

Unsupervised Latent Regression through Information Maximization - Contrastive Regularized GAN

Vicky Sintunata, Siying Liu, Dinh Nguyen Van, Zhao Yong Lim,
Ryan Lee Zhikuan, Yue Wang, Jack Ho Jun Feng, Karianto Leman

*Institute for Infocomm Research (I²R), A*STAR, Singapore*

{Vicky_Sintunata, liusy1, nguyen_van_dinh, lim_zhao_yong, ryan_lee, wang_yue, jack_ho, karianto}@i2r.a-star.edu.sg

Abstract—Most of labelled image public datasets are discrete in nature, e.g. cat vs dog, human, car, etc. With the growing complexity of tasks, fine-grained label data is needed. Fine-grained labels are costly because there is a need for experts to label them. Generative Adversarial Network (GAN) has been gaining a lot of attentions due to its ability to not only generate realistic images but also to disentangle the attributes of the images. Unfortunately, GAN's disentanglement methods usually lack the ability to quantify such attributes. The objective of this work is to quantify the attributes of the target (object/image) based on the disentangled properties without supervision. In order to get the (disentangled) attributes, we leverage GAN with information maximization and contrastive regularizer. Regression is done by adding additional layer to the contrastive networks of the model. The regression quality of the proposed method is quantified by order quality measured using normalized Kendall's Tau. Furthermore, an application in denoising image is also presented.

Index Terms—Generative Adversarial Network, Regression, Disentanglement, Gaussian White Noise Denoising

I. INTRODUCTION

It is no surprise that training deep learning model requires massive amount of data. Through the years, we have seen amazing performances of deep models by utilizing these data. Starting to just recognizing hand-written digit of MNIST [1], to object detection [2], [3], to object segmentation [4] and many more advance tasks. Producing these data has it's own challenges, mainly with the labelling and collecting the target data. Moreover, with the increasing complexity of the tasks, finer-grained attribute annotations are required to feed into a deep model. These fine-grained and granular level of annotations most often require an advance level of expertise and the process can be costly.

In the past years, generative adversarial networks (GANs) have gained a lot of attentions particularly of its ability to generate realistic images. GAN consist of a generator and a discriminator model in min-max game. The main goal of GAN is to produce an output (image) as realistically as possible resembling the dataset it trained on. Usually using a gaussian noise as inputs, the generator is tasked to fool the discriminator to assess the generator's output as real. On the other hands, the discriminator is tasked to differentiate whether the inputs it is given are real or fake.

Another interesting part of GAN is its ability to disentangle the attributes of the images. Disentangling attributes of images is important factor here, since it can then be used for finer-grained detection or classification. The disentanglement property of GANs make it easier for user to control the output of the image with certain attributes and hence many of the applications of GAN is for image editing [5]–[7]. The ability to edit images will certainly make finergrained data generation easier. However, the manual labor involved in this task remains demanding. Note also that the conventional approach of image editing using GAN is to first convert the target image into a latent vectors, such that if it is given to the generator, it will generate the same image as the target image. In this case, the discriminator is practically discarded after the initial GAN training.

The objective of this work is to quantify the attributes of an image based on the disentangled properties without any supervision. We train a GAN with additional layer of neural network so it allows discriminator to jointly learn to distinguish real or fake images and to quantify specific attributes changes. Our approach is inspired by the capabilities of the work done in [8]. Particularly the abilities of the network to identify which latent factors the pair inputs (images) correspond to each other. Originally, it is used to help with more diverse generation of images through contrastive regularizer, but we notice that it has potential to be used to as a regressor. In order to quantify our methods, we devise experiments based on a ranking/ordering task. The idea is if we can re-order the inputs according to a specific ground truth attributes, then we can assess how well our model can measure the value of that attribute correctly. During inference, both the generator and discriminator are used. The generator is used to generate image with "base" attributes value that will be compared with the target image's attributes value. To validate our findings, a normalized kendall's tau distance is used on the ordering results. From the experiments done, we show that the proposed method can achieve almost perfect score for the ordering task in each of the attributes of interest. Furthermore, an application of the proposed method in denoising is also presented.

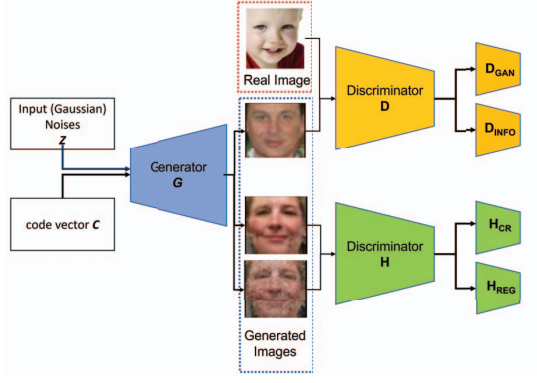


Fig. 1. Proposed Architecture. Building upon InfoGAN-CR [8], we add additional regression layer that quantify the similarity (or differences) of the input images attributes

II. RELATED WORK

Some previous works have also utilized the potential of discriminator as a classifier [9], [10]. In Auxiliary Classifier GAN (AC-GAN [9]), the discriminator is tasked to assess the distribution of the input (whether it comes from real image or generated/fake image) and to give label associated with it. The generator then take inputs a random noise vector along with the class label. ADC-GAN [10] improves the capabilities of AC-GAN especially in intra-class diversity of the generation by adding a discriminative classifier. Note that both of these methods require labelled datasets while our approach is done in unsupervised manner.

GAN's latent space disentanglement has been garnering interests for the past years. Some of the past research specifically try to disentangle a specified attribute of the inputs, e.g. pose and 3D representations [11], face representation [12]. Others have more general settings [8], [13], [14]. There are two approaches in disentangling the attributes in GAN. The first approach is through a manipulation in the latent space directly, i.e., in the layers of the GAN [5], [15], [16]. The second approach is to add control vectors to the input generator, i.e., the inputs to generator is random noise plus some additional vectors (usually continuous to represent continuous latent attributes) [8], [13], [17]. In [14], the authors proposed a regularization loss such that the generative model has a diagonal hessian matrix with respect to its input. By having a more diagonal hessian matrix, the latent space are expected to have more disentanglement. InfoGAN [8], [13] on the other hand, promotes disentanglement by maximizing mutual information between the latent variables. We will review a bit more on InfoGAN and subsequently InfoGAN CR in section III.

At the core, [18] is probably the most related to our work. In [18], the authors proposed of using a trained model from StyleGAN2 [15] combined with an inversion model from e4e [19]. The inversion model is used to convert the inputs (image space) to the latent space of StyleGAN and then the

resulting latent vectors are compared between the images in order to get a linear regressor. Different with their approach, our method is end-to-end. While in [18] some of the results are measured subjectively, in our works we propose a more objective quantification and results.

III. METHOD

A. InfoGAN and InfoGAN-CR

We will first briefly review InfoGAN [13] and InfoGAN-CR [8]. As like many other GANs, InfoGAN tries to optimize the competing models between the generator G and the discriminator D through the adversarial loss which is often using a binary cross entropy loss. x and x' represents the real image and generated image respectively.

$$\min_G \max_D \mathcal{L}_{adv}(D, G) = \min_G \max_D \mathbb{E}[\log(D(x)) + \mathbb{E}[\log(1 - D(x'))]] \quad (1)$$

On top of this, InfoGAN tries to achieve disentanglement by incorporating the information maximization through a mutual information regularizer I .

$$\min_G \max_D \mathcal{L}_{adv}(D, G) - \lambda I(c; G(c, z)) \quad (2)$$

$c \in \mathbb{R}^k$ is the disentangled code vector and the sample $G(c, z)$ where $z \in \mathbb{R}^d$ is the latent code usually sampled from Gaussian distribution. The authors in [8] argue that the factorized Gaussian with identity covariance as used in InfoGAN is necessary to keep the loss bounded, but it creates an implicit bias and therefore, reduce the disentanglement capability.

In order to reduce the implicit bias effect on the disentanglement, the authors in [8] then proposed to add contrastive regularizer based on the cross entropy, denoted as $\langle \cdot, \cdot \rangle$.

$$\mathcal{L}_c(G, H) = \mathbb{E}_{R \sim U([k]), (x, x') \sim Q^{(R)}} [\langle R, \log H(x, x') \rangle] \quad (3)$$

$Q^{(R)}$ and R represent the joint distribution of the paired images and random index one-hot encoding. The basic idea is that disentanglement is measured through the changes in latent space traversal. Measuring the changes requires the discriminator to make decision based on multiple samples with differing latent codes. To this end, additional discriminator H is created which sole purpose is to determine which latent code c the coupled input (x, x') is (are) sharing. Note that while discriminator D takes input real and (artificial) images generated by the generator G , discriminator H only takes input from the generator. In other words, given two or several images generated by generator G , discriminator H is tasked to determine which disentangled code vector c is responsible for the changes in the input (images).

B. GAN as Regression

Building on the success of InfoGAN-CR, we propose an additional layer that utilizes the capability of the discriminator H to not only distinguish which latent code c is similar (or different), but also to quantify how much the similarity (or differences) are. In order to leverage discriminator H 's

capability, we make a branching point to an encoder from discriminator H as shown in Fig. 1. The newly added encoder works as the regressor with a $(k \times N)$ -dimensional output vector where N is the batch size.

Recall that $x = G(z, c)$ is the output of the generator with z and c as inputs. Similarly, $x' = G(z, c')$ is the output of the generator with the same z vector and (different) c' latent code vector. There are two alternatives on how to train the discriminator H based on the difference between c and c' . The first one, which is also used in [8], is to share one (randomly picked) latent code c_j and then task the discriminator to correctly identify j . The second one is to fix all of the other latent codes and randomly pick one latent code (c_j) to be different and similarly, tasking discriminator H to correctly identify j .

The objective in this work is to measure the difference of the latent code and therefore, the latter option is used. More precisely, let c be sampled from gaussian distribution and c' is the copy of c , then we randomly select j -th latent code such that $c'_j = 0$. By opting for this approach, it will make training the discriminator easier. Then, the regressor can be trained with the Mean Squared Error as its loss (4). Note that c_j^i denotes j -th latent code of i -th batch (similarly for the c' part). Combining (4) with (2), the total loss during training can then be written as in (5).

$$\mathcal{L}_{reg} = \frac{1}{N \times k} \sum_{i=0}^N \sum_{j=0}^k (c_j^i - c_j'^i)^2 \quad (4)$$

$$\begin{aligned} \min_{G, H_{CR}, H_{reg}} \max_D \mathcal{L}_{adv}(D, G) \\ - \lambda_{Info} I(c; G(c, z)) \\ - \lambda_{CR} \mathcal{L}_c(G, H_{CR}) \\ - \lambda_{reg} \mathcal{L}_{reg}(G, H_{reg}) \end{aligned} \quad (5)$$

IV. EXPERIMENT

In all of our experiments, we are utilizing two trained models. The first one is the trained generator G to generate the base image, i.e., the image generated with all of its latent codes $\forall c_j = 0$, for comparison. The second one, the discriminator H which will take the base image and the target image from the dataset as inputs.

In order to quantify the result of our method, an ordering task is utilized. Given several images sampled (and shuffled) from the same domain, a good regressor will be able to re-order the said images according to its ground-truth features. For example, the target feature is people with long hair the regressor should be able to re-order the collection of images from people with short hair gradually to long hair. Unfortunately, such datasets with its ground truth values, to the best of our knowledge doesn't exist. So, several approaches like in [18] used a survey-based method to quantify it's proposal, which is tend to be subjective and prone to bias. Here, we are using a more reliable method to quantify our approach although most of them are synthetically generated images.

We run the experiments in several datasets: dSprites [20] and chair [21]. All of the experiments are using the same architecture described in Table I, trained in Pytorch. Unless otherwise specified, in general the input latent code to the generator is set to $z = 256$. Adam optimizer is used for all the models with learning rate for $G = 0.001$, $D = 0.0002$, and $H = 0.0002$. All using the beta parameter set up to (0.5, 0.999). As a warming up, the first epoch of the training step exclude discriminator H .

A. Ordering Measurement

Let $\sigma_1(i)$ and $\sigma_2(i)$ be the rankings of i -th element in the list σ_1 and σ_2 and let's assume that σ_1 is the ground truth ordering of the data, obtained from the labeled data. For example in the chair dataset, the ground truth ordering is based on the value of the yaws and pitch. If there are three chairs (A, B, C) with yaws values 30, 60, 45 respectively, then the order of the ground truth is (A, C, B). In practice, it is dependent on whether ascending or descending order is preferred. The quality of the ordering results are assessed using the normalized Kendall's Tau distance [30] defined in (6), where d is the Kendall's tau distance which measures the total number of discordant pairs in the lists.

$$K(\sigma_1, \sigma_2) = \frac{2|d|}{n * (n - 1)} \quad (6)$$

$$\begin{aligned} d = \{ (i, j) \in S \times S | i < j \wedge (\\ \sigma_1(i) < \sigma_1(j) \wedge \\ \sigma_2(i) > \sigma_2(j) \vee \\ \sigma_1(i) > \sigma_1(j) \wedge \sigma_2(i) < \sigma_2(j)) \} \end{aligned} \quad (7)$$

$$d = \begin{cases} d & \text{for } d \geq 0.5 \\ 1 - d & \text{for } d < 0.5 \end{cases} \quad (8)$$

The normalized Kendall's Tau distance has value ranging from 0 to 1, where 0 means a perfect match (of ordering) between the list and 1 is the total opposite ordering of the lists. Note that in our case, 1 can be regarded to have the same value as 0, i.e., a good value to have, because this means the discriminator managed to distinguish and order the level of the latent codes perfectly, only in the opposite direction. In summary, the ideal value of the normalized Kendall's Tau distance is approaching either 0 or 1. In general, all values reported are normalized to 1 as the best result, i.e., if a normalized Kendall's Tau distance value is less than 0.5, then it will be calculated using Equation 8, unless otherwise stated.

B. dSprites Dataset

The dSprites Dataset consists of 737,280 images of generated sprites with 6 independent latent factors, i.e., color, shape, scale, orientation, x and y position. Although the color itself is set up to only one color (white), the rest of the latent factors has varying values. There are 3 shapes (square, ellipse, heart), 6 scales (linearly spaced in [0.5, 1]), 40 orientations ranging from $[0, 2\pi]$, 32 x positions in $[0, 1]$, and 32 y positions in $[0, 1]$.

TABLE I
DETAILED OF THE ARCHITECTURE USED FOR EXPERIMENTS. FOR THE CONVOLUTION PART SHOWS THE KERNEL SIZE ($n \times n$) FOLLOWED BY (OUTPUT SIZE, NORMALIZATION METHOD, ACTIVATION). BN: BATCH NORMALIZATION, SN: SPECTRAL NORMALIZATION, I: IDENTITY FUNCTION

Generator G	Discriminator D	Discriminator H
Input ($c, 256$) FC (512, ReLU) FC ($4 \times 4 \times 64$, ReLU) 4×4 UpConv.1 (256, BN, lReLU) 4×4 UpConv.2 (128, BN, lReLU) 4×4 UpConv.3 (64, BN, lReLU) 4×4 UpConv.4 (3, tanh)	Input (64,64,3) 4×4 Conv.1 (64, SN, lReLU) 4×4 Conv.2 (128, SN, lReLU) 4×4 Conv.3 (256, SN, lReLU) 4×4 Conv.4 (256, SN, lReLU) FC (512, SN, lReLU) Branch 1: D_{GAN} FC(1, sigmoid) Branch 2: D_{Info} FC($2 \times c$, SN, [sigmoid, I])	Input(64,64,6) 4×4 Conv.1 (64, lReLU) 4×4 Conv.2 (128, BN, lReLU) 4×4 Conv.3 (256, BN, lReLU) 4×4 Conv.4 (256, BN, lReLU) FC(512) Branch 1: H_{CR} FC(c, I) Branch 2: H_{REG} FC(c , tanh)

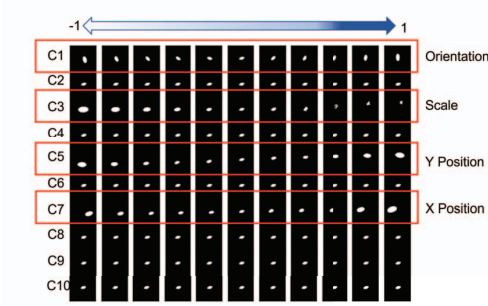


Fig. 2. Sample of the generated images learned from dSprites dataset with varying each latent code c value from $[-1, 1]$

During training, the parameters are set as follow: 100 epochs, batch size = 64, $c = 10$, $\lambda_{CR} = 5$, $\lambda_{Info} = 0.2$, and $\lambda_{reg} = 20$. Fig. 2 shows the generated images learned from the dataset. Highlighted on with the red boxes are the latent of interest that will be used for the quantitative experiment. Orientation, scale, y position and x position will be measured with latent code c_1 , c_3 , c_5 , and c_7 respectively.

For the quantitative experiment, we tested our model with each latent factors (excluding the color latent factor) for each shapes separately. Note that for most of the experiments, first we will randomly select and fix the parameters of the attributes. For example, in the scale experiment, we will randomly select one parameter for each of other attributes (orientation, x-y positions) and fix them, while the scale parameter will be used in full, i.e., all 6 scales. The exception is for the orientation experiment where only the first 10 orientations (0 to $\pi/2$) will be used for simplicity. All experiments are repeated for 1000 times and the results can be seen in Table II. The summary based on the observation from Fig. 2 is presented in Table III.

C. Chair Dataset

The 3D Chair dataset consists of 86,366 images from 1,393 types of rendered 3D chairs with 62 different viewing orientation (31 yaws (Θ): from 0 to 348 degrees, 2 pitch (Φ) values: 20 and 30 degrees). with the total of 86,366 rendered



Fig. 3. Sample of generated image of learned 3D chair data with varying each latent code c value ranging from $[-1, 1]$. Latent code c_1 will be used for quantitative experiment.
































































































               	GT
               	0.0095
               	0.1238
               	0.5523
               	0.2666
               	0.8285

TABLE II
AVERAGE NORMALIZED KENDALL’S TAU DISTANCE VALUES FOR EACH LATENT CODE c . BOLD FONT INDICATES THE OBSERVED LATENT CODE BASED ON FIG.2, UNDERLINE INDICATES THE BEST VALUE

Type	Ave.Normal Kendall’s Tau									
	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}
Orientation (Box)	<u>0.692</u>	0.639	0.684	0.644	0.651	0.656	0.661	0.657	0.662	0.688
Orientation (Ellipse)	0.994	0.892	0.8636	0.75	0.903	0.754	0.733	0.877	0.724	0.724
Orientation (Heart)	0.743	0.75	<u>0.816</u>	0.752	0.811	0.75	0.73	0.788	0.756	0.715
Position X	0.777	0.716	0.614	0.702	0.839	0.657	<u>0.992</u>	0.715	0.635	0.653
Position Y	0.788	0.746	0.571	0.714	<u>0.937</u>	0.717	0.815	0.698	0.689	0.644
Scale	0.89	0.838	<u>0.978</u>	0.821	0.969	0.808	0.957	0.875	0.86	0.823

TABLE III
AVERAGE NORMALIZED KENDALL’S TAU VALUE FOR DSPRITES EXPERIMENTS BASED ON THE OBSERVATION IN FIG. 2

Type	Ave.Normal Kendall’s Tau
Orientation (Box)	0.6924
Orientation (Ellipse)	0.9944
Orientation (Heart)	0.743
Position X	0.992
Position Y	0.937
Scale	0.978
Mean of Average	0.889

TABLE IV
AVERAGE NORMALIZED KENDALL’S TAU VALUE FOR 3D CHAIR EXPERIMENTS BASED ON THE OBSERVATION IN FIG. 3

Range	Ave.Normal Kendall’s Tau				
	c_1	c_2	c_3	c_4	c_5
$\Theta : 0 - 180; \Phi : 20$	0.9006	0.6648	0.5549	0.6445	0.7167
$\Theta : 180 - 360; \Phi : 20$	0.9107	0.2509	0.5416	0.6135	0.6773
$\Theta : 0 - 180; \Phi : 30$	0.9103	0.625	0.4653	0.6261	0.3565
$\Theta : 180 - 360; \Phi : 30$	0.9385	0.7786	0.5344	0.6183	0.7

D. Gaussian White Noise Image Denoising

Here, we are applying the proposed method to solve denoising problem, particularly in gaussian white noise denoising. There are several datasets used for training i.e., DIV2K [26], Flickr2K, BSD500 [27], and WED [28]. We adopt the implementation of BM3D [29] for the denoising algorithm. BM3D requires sigma (level of white noise) value as input, which can be estimated from the proposed method, particularly in the output of H . In order to map from the discriminator H output to the actual sigma values, we train a simple linear regression with 500 labelled data ($\sigma \in \{15, 25, 50\}$) from the training dataset. The test set is taken from Set12 [24], (C)BSD [23], and Kodak [25] datasets.

There are some adjustment made during the training phase. Firstly the parameters are set as follow: 300 epochs, batch size = 64, $z = 512$, $c = 5$, $\lambda_{CR} = 0.5$, $\lambda_{Info} = 0.2$, and $\lambda_{reg} = 10$. Secondly, instead of generating the corrupted images (noise free image + gaussian white noise with certain level of noise) directly, the generator is tasked to generate only the noise component. Hence, the last adjustment is the input to discriminator H . Instead of only sending the generator’s output, we combine it with real (noise-free) images. Fig.5 visualize the adjustment made.

The result of the denoising algorithm is quantified by the

TABLE V
QUANTITATIVE COMPARISON IN TERMS OF PSNR (DB) OF THE DENOISED IMAGES

Noise Level	Dataset	Ours	N2Score (Known)	N2Score(Unknown)
$\sigma = 25$	Set12	32.267	30.13	30.08
	Kodak	31.234	31.89	31.78
	CBSD68	30.208	30.85	30.78
	BSD68	30.666	29.12	28.95
			27.16	26.65
$\sigma = 50$	Set12	28.236	27.15	28.13
	Kodak	27.175	28.83	28.13
	CBSD68	25.973	27.75	27.32
	BSD68	26.383	26.21	25.81

PSNR value which can be seen in Table V. We compare the results with another method proposed in N2Score [22]. There are two values taken from [22] namely the known and unknown parameters (i.e., known sigma values). Note that for fair comparison, we compare the results here only with the unknown parameters value, while the known parameters value is shown only for completeness sake.

V. DISCUSSION & LIMITATIONS

We have presented our work on unsupervised GAN regression by building upon information maximization and contrastive regularizer GAN model. We are also proposing a way of a more objective assessment of attribute regression through the usage of normalized Kendall’s Tau distance value. As seen by the results, the proposed method can achieve high ordering accuracy as measured by the normalized Kendall’s Tau distance. Interesting to note in the result shown in Table II, that the model learn particularly well for the attributes like position and scale and to some extent the orientation (ellipse). Unfortunately, the orientation regressor doesn’t seem to be quite accurate for other ”entities” like the heart and box shaped object. Although, the result in Table II suggests that the latent code is still entangled. For example, the average score for the heart orientation has highest value in latent code c_3 which is more prominent in the scale attribute. Interestingly, apart from the other ”reserved” latent codes, the highest score for heart orientation lies in c_8 , although looking back at the visualization in Fig. 2 shows that there is no perceivable differences when manually changing the latent code values. Another possible reason is precisely that it is a different entity, hence the orientation base or origin is different. This is supported by the result in the 3D chair dataset where the model can learn orientation well.

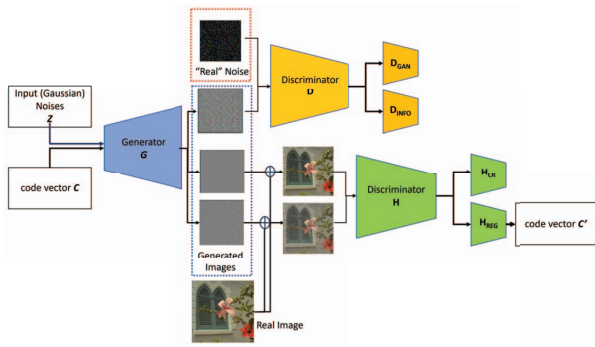


Fig. 5. Adjustment to the training for denoising experiment.

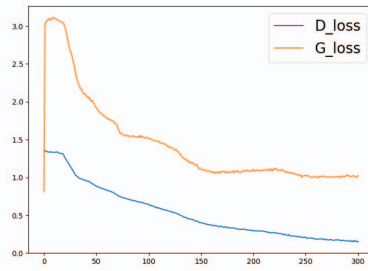


Fig. 6. Losses plot during denoising training. Note that the first jump in the generator loss is due to the warming up of the generator

We have also presented one of possible application of the proposed method through image denoising experiment. Fig. 6 shows the losses plot during training which shows the stability of the proposed method. Also, it performs comparably well with deep learning based method [22], especially in lower noise level. While here we only presented one noise type, we are looking to incorporate multiple noise types and more potential uses of the proposed method for our future work.

REFERENCES

- [1] Y. LeCun, C. Cortes, and C. Burges, "Mnist hand-written digit database," ATT Labs [Online]. Available: <http://yann.lecun.com/exdb/mnist>, vol.2, 2010.
- [2] A. Krizhevsky, "Learning multiple layers of features from tiny images," tech. rep., 2009.
- [3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp.248-255, 2009.
- [4] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," CoRR, vol. abs/1405.0312, 2014.
- [5] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in CVPR, 2020.
- [6] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," in CVPR, 2021.
- [7] H. Ling, K. Kreis, D. Li, S. W. Kim, A. Torralba, and S. Fidler, "Editgan: High-precision semantic image editing," in Advances in Neural Information Processing Systems (NeurIPS), 2021.
- [8] Z. Lin, K. K. Thekumparampil, G. Fanti, and S. Oh, "Infogan-cr and model centrality: Self-supervised model training and selection for disentangling gans," in ICML'20: Proceedings of the 27th International Conference on Machine Learning, pp.6127-6139, 2020.
- [9] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in International Conference on Machine Learning, 2016.
- [10] L. Hou, Q. Cao, H. Shen, S. Pan, X. Li, and X. Cheng, "Conditional GANs with auxiliary discriminative classifier," in Proceedings of the 39th International Conference on Machine Learning, pp.8888-8902, PMLR, 2022.
- [11] T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, "Hologan: Unsupervised learning of 3d representations from natural images," in The IEEE International Conference on Computer Vision (ICCV), 2019.
- [12] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning gan for pose-invariant face recognition," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1283-1292, 2017.
- [13] X. Chen, Y. Duan, I. S. Rein Houthoofd, John Schulman, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in Adv Advances in Neural Information Processing Systems, pp. 2172-2180, 2016.
- [14] W. Peebles, J. Peebles, J.-Y. Zhu, A. A. Efros, and A. Torralba, "The hessian penalty: A weak prior for unsupervised disentanglement," in Proceedings of European Conference on Computer Vision (ECCV), 2020.
- [15] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 43, pp. 4217-4228, 2021.
- [16] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in Proc.CVPR, 2020.
- [17] B. Liu, Y. Zhu, Z. Fu, G. de Melo, and A. Elgammal, "Oogan: Disentangling gan with one-hot sampling and orthogonal regularization," ArXiv, vol. abs/1905.10836, 2019.
- [18] Y. Nitzan, R. Gal, O. Brenner, and D. Cohen-Or, "Large: Latent-based regression through gan semantics," in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 19217-19227, 2022.
- [19] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for stylegan image manipulation," ACM Transactions on Graphics (TOG), vol. 40, pp. 1 - 14, 2021.
- [20] L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner, "dsprites: Disentanglement testing sprites dataset." <https://github.com/deepmind/dsprites-dataset/>, 2017.
- [21] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic, "Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models," in CVPR, 2014.
- [22] K. Kim and J. C. Ye, "Noise2Score: Tweedie's approach to self-supervised image denoising without clean images," in NeurIPS, 2021.
- [23] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in ICCV, 2001.
- [24] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," TIP, 2017.
- [25] R. Franzen, "Kodak lossless true color image suite. <http://r0k.us/graphics/kodak/>, 1999.
- [26] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in CVPR Workshops, 2017.
- [27] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," IEEE Trans. Pattern Analysis & Machine Intelligence, 2011.
- [28] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang, "Waterloo exploration database: New challenges for image quality assessment models," TIP, 2016.
- [29] Y. Mäkinen, L. Azzari and A. Foi, "Collaborative Filtering of Correlated Noise: Exact Transform-Domain Variance for Improved Shrinkage and Patch Matching," in IEEE Trans. Image Processing, vol.29, pp.8339-8354, 2020.
- [30] V. A. Cicirello, "Kendall tau sequence distance: Extending kendall tau from ranks to sequences," EAI Endorsed Transactions on Industrial Networks and Intelligent Systems, vol. 7, no. 23, 2020.