Generation of data on discontinuous manifolds via continuous stochastic non-invertible networks

Mariia Drozdova[1,2]*, **Vitaliy Kinakh**[1], **Guillaume Quétant**[1,2],
Tobias Golling[2] **& Slava Voloshynovskiy**[1]
[1]Department of Computer Science
[2]Department of Particle Physics
University of Geneva
Switzerland
{mariia.drozdova,svolos}@unige.ch

Abstract

The generation of discontinuous distributions is a difficult task for most known frameworks such as generative autoencoders and generative adversarial networks. Generative non-invertible models are unable to accurately generate such distributions, require long training and often are subject to mode collapse. Variational autoencoders (VAEs), which are based on the idea of keeping the latent space to be Gaussian for the sake of a simple sampling, allow an accurate reconstruction, while they experience significant limitations at generation task. In this work, instead of trying to keep the latent space to be Gaussian, we use a pre-trained contrastive encoder to obtain a clustered latent space. Then, for each cluster, representing a unimodal submanifold, we train a dedicated low complexity network to generate this submanifold from the Gaussian distribution. The proposed framework is based on the information-theoretic formulation of mutual information maximization between the input data and latent space representation. We derive a link between the cost functions and the information-theoretic formulation. We apply our approach to synthetic 2D distributions to demonstrate both reconstruction and generation of discontinuous distributions using continuous stochastic networks.

1 Introduction

The generation of data with discontinuous distributions with non-invertible networks represents a great interest for many problems in high energy physics, astrophysics and chemistry all dealing with high dimensional data. The previous attempts to develop generative models for discontinuous distributions show limited performance of GANs [1][2][3] and VAE models [4]. Flow models [5] can handle this problem to some extent but they face the complexity issues when the dimensionality of data increases. Hybrid models such as SurVAE [6] try to solve this problem by a combination of non-invertible and invertible networks based on Flows.

In this paper, we present a new information-theoretic stochastic contrastive generative adversarial network SC-GAN. The SC-GAN is a hybrid system that is based on a deterministic encoder producing an interpretable latent space and a stochastic decoder representing a generator. The generator architecture is a set of fully connected layers implemented based on a stochastic EigenGAN network [7] conditioned on a set of random noise vectors at each layer. The model is trained both in the *reconstruction* mode (with fixed noise vectors) and in the *generative* mode. The contrastive encoder is trained independently of the generator. The latent space of the encoder is then clustered using K-Means and approximated by many low-complexity mapping networks which try to shape Gaussians

---

*M. Drozdova and S. Voloshynovskiy are corresponding authors.

into corresponding cluster distributions. Finally, the decoder is trained jointly for reconstruction and generation using the likelihood with the corresponding discriminators.

We provide an information-theoretical interpretation of the proposed model in Section 2. In Section 3, we perform the analysis of both auto-encoding mode and generation of toy 2D datasets: Eight Gaussians, Checkerboard, Two spirals, Abs, Sinewaved cube and Four circles.

2 Information-theoretic formulation

The proposed framework is schematically shown in Figure 1 and consists of three stages of training.

2.1 The training of the encoder (stage 1)

The encoder is trained to maximize the mutual information between the data $\mathbf{X}$ and its latent space representation $\mathbf{E}$:

$$\hat{\phi}_\varepsilon = \underset{\phi_\varepsilon}{\operatorname{argmax}}\, I_{\phi_\varepsilon}(\mathbf{X}; \mathbf{E}), \tag{1}$$

where $I_{\phi_\varepsilon}(\mathbf{X}; \mathbf{E}) = \mathbb{E}_{p(\mathbf{x},\varepsilon)}\left[\log \frac{q_{\phi_\varepsilon}(\varepsilon|\mathbf{x})}{q_{\phi_\varepsilon}(\varepsilon)}\right]$.

The encoder is trained independently from the decoder using contrastive losses (Figure 1a). The maximization can be considered in the scope of the InfoNCE framework [8] and technically implemented using for example SimCLR contrastive learning [9]. For our toy datasets we choose simple augmentations based on the addition of small noise to the input data.

(a) Stage 1

(b) Stage 2

(c) Stage 3 : generation step
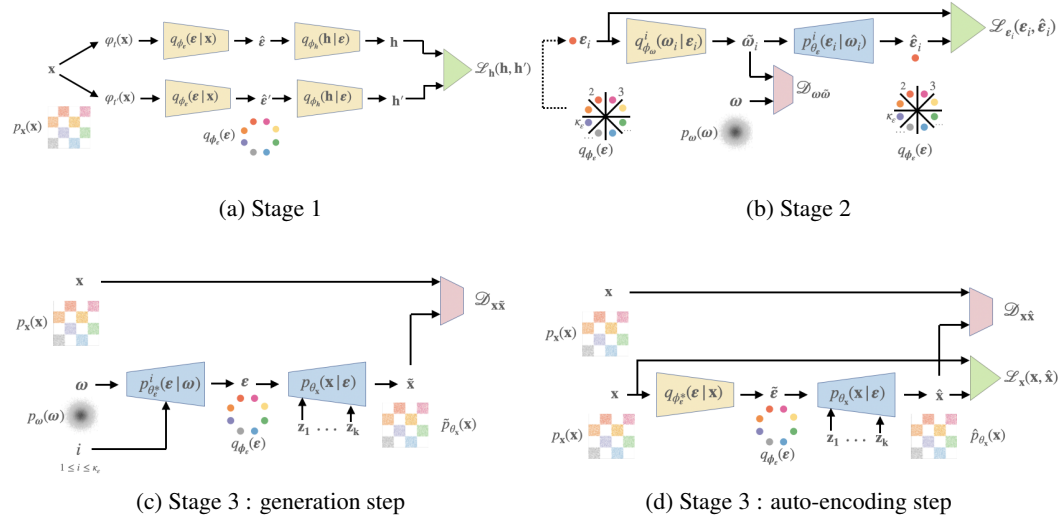
(d) Stage 3 : auto-encoding step

Figure 1: The proposed framework: Stage 1 - training of the encoder, Stage 2- training of the mapping network, Stage 3 - training of the decoder for the simultaneous reconstruction and generation.

2.2 The training of the latent space mapping networks (stage 2)

The mapping networks aim at generating the complex latent space ε from a simple distribution $p_\omega(\omega)$ (Fig. 1b). The latent space ε has a discontinuous clustered nature following the distribution $q_{\phi_\varepsilon}(\varepsilon)$. A simple continuous MLP mapper cannot generate complex $q_{\phi_\varepsilon}(\varepsilon)$ from a unimodal $p_\omega(\omega)$. For this reason, we consider splitting $q_{\phi_\varepsilon}(\varepsilon)$ on a set of unimodal sub-distributions in such a way that a simple MLP model can be used to generate each unimodal sub-distribution. We use a simple K-means [10] clustering to produce K_ε distinct subsets. Then for all $\varepsilon \in \mathcal{C}_i$, $i = 1, ..., K_\varepsilon$, we trained adversarial auto-encoders (AAE) [11] with the latent space $p_\omega(\omega)$ following the Gaussian distribution. Alternatively, one can use Flows to construct a mapper from $p_\omega(\omega)$ to $q_{\phi_\varepsilon}(\varepsilon)$. Each AAE is defined by a pair of the encoder $q_{\phi_\omega}^i(\omega|\varepsilon)$ and the decoder $p_{\theta_\varepsilon}^i(\varepsilon|\omega)$ for each cluster $i \in 1, ..., K_\varepsilon$. The
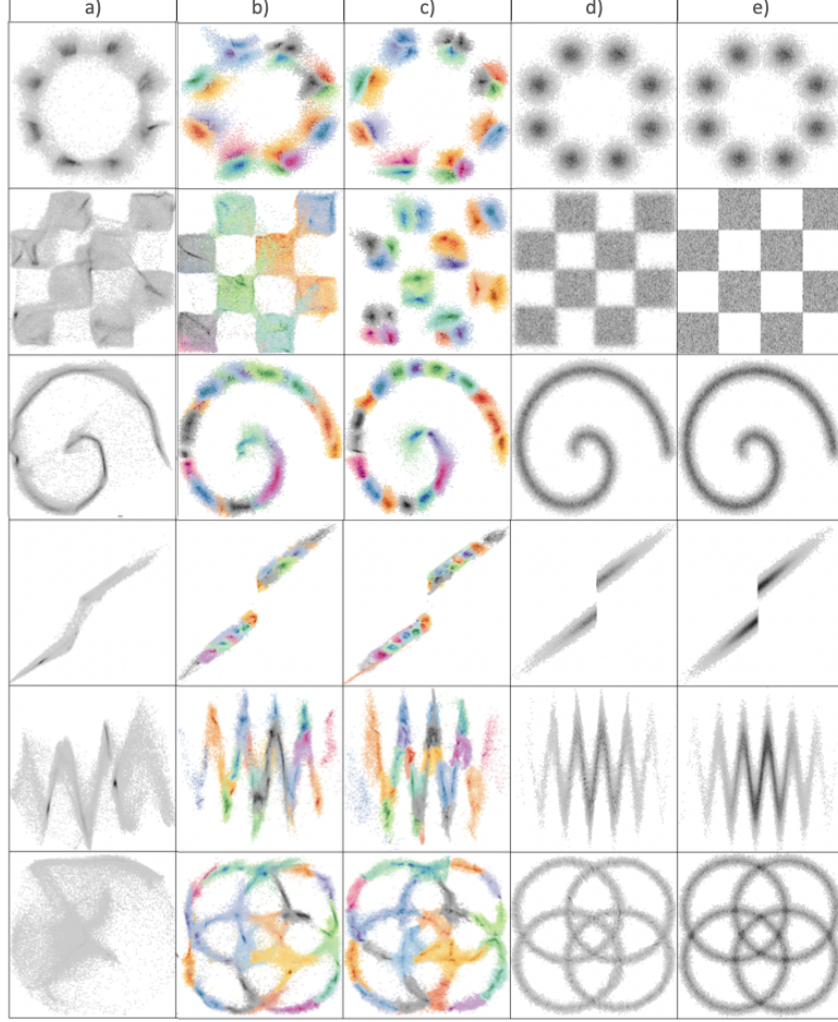
Figure 2: The generation results from different latent space models : a) from a single Gaussian pdf, b) from cluster centers with the Gaussian noise, c) from the mapping network, d) from real data. e) is a target distribution.

training of the AAE is based on the optimization problem:

$$(\hat{\phi}^i_\omega, \hat{\theta}^i_\varepsilon) = \operatorname*{argmax}_{\phi^i_\omega, \theta^i_\varepsilon} I^i_{\phi_\omega}(\mathbf{E}; \mathbf{W}) + \lambda_\varepsilon I^i_{\phi_\omega, \theta_\varepsilon}(\mathbf{W}; \mathbf{E}), \tag{2}$$

where

$$I^i_{\phi_\omega}(\mathbf{E}; \mathbf{W}) = \mathbb{E}_{q^i_{\phi_\varepsilon(\varepsilon)}} \left[\mathbb{E}_{q^i_{\phi_\omega(\omega|\varepsilon)}} \left[\log \frac{q^i_{\phi_\omega}(\omega|\varepsilon)}{q^i_{\phi_\omega}(\omega)} \times \frac{p_\omega(\omega)}{p_\omega(\omega)} \right] \right]$$

$$= \mathbb{E}_{q^i_{\phi_\varepsilon(\varepsilon)}} \left[\mathbb{D}_{\mathrm{KL}}(q^i_{\phi_\omega}(\omega|\mathbf{E}=\varepsilon)||p_\omega(\omega)) \right] - \mathbb{D}_{\mathrm{KL}}(q^i_{\phi_\omega}(\omega)||p_\omega(\omega)),$$

and

$$I^i_{\phi_\omega, \theta_x}(\mathbf{W}; \mathbf{E}) = \mathbb{E}_{q^i_{\phi_\varepsilon(\varepsilon)}} \left[\mathbb{E}_{q^i_{\phi_\omega(\omega|\varepsilon)}} \left[\log \frac{p^i_{\theta_\varepsilon}(\varepsilon|\omega)}{q^i_{\phi_\varepsilon}(\varepsilon)} \right] \right] = H_{\phi^i_\varepsilon}(\mathbf{E}) - H_{\phi^i_\omega, \theta_x}(\mathbf{E}|\mathbf{W}), \tag{3}$$

where $H_{\phi^i_\varepsilon}(\mathbf{E}) = -\mathbb{E}_{q^i_{\phi_\varepsilon(\varepsilon)}} \left[\log q^i_{\phi_\varepsilon}(\varepsilon) \right]$, $H_{\phi^i_\omega, \theta_x}(\mathbf{E}|\mathbf{W}) = -\mathbb{E}_{q^i_{\phi_\varepsilon(\varepsilon)}} \left[\mathbb{E}_{q^i_{\phi_\omega(\omega|\varepsilon)}} \left[\log p^i_{\theta_\varepsilon}(\varepsilon|\omega) \right] \right]$ and λ_ε is a constant for the importance of each term. As $\mathbb{E}_{q^i_{\phi_\varepsilon(\varepsilon)}} \left[\mathbb{D}_{\mathrm{KL}}(q_{\phi_\omega}(\omega|\mathbf{E}=\varepsilon)||p_\omega(\omega)) \right] \geq 0$ and $H_{\phi^i_\varepsilon}(\mathbf{E})$ does not depend on the parameters of the networks, the training objective can be rewritten as: $(\hat{\phi}^i_\omega, \hat{\theta}^i_\varepsilon) = \operatorname{argmin}_{\phi^i_\omega, \theta^i_\varepsilon} \mathbb{D}_{\mathrm{KL}}(q^i_{\phi_\omega}(\omega)||p_\omega(\omega)) + \lambda_\varepsilon H_{\phi^i_\omega, \theta_x}(\mathbf{E}|\mathbf{W})$, as the original AAE loss [11].

2.3 The training of the decoder (stage 3)

The decoder training is performed for both *reconstruction* and *generation* modes based on the optimization problem:

$$\hat{\theta}_{\mathrm{x}} = \underset{\hat{\theta}_{\mathrm{x}}}{\mathrm{argmax}}\, I_{\phi_\varepsilon^*, \theta_{\mathrm{x}}}(\mathbf{E}; \mathbf{X}) + \lambda_{\mathrm{x}} I_{\theta_\varepsilon^*, \theta_{\mathrm{x}}}(\mathbf{E}; \mathbf{X}), \tag{4}$$

where the reconstruction mode corresponds to the term:

$$I_{\phi_\varepsilon^*, \theta_{\mathrm{x}}}(\mathbf{E}; \mathbf{X}) = \mathbb{E}_{p_{\mathbf{x}}(\mathbf{x})} \left[\mathbb{E}_{q_{\phi_\varepsilon^*}(\varepsilon|\mathbf{x})} \left[\log \frac{p_{\theta_{\mathrm{x}}}(\mathbf{x}|\varepsilon)}{p_{\mathbf{x}}(\mathbf{x})} \times \frac{\hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})}{\hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})} \right] \right]$$
$$= -H_{\phi_\varepsilon^*, \theta_\varepsilon}(\mathbf{X}|\mathbf{E}) - \mathbb{D}_{\mathrm{KL}}(p_{\mathbf{x}}(\mathbf{x})||\hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})) + H(p_{\mathbf{x}}(\mathbf{x}); \hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x}))$$

with $\quad H_{\phi_\varepsilon^*, \theta_\varepsilon}(\mathbf{X}|\mathbf{E}) \quad = \quad \mathbb{E}_{p_{\mathbf{x}}(\mathbf{x})} \left[\mathbb{E}_{q_{\phi_\varepsilon^*}(\varepsilon|\mathbf{x})} [\log p_{\theta_{\mathrm{x}}}(\mathbf{x}|\varepsilon)] \right], \quad \mathbb{D}_{\mathrm{KL}}(p_{\mathbf{x}}(\mathbf{x})||\hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})) \quad = \quad$
$\mathbb{E}_{p_{\mathbf{x}}(\mathbf{x})} \left[\log \frac{p_{\mathbf{x}}(\mathbf{x})}{\hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})} \right]$ and $H(p_{\mathbf{x}}(\mathbf{x}); \hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})) \quad = \quad -\mathbb{E}_{p_{\mathbf{x}}(\mathbf{x})} [\log \hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})]$ and the generation mode corresponds to the term:

$$I_{\theta_\varepsilon^*, \theta_{\mathrm{x}}}(\mathbf{E}; \mathbf{X}) = \mathbb{E}_{p_{\mathbf{x}}(\mathbf{x})} \left[\mathbb{E}_{p_\omega(\omega)} \left[\mathbb{E}_{p_{\theta_\varepsilon^*}(\varepsilon|\omega)} \left[\mathbb{E}_{p_{\theta_{\mathrm{x}}}(\mathbf{x}|\varepsilon)} \left[\log \frac{p_{\theta_{\mathrm{x}}}(\mathbf{x}|\varepsilon)}{p_{\mathbf{x}}(\mathbf{x})} \times \frac{\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x})}{\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x})} \right] \right] \right] \right]$$
$$= \mathbb{E}_{p_{\theta_\varepsilon(\varepsilon)}} [\mathbb{D}_{\mathrm{KL}}(p_{\theta_{\mathrm{x}}}(\mathbf{x}|\mathbf{E} = \varepsilon)||\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x}))] - \mathbb{D}_{\mathrm{KL}}(p_{\mathbf{x}}(\mathbf{x})||\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x})),$$

with $p_{\theta_\varepsilon^*}(\varepsilon) = \mathbb{E}_{p_\omega(\omega)} [p_{\theta_\varepsilon}(\varepsilon|\omega)]$ and $\mathbb{D}_{\mathrm{KL}}(p_{\mathbf{x}}(\mathbf{x})||\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x})) = \mathbb{E}_{p_{\mathbf{x}}(\mathbf{x})} \left[\log \frac{p_{\mathbf{x}}(\mathbf{x})}{\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x})} \right]$ and λ_{x} is a constant controlling the trade-off between the two terms, $\hat{p}_{\mathbf{x}}$ and $\tilde{p}_{\mathbf{x}}$ denote the distributions of reconstruction and generated data, respectively. Since $H(p_{\mathbf{x}}(\mathbf{x}); \hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})) \geq 0$ and $\mathbb{E}_{p_{\theta_\varepsilon(\varepsilon)}} [\mathbb{D}_{\mathrm{KL}}(p_{\theta_{\mathrm{x}}}(\mathbf{x}|\mathbf{E} = \varepsilon)||p_{\theta_{\mathrm{x}}}(\mathbf{x}))] \geq 0$, the above optimization problem can be reduced to:

$$\hat{\theta}_{\mathrm{x}} = \underset{\hat{\theta}_{\mathrm{x}}}{\mathrm{argmin}}\, H_{\phi_\varepsilon^*, \theta_\varepsilon}(\mathbf{X}|\mathbf{E}) + \mathbb{D}_{\mathrm{KL}}(p_{\mathbf{x}}(\mathbf{x})||\hat{p}_{\theta_{\mathrm{x}}}(\mathbf{x})) + \lambda_{\mathrm{x}} \mathbb{D}_{\mathrm{KL}}(p_{\mathbf{x}}(\mathbf{x})||\tilde{p}_{\theta_{\mathrm{x}}}(\mathbf{x})).$$

3 Experiments and Conclusions

	single Gaussian pdf	noisy cluster centers	mapping network	real data
Eight Gaussians	2.339	0.053	0.065	0.016
Checkerboard	0.337	0.034	0.015	0.009
Two Spirals	1.771	0.058	0.062	0.011
Abs	0.133	0.021	0.029	0.019
Sinewaved cube	0.062	0.023	0.024	0.021
Four circles	0.031	0.034	0.044	0.035

Table 1: The reconstruction results for the different latent space models: latent vector sampled from a single Gaussian pdf, from cluster centers with Gaussian noise, from the mapping network output and from random training data latents.

We perform generation experiments on 2D datasets using different ways for latent space modeling. In the first setting ε is sampled from the Gaussian probability density function. Then we cluster the ε-space using K-Means. In the second setting we place the Gaussian in the cluster centers and use this as input. Finally, we train an individual network for each cluster to shape the Gaussian closer to the real shape of the cluster (stage 2 of the training). For generation we can also use ε from the subset used to train the encoder. This case is an extreme case when the number of clusters is equal to the size of the dataset. We show the results of the generation in Fig. 2.

Covering the latent space with the clusters which are approximated by simple fully connected layers leads to state-of-the-art results (the fourth column in Fig. 2) for the generative models which are not based on INNs.

We show the Mean Square Error in Table 1 to demonstrate reconstruction error. We fix the noise vectors and we take the latent vector directly from the output of the encoder from Stage 1. The

training stage of generation is different: we take the latent vector from a single Gaussian pdf, from cluster centers with the Gaussian noise, from the mapping network and from real data. We notice that despite poor generation, network can still perform good results in reconstruction with simplest latent space modeling (one Gaussian) which is a sign of overfitting. It is also interesting to note that the better generation is, the worth are the results of reconstruction and vice versa. Modeling the latent space as discontinuous allow us to marry the mode of reconstruction and generation. In the limit case with number of clusters equal to the number of points in the train set we get the best results.

Acknowledgments and Disclosure of Funding

References

[1] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville. Adversarially learned inference. *ArXiv*, abs/1606.00704, 2017.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[3] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019.

[4] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[6] Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. Survae flows: Surjections to bridge the gap between vaes and flows. In *NeurIPS*, 2020.

[7] Zhenliang He, Meina Kan, and Shiguang Shan. Eigengan: Layer-wise eigen-learning for gans. In *International Conference on Computer Vision (ICCV)*, 2021.

[8] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020.

[10] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[11] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015.