *IEEE Transactions on Image Processing*, 23(8):3336–3351, May 2014.
[34] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Bm3d image denoising with shape-adaptive principal component analysis. *Proc. Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS'09)*, Apr 2009.
[35] A. Buades, B. Coll, and J. . Morel. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65 vol. 2, June 2005.
[36] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[37] P. Getreuer. Total Variation Inpainting using Split Bregman. *Image Processing On Line*, 2:147–157, 2012.
[38] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.
[39] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. *arXiv preprint arXiv:1801.07892*, 2018.
[40] W. Dong, L. Zhang, G. Shi, and X. Wu. Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. *IEEE Transactions on Image Processing*, 20(7):1838–1857, July 2011.
[41] Li Xu and Jiaya Jia. Two-phase kernel estimation for robust motion deblurring. 2010.
[42] R. Yan and L. Shao. Blind image blur estimation via deep learning. *IEEE Trans Image Process*, 25(4):1910–21, Apr 2016.
[43] Victor Lempitsky Dmitry Ulyanov, Andrea Vedaldi. Deep image prior. 1711.10925, 2017.
[44] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. pages 105–114, 07 2017.
[45] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.
[46] Thomas Ramstad. Bentheimer micro-ct with waterflood. http://www.digitalrocksportal.org/projects/172, 2018.
[47] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.
[48] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.
[49] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.
[50] Zhou Wang, Alan Bovik, Hamid Sheikh, and Eero Simoncelli. Image quality assessment: From error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13:600 – 612, 05 2004.
[51] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017.
[52] H. Zhang, Zizhao Zhang, Augustus Odena, and H. Lee. Consistency regularization for generative adversarial networks. *ArXiv*, abs/1910.12027, 2020.
[53] Ting Chen, Xiaohua Zhai, M. Ritter, M. Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12146–12155, 2019.
[54] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. *CoRR*, abs/1907.10786, 2019.
[55] Xiaodan Liang, Hao Zhang, and Eric P. Xing. Generative semantic manipulation with contrasting GAN. *CoRR*, abs/1708.00315, 2017.

Received xxxx 20xx; revised xxxx 20xx.

E-mail address: e0011814@u.nus.edu
E-mail address: haizhao@purdue.edu