Latent Space Oddity: On the Curvature of Deep Generative Models

Théo Danielou
theo.danielou@telecom-sudparis.eu
Master MVA - ENS Paris-Saclay
France

Clément Weinreich
cweinrei@ens-paris-saclay.fr
Master MVA - ENS Paris-Saclay
France

Elie Bakouch
elie.bakouch@dauphine.eu
PSL Research University
France

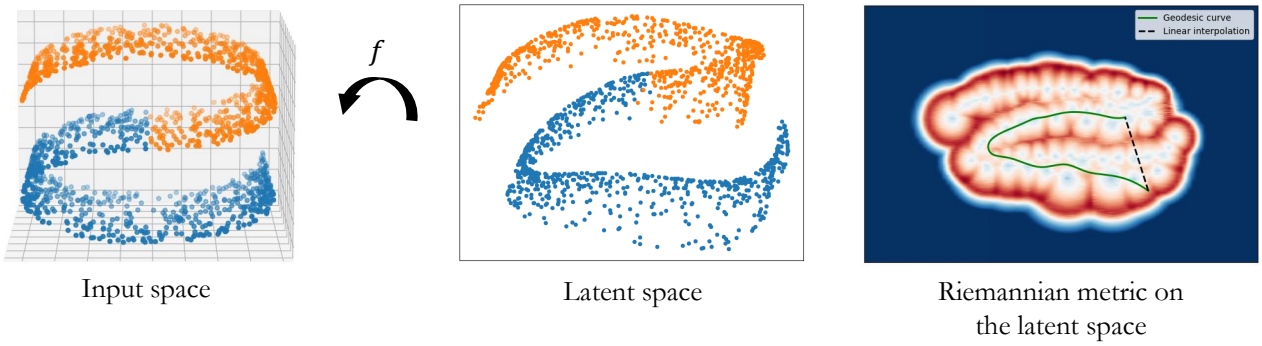


Figure 1: A Variational Auto-Encoder (VAE) is trained on the Z-shaped data distribution living in the input space $\mathcal{X}$. The VAE encoded a lower-dimensional representation of the data into the two-dimensional latent space $\mathcal{Z}$ (middle). An approximation $\overline{\mathbf{M}}_z$ of a Riemannian metric is computed on the latent space using the mean and variance functions of the generator $f : \mathcal{Z} \to \mathcal{X}$ (right). Thanks to a well-behaved variance function based on a Radial Basis Function neural network, we obtain a meaningful curvature of the latent space allowing the computation of geodesics on the data manifold.

ABSTRACT

This report provides an in-depth analysis of the paper "Latent Space Oddity: On the Curvature of Deep Generative Models" [2]. It explores the author's approach of using Riemannian geometry to understand and manipulate the latent spaces of Variational Autoencoders (VAEs). The non-linearity of the generator (or decoder) of these models results in a latent space that presents a distorted representation of the input space. The study addresses this distortion by characterizing it with an approximation of a Riemannian metric, that benefits from an original generator architecture designed to enhance the accuracy of variance estimates. The report examines the theoretical foundations and methodologies proposed for estimating the curvature in the latent space of these models. We further deepen our analysis through experiments showcasing the efficacy and relevance of the method, concluding with a discussion on its potential limitations. Our code is publicly available on GitHub *https://github.com/Clement-W/latent_space_oddity_MVA*.

1 INTRODUCTION

Deep generative models, particularly Variational Autoencoders (VAEs) [11], have revolutionized the field of unsupervised learning, providing profound insights into the intricate world of data generation. The paper we studied here [2] delves into the nuanced exploration of the latent spaces, and especially the VAEs, through a Riemannian geometry perspective. A significant aspect of this study is the exploration of the latent space in VAEs, particularly for the purpose of image interpolation. The capability to interpolate between different images by traversing the latent space provides an avenue for understanding and visualizing how VAEs perceive and reconstruct data, offering a tangible perspective on the model's internal representations.

The VAE's edge over traditional Auto-Encoders (AEs) lies in its probabilistic approach to encoding inputs into latent space. Whereas AEs learn deterministic mappings, VAEs benefit from the re-parametrization trick [11] and take into account the data distribution, leading to a more robust and interpretable latent space. This characteristic is particularly beneficial for tasks that require a deeper understanding of the data structure, like interpolation or generative tasks. The architecture of a VAE is composed of two primary components: the encoder and the decoder (also called the generator). The encoder, denoted as $q_\phi(z|x)$, maps the input data $x \in \mathcal{X}$ with $\mathcal{X} = \mathbb{R}^D$ to a latent distribution in $\mathcal{Z} = \mathbb{R}^d$ characterized by parameters such as mean μ_ϕ and variance σ^2_ϕ that are neural networks. The decoder f attempts to reconstruct the input data x from the latent representation z:

$$x \approx f(z) = \mu_\theta(z) + \sigma_\theta \odot \epsilon$$

with the mapping functions (that are neural networks) $\mu_\theta : \mathcal{Z} \to \mathcal{X}$ generating a surface in $\mathcal{X}$, and $\sigma_\theta : \mathcal{Z} \to \mathbb{R}^D_+$ capturing the uncertainty of the reconstruction. The random vector ϵ follows a standard normal distribution, which allows the sampling from the latent distribution when applying the re-parametrization trick. The

likelihood can also be defined as sampling a Gaussian distribution $p_\theta(x|z) = \mathcal{N}(x; \mu_\theta(z), \mathbb{I}_D \sigma^2_\theta(z))$.

Some previous work already explored the geometrical structure of probabilistic generative dimensionality reduction models using tools of Riemannian geometry [17]. Similarly to the approach of [2], they treat the latent variable model as a Riemannian manifold and use the expectation of the Riemannian metric to define interpolation paths and measure distance between latent points. At the same time as [2] was conducting their research, studies by [4] and [15] were also exploring similar themes in the field. The authors of [4] approach the estimation of Riemannian geometry in latent variable models by using an importance-weighted autoencoder (IWAE) framework [3] that allows modeling more complex data distributions. They approximate the shortest path on the manifold using a neural network, in particular by parametrizing a curve in the latent space with distances measured based on the generative model's distortions. The authors of [15] present algorithms for computing geodesic curves, providing an intrinsic measure of distance on the Riemannian manifold learned by deep generative models. Surprisingly, their experiments suggest that manifolds learned by VAEs have little curvature, indicating that linear paths in latent space closely approximate geodesics on the generated manifold. According to our analysis and our experiments these findings could be due to the absence of meaningful variance estimation. The paper presented in this report is mainly based on [17], and applies the same theorems in the context of VAEs, in particular by introducing a Radial Basis Function Neural Network [14] to model the variance of the decoder, which leads to an accurate estimation of the curvature of the latent space as one can see in Figure 1.

In Section 2.1, we explore how data distributions in Variational Autoencoders (VAEs) can be conceptualized as high-dimensional manifolds, highlighting the role of the generator's Jacobians in shaping the latent space. Section 2.2 delves into the challenges and novel approaches for integrating Riemannian geometry with the stochastic nature of VAE generators. We present a theorem for estimating a Riemannian metric, adapting it to the stochasticity inherent in VAEs. In Section 2.3, we critique traditional variance approximations in VAEs and introduce a refined method using Radial Basis Function (RBF) for more accurate variance modeling, crucial for the latent space's geometry. Section 3, presents the results of our experiments with the Fashion-MNIST dataset, demonstrating the practical benefits of the theoretical developments in improving latent interpolations and data representation. Section 4, offers a critical analysis of our findings, discussing their significance, potential limitations, and applicability to various domains. Section 5, contextualizes our study within the existing body of research, highlighting how more recent research papers deal with the latent space of generative models and made improvements to its geometric interpretation. Finally, we conclude this study in Section 6.

2 MAIN CONTENT OF THE PAPER

2.1 Geometric interpretation of representation learning

According to the manifold hypothesis [7], high-dimensional data, despite its apparent complexity, often adheres to a simpler underlying structure, typically represented as a manifold, or a lower-dimensional surface embedded within the larger dimensional space. Performing computations within these high-dimensional environments presents significant challenges. A practical approach is to parameterize the surface in $\mathcal{X}$ by a low-dimensional variable $\mathbf{z} \in \mathcal{Z}$ created with a suitable smooth generator function $f : \mathcal{Z} \to \mathcal{X}$. This generator defines a surface in the input space from a latent representation which remains difficult to interpret in terms of geometry. It can easily be shown that the natural distance in $\mathcal{Z}$ is changing locally as it depends on the Jacobian of the generator. For a latent point $\mathbf{z}$ and $\Delta\mathbf{z}_1$, $\Delta\mathbf{z}_2$ infinitesimal distances. By Taylor's theorem we have $f(\mathbf{z} + \Delta\mathbf{z}_1) \approx f(\mathbf{z}) + \Delta\mathbf{z}_1 J_\mathbf{z}$ with $J_\mathbf{z} = \left.\frac{\partial f}{\partial \mathbf{z}}\right|_{\mathbf{z}=\mathbf{z}}$. Hence we compute the squared distance as:

$$\begin{aligned}\|f(\mathbf{z} + \Delta\mathbf{z}_1) - f(\mathbf{z} + \Delta\mathbf{z}_2)\|^2 &\approx \|(\Delta\mathbf{z}_1 - \Delta\mathbf{z}_2) J_\mathbf{z}\|^2 \\ &= (\Delta\mathbf{z}_1 - \Delta\mathbf{z}_2)^\top (J_\mathbf{z}^\top J_\mathbf{z})(\Delta\mathbf{z}_1 - \Delta\mathbf{z}_2)\end{aligned}$$

As distances in $\mathcal{Z}$ depend on J_z, the latent space cannot be considered as an Euclidean space but rather as a curved space. Hence, to compute distances on the high-dimensional input space, it makes sense to seek the shortest curve $\gamma_t : [0, 1] \to \mathcal{Z}$ in the latent space. As the low-dimensional space $\mathcal{Z}$ is not equipped with an explicit metric, the length in the input space can be measured by mapping the shortest curve through the generator f. The curve γ_t has length $\int_0^1 \dot{\gamma}_t dt$ with $\dot{\gamma}_t = \frac{d\gamma_t}{dt}$. Thus, we measure lengths in the input space with:

$$\begin{aligned}\text{Length}[f(\gamma_t)] = \int_0^1 \left\|\dot{f}(\gamma_t)\right\| dt &= \int_0^1 \left\|J_{\gamma_t}\, \dot{\gamma}_t\right\| dt, \quad J_{\gamma_t} = \left.\frac{\partial f}{\partial z}\right|_{z=\gamma_t} \\ &= \int_0^1 \sqrt{(J_\gamma \dot{\gamma})^\top (J_\gamma \dot{\gamma})}\, dt \\ &= \int_0^1 \sqrt{\dot{\gamma}^\top \mathbf{M}_\gamma \dot{\gamma}}\, dt, \quad \mathbf{M}_\gamma = J_\gamma^\top J_\gamma\end{aligned}$$

Hence, the length of the curve along the surface on $\mathcal{X}$ can be directly computed in the (curved) latent space using the locally defined norm $\sqrt{\dot{\gamma}^\top \mathbf{M}_\gamma \dot{\gamma}}$. We can then define the distance between two points $\mathbf{z}_1$, $\mathbf{z}_2$ as the length of the shortest curve connecting these two points on the manifold, which we call a geodesic:

$$\gamma_t^{(\text{geodesic})} = \underset{\gamma_t}{\text{argmin}}\, \text{Length}[f(\gamma_t)], \quad \gamma_0 = z_0, \quad \gamma_1 = z_1.$$

When the generator function is smooth enough, $\mathbf{M}_\gamma$ represents a smoothly changing inner product. In that context, we can define $\mathbf{M}_\gamma : \mathcal{Z} \to \mathbb{R}^{dxd}$ (with d the dimension of $\mathcal{Z}$) as a Riemannian metric, i.e. a smooth function that assigns a symmetric positive definite matrix to any point in $\mathcal{Z}$. Nonetheless, when dealing with generative models, it is common to encounter stochastic generators. This aspect necessitates adjustments to the Riemannian framework to ensure its suitability and effectiveness in these contexts.

2.2 A Riemannian perspective to stochastic generators

The decoder $f : \mathcal{Z} \to \mathcal{X}$ of the VAE maps the latent space to the input space is a generator, which we define as follows:

$$f(z) = \mu_\theta(z) + \sigma_\theta(z) \odot \epsilon \tag{1}$$

This method of characterizing the VAE decoder through both mean and variance offers a somewhat unconventional yet effective means for a more explicit and adaptable representation of the data distribution generated by the model. This can be particularly beneficial in scenarios where the data distribution is complex, or when it is essential to model distinct uncertainties in data generation. Because of ϵ, the generator function f is not deterministic but stochastic, which means that the direct use of the Riemannian metric $\mathbf{M}_z = J_z^T J_z$ is not feasible. The following theorem focuses on obtaining an approximation of this Riemannian metric in the context of our stochastic generator.

Theorem 1. *If the stochastic generator f has mean and variance functions that are at least twice differentiable, then the expected metric equals*

$$\overline{\mathbf{M}}_z = \mathbb{E}_{p(\epsilon)}[\mathbf{M}_z] = (J_z^{(\mu)})^T (J_z^{(\mu)}) + (J_z^{(\sigma)})^T (J_z^{(\sigma)})$$

where $J_z^{(\mu)}$ and $J_z^{(\sigma)}$ are the Jacobian matrices of the mean and variance functions.

Proof. We have $J_z = \frac{\partial f(z)}{\partial z}$ with $f(z)$ defined in equation 1. By denoting $A = J_z^{(\mu)}$ and B the Jacobian of $\sigma(z) \odot \epsilon$ with respect to z, we have:

$$\begin{aligned}\overline{\mathbf{M}}_z = \mathbb{E}_{p(\epsilon)}[\mathbf{M}_z] &= \mathbb{E}_{p(\epsilon)}[(A+B)^T(A+B)] \\ &= \mathbb{E}_{p(\epsilon)}[A^TA + A^TB + B^TA + B^TB]\end{aligned}$$

Since A is not random, we have: $\mathbb{E}_{p(\epsilon)}[A^TA] = (J_z^{(\mu)})^T(J_z^{(\mu)})$. By the linearity of expectation and considering that $\mathbb{E}[\epsilon] = 0$, we obtain $\mathbb{E}[A^TB] = \mathbb{E}[B^TA] = 0$. Expanding the expression $\mathbb{E}_{p(\epsilon)}[B^TB]$ and knowing that $Var[\epsilon] = I_D$, we ultimately derive that $\mathbb{E}_{p(\epsilon)}[B^TB] = (J_z^{(\sigma)})^T(J_z^{(\sigma)})$ □

As explained in Section 2.1, for $\mathbf{M}_z$ to qualify as a Riemannian metric, it is required to be smooth, which necessitates that both μ_θ and σ_θ are C^1 functions. In appendix A of [2] the calculation of the geodesics differential equation necessitates differentiating $\overline{\mathbf{M}}_z$, which implies that the mean and variance functions must be twice differentiable as stated in the previous theorem.

One advantage to be noted here is that defining such an approximation of the Riemannian metric does not require any specific training as it is directly derived from the parameters of the generator.

2.3 Meaningful variance functions to ensure proper geometry

The geometry of the latent space, as we define it, is influenced by the mean and variance parameters of the generator via the metric $\overline{\mathbf{M}}_z$. This metric is crucial for calculating geodesics in this space. The concept of the length of a path is here defined by $\overline{\mathbf{M}}_z$, that is, by the Jacobian of the mean and variance of the decoder. This implies that significant changes in variance estimation of the VAE will greatly impact the path length. The variance of the generator, defined as a 'classical' neural network, should provide good estimations in regions close to the data. However, predicting the behavior of this variance outside the data support is challenging. Moreover, previous work ([20], [6], [16]) showed that neural networks cannot be guaranteed to extrapolate effectively outside the support of the data distribution. Hence, practical variance estimates with σ_θ in regions without data are uninformative. Ideally, the variance estimation must match the distribution of the data, and the uncertainty must grow by moving away from the data. To meet these two conditions, the authors suggest modeling the precision $\beta_\psi(z) = \frac{1}{\sigma_\psi^2}$ using a Radial Basis Function Neural Network (RBFNN) [14]. The RBFNN can be expressed as:

$$\beta_\psi(z) = Wv(z) + \xi$$

$$\text{with } v_k(z) = \exp\left(-\lambda_k \|z - c_k\|_2^2\right),\ k = 1, \ldots, K_{rbf},$$

and $W \in \mathbb{R}_{+*}^{D \times K_{rbf}}$ the positive weights, and ξ as positive constants to prevent division by zero. As depicted in Figure 2, the c_k are the centers of the K_{rbf} radial basis functions. To obtain these centers,

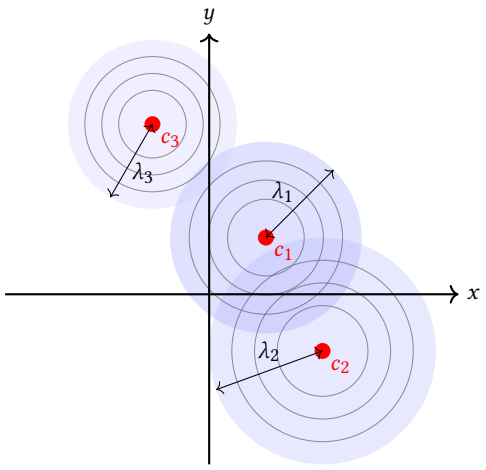


Figure 2: Illustration of multiple Radial Basis Function (RBF) centers on a 2D grid. Each center c_k has an associated influence radius determined by λ_k. The concentric circles represent level sets of the RBF's intensity.

the training dataset is fed into the encoder of the VAE to obtain its latent space representations. Subsequently, a K-Means clustering algorithm is applied to these latent representations to determine the cluster centers c_k and their corresponding cluster groups C_k. The λ_k corresponds to the bandwidths of the K_{rbf} radial basis functions, and are defined as:

$$\lambda_k = \frac{1}{2}\left(\frac{a}{|C_k|}\sum_{z_j \in C_k} \|z_j - c_k\|_2\right)^{-2}$$

with $a \in \mathbb{R}_+$ the curvature hyperparameter. As depicted in Figure 2, the λ_k corresponds to the radius of influence of the associated center c_k. Finally, the weights W are optimized with projected

gradient descent on the mean square error between the standard variance estimation and the inverse of the output of the RBFNN. This allows us to obtain the desirable properties of the variance function with the correct range for the variance estimation. The precision β_ψ can be seen as akin to a Dirac delta function when a tends to 0. It exhibits its highest values at the cluster centers, and these values progressively approach 0 as the distance increases from these central points. Since the clusters are situated in areas of the data space with high data density, the variance σ^2_ψ, which is the inverse of precision, aligns closely with the data distribution. This alignment results in a gradual increase in variance as one moves away from these densely populated data zones. Overall, the RBFs can be interpreted as a Gaussian mixture model where K_{rbf} corresponds to the number of Gaussians, and a^2 is a multiplicative factor of the covariances.

3 CONDUCTED EXPERIMENTS

In order to assess the correctness of the method, we re-implemented a significant portion of the proposed method to test it in different settings. Our code is available on GitHub: *https://github.com/Clement-W/latent_space_oddity_MVA*. The codebase provided by the authors of the original paper [1] has been used for computing geodesics according to a metric tensor. For these experiments, we used the Fashion-MNIST [18] dataset which consists of 28x28 grayscale images associated with a label from 10 classes. This dataset is more challenging than MNIST [5] which is used in the experiments of the original paper. To facilitate visualization of the latent space, we restricted the dataset to 3 classes (0: T-shirt/top, 1: Trouser, 7: Sneaker) which makes up a dataset of 18 000 images.

3.1 Training procedure

The following experiments were conducted with a VAE with smooth activation functions as detailed in table 1.

Hyperparameters	**Value**
Input Dimension	28×28
Layer dimensions of the encoder	64, 32
Layer dimensions of the decoder	32, 64
Latent Space Dimension	2
Hidden Layer Activation	Tanh
Encoder Output (μ_z) Activation	Identity
Encoder Output ($\log(\sigma_z^2)$) Activation	Softplus
Decoder Output (μ_x) Activation	Sigmoid
Decoder Output ($\log(\sigma_x^2)$) Activation	Softplus

Table 1: VAE Hyperparameters

This model was trained on 800 epochs with a batch size of 128 using the Adam optimizer [10] and a learning rate of 8×10^{-5}. The loss function assumes a Gaussian prior and posterior for the latent space and follows the variational lower bound presented by [11]:

$$\mathcal{L}_{\text{VAE}} = P(X|Z) + r \cdot Q(Z|X) + r \cdot P(Z)$$

Where X corresponds to the observed data, Z are the latent variables and we have:

- the reconstruction loss $P(X|Z)$, which measures the likelihood of the data given the latent variables:

$$P(X|Z) = \frac{1}{2} \sum_{i=1}^{N} \left(D \cdot \log(\sigma_x^2)_i + \frac{\sum_{j=1}^{D} (x_{j_i} - \mu_{x_{j_i}})^2}{\sigma_x^2} \right)$$

  where N is the number of samples, D is the dimensionality of the data, σ_x^2 is the variance of the reconstructed data, and μ_x is the mean of the reconstructed data from the decoder;
- the KL divergence term $Q(Z|X)$, which acts as a regularizer by measuring the divergence between the approximate posterior and the prior distribution of the latent variables:

$$Q(Z|X) = -\frac{1}{2} \sum_{i=1}^{N} \log(\sigma_z^2)_i$$

  where σ_z^2 is the variance of the latent variables as encoded by the encoder;
- the prior term $P(Z)$, which promotes the distribution of the latent variables to match the prior distribution:

$$P(Z) = \frac{1}{2} \sum_{i=1}^{N} \left(\sum_{k=1}^{d} \mu_{z_{k_i}}^2 + \sigma_z^2 \right)$$

  where d is the dimensionality of the latent space, μ_z is the mean of the latent variables, and σ_z^2 is their variance.

Note that we worked with the log variance for numerical stability and to ensure the positivity of the variance. A scaling factor r was used at the beginning of the training to allow an unrestricted exploration of the latent space. This scaling factor was incrementally increased, thereby slowly imposing the Gaussian prior and posterior constraints. The scaling reached $r = 1$ at one-third of the training duration, remaining at this level for the remainder of the training process. After training the VAE, we trained several RBFNNs with different values for the curvature parameter a and the number of centroids K_{rbf}, by following the procedure described in Section 2.3.

3.2 Results and study of the variance estimation

This section presents an analysis of the data representation and variance estimation in the latent space of VAEs using an RBFNN. Figure 3 illustrates the VAE's ability to disentangle and encode input data into a coherent and organized two-dimensional latent space. The decoded reconstructions reveal how the mean output of the decoder changes according to variations along the latent dimensions. The encoded data in the latent space shows distinct clusters, which indicates the effectiveness of VAEs in learning the data manifold. Figure 4 represents the (log) variance estimation of the decoder of the VAE, projected into the latent space. It highlights areas of the latent space where the model is more or less certain about the reconstructed inputs, which shows how poorly the variance is estimated. Comparing with the points in the latent space shown in Figure 3, we observe that the area where the decoder is strongly confident is not accurate. Furthermore, the area of high uncertainty located between the green and blue (sneakers and t-shirt classes) points seems not to closely reflect actual uncertainty. Figure 5 extends our analysis to the variance estimated

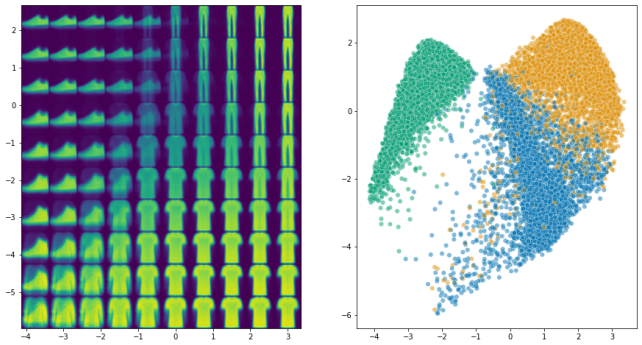


Figure 3: Visualisation of encoded input data points in the two-dimensional latent space (right) and the corresponding decoded image for 10×10 regular points in the latent space (left).

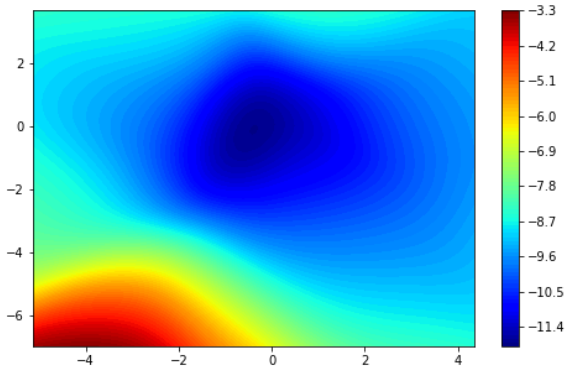


Figure 4: Visualization of the logarithm of the summed standard deviation $\log(\sum_{j=1}^{D} \sigma_j(z))$ across the latent space. Each point on the contour plot corresponds to the collective variance at that location, estimated by the VAE's decoder. The color intensity represents the magnitude of variance, with cooler colors indicating lower variance and warmer colors signifying higher variance.

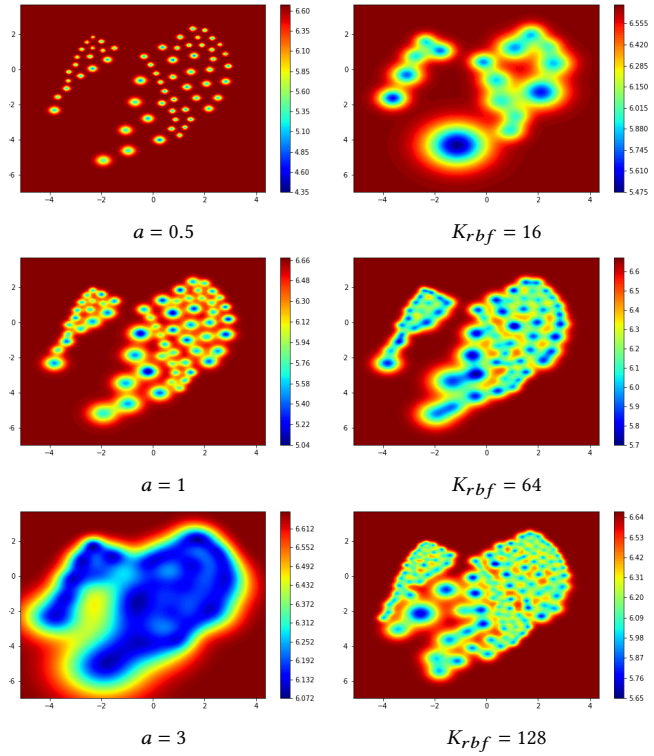
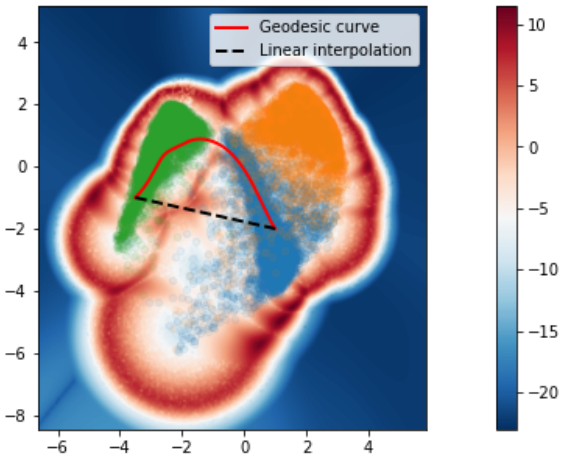


Figure 5: Visualization of variance estimation using an RBFNN across different configurations by plotting $\log(\sum_{j=1}^{D} \sigma_j(z))$. The first column illustrates the effect of varying the curvature parameter a while keeping K_{rbf} fixed at 64. The second column shows the impact of varying K_{rbf} while keeping a fixed at 1.25.

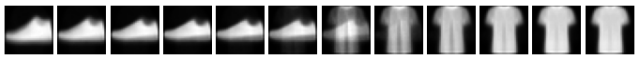
by the RBFNN. Without focusing too much on the different parameters, we first observe that the use of the RBFNN allowed to model the variance and the uncertainty more accurately. The left column of images displays how the curvature parameter a affects the model's uncertainty in its estimation. Lower values of a depict a more concentrated estimation of variance around the RBF centers, while higher values distribute this estimation more diffusely across the space. This was expected as the variance is inversely proportional to a. In contrast, the right column explores the effect of the number of RBF centers K_{rbf}. A small number of centroids create a coarse estimation of the variance (similar to underfitting), while increasing this number is observed to refine the model's discernment of high-variance regions. When K_{rbf} is too large ($K_{rbf} = 128$ in Figure 5) we observe a behavior similar to overfitting, where some clusters were assigned to uncertainty regions associated with a few points that do not represent the actual distribution of the data. To reflect the variance in a correct way, it seems that one must find a balanced number of clusters, with a curvature parameter that provides a relevant diffusion of the uncertainty around the clusters. From our experiments, an RBFNN with $K_{rbf} = 64$ and $a = 1.25$ provides a good estimation of the variance (corresponds to the row-2 column-2 subfigure in Figure 5). Overall, as stated in section 2.3, we observe that the RBFNN shows similar behavior to a Gaussian mixture model with a scaling the covariance matrix and K_{rbf} controlling the number of Gaussian distributions.

3.3 Meaningful interpolations in the latent space

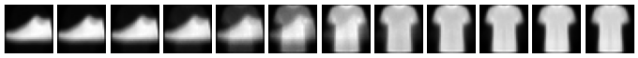
In this section, we analyzed if the approximation of the Riemannian metric could provide more meaningful interpolations compared to the Euclidean metric. Using the same VAE, with an RBFNN with $K_{rbf} = 64$ and $a = 1.25$, we computed $\overline{\mathbf{M}}_z$ and displayed the measure of the latent space on Figure 6. Moreover, the shortest path between two points in the latent space is computed using an Euclidean metric (linear interpolation) and the Riemannian metric (leading to a geodesic). To compute the geodesics in practice, we used a graph solver based on a K-nearest-neighbors (KNN) graph that approximates the manifold's structure. The shortest discrete path



(a) Measure of the latent space with the Riemannian metric $\overline{\mathbf{M}}_z$



(b) Images generated by traversing the geodesic curve

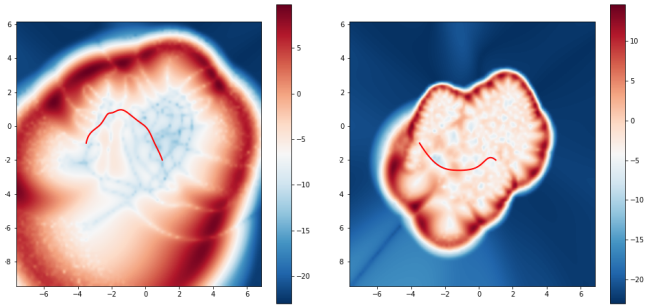


(c) Images generated along a straight-line interpolation

Figure 6: Heatmap visualization of the Riemannian metric, indicated by $\sqrt{det(\overline{\mathbf{M}}_z)}$ computed with an RBFNN with $K_{rbf} = 64$ and $a = 1.25$. Superimposed on this heatmap are two distinct paths: the solid line represents the geodesic curve, which is the shortest path in the manifold's curved geometry, while the dashed line illustrates the straight-line or Euclidean interpolation between two points in the latent space (a). The sequences of images below (b and c) are the VAE's output corresponding to points along these paths. The upper sequence (b) follows the geodesic trajectory, while the lower sequence (c) aligns with the Euclidean interpolation.

is first computed on the graph, then this discrete path is smoothed using a heuristic method. Finally, a cubic spline is used to interpolate the points on the smoothed path, creating a continuous curve approximating the geodesic.

Figure 6 highlights the capacity of the Riemannian metric, derived from the mean output of the VAE's decoder and the RBFNN, in capturing the curvature of the latent space. By definition $\sqrt{\det(\overline{\mathbf{M}}_z)}$ is the geometric volume measure that captures the volume of an infinitesimal area in the input space. Plotting this measure allows to visualize the curvature of the latent space based on the new estimation of the variance of the RBFNN. We observe that the measured distances are large in regions of the latent space where the generator is highly uncertain. This comes from the variance term $(J_z^{(\sigma)})^T(J_z^{(\sigma)})$ of $\overline{\mathbf{M}}_z$ which is large in regions of the latent space where the generator has large variance. Hence the shortest path between two points in the latent space tends to avoid these regions as



$a = 3$ **and** $K_{rbf} = 64$ $\quad$ $a = 1.25$ **and** $K_{rbf} = 128$

Figure 7: Comparative Visualization of Geodesic Paths in VAE Latent Spaces with "extreme" values for the RBFNN Parameters.

we can see with the geodesic in the same figure. When generating the samples along the shortest paths according to the Riemannian and Euclidean metric in Figure 6 (b) and (c), we observe that the Riemannian interpolant gives smoother changes in the generated image. The shoe slowly deforms into pants, then into a t-shirt as it follows the area of low variance. However using a simple linear interpolation without paying attention to the curvature of the latent space gives an abrupt transition in the interpolation between the shoe and the t-shirt, which cannot be desirable depending on the application. For example, in the case of a VAE that aims to learn to reconstruct the image of an organ, Euclidean interpolation tends to generate data that does not respect the real physics of that organ, whereas Riemannian interpolation aims to recreate images that are closer to those it has learned about.

We can also explore the geodesic paths on VAE with RBFNN that have "extreme" values for their hyperparameters a and K_{rbf}. Figure 7 shows the metric $\overline{\mathbf{M}}_z$ computed with two different RBFNNs but with the same VAE. We observe an expected behavior according to the variance estimations shown in Figure 5. The left image of Figure 7 shows the measure of the latent space with the parameter a set to 3. This leads to a very diffused measure which depicts how this parameter influences the curvature of the space. Despite this, the geodesic follows the areas of lowest variance as we would expect. The right image of the same plot illustrates the geodesic path with 128 centroids for the RBFNN. The phenomena of overfitting described in section 3.2 is clearly visible, where centroids with a small number of assigned points are biasing the variance estimation, thereby impacting the Riemannian metric. Thus, the geodesic passes from below, which does not make sense when we look at the distribution of points in the latent space in Figure 3. Both geodesics in Figure 7 connect the same pair of points in the latent space, highlighting the impact of the RBFNN parameters on the geometrical structure of the latent manifold.

Furthermore, to confirm the obligation of having a meaningful variance function to ensure proper geometry in the latent space, Figure 8 displays the measure $\sqrt{\det(\overline{\mathbf{M}}_z)}$ using the VAE's variance

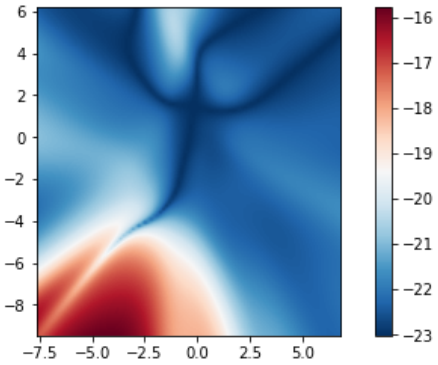


Figure 8: Heatmap visualization of the Riemannian metric, indicated by $\sqrt{det(\overline{\mathbf{M}}_z)}$ computed with the VAE's variance output.

output instead of the estimation of the RBFNN. This measure seems not to be informative on the curvature of the latent space and does not reflect the actual curvature coming from the data manifold.

3.4 Random walks in the latent space

Random walks are a standard technique for exploring high-dimensional spaces, and are particularly useful in revealing the structure and distribution of the data within these spaces. In this section, we consider an unrestricted random walk with Brownian motion under both Euclidean and Riemannian metrics, performed within the latent space of the same VAE and RBFNN with hyperparameters $a = 1.25$ and $K_{rbf} = 64$. Two separate random walks are performed: one that evolves according to the Riemannian metric of the latent space, and another that moves in a standard Euclidean manner. For each step of the Riemannian random walk:

(1) The metric tensor $\overline{\mathbf{M}}_z$ is computed at the current point $\mathbf{z_i}$, representing the local geometry of the latent space at this point.
(2) The eigendecomposition of $\overline{\mathbf{M}}_z$ is computed, yielding to nonegative eigenvalues L (as $\overline{\mathbf{M}}_z$ is positive semi definite) and eigenvectors U.
(3) A random vector ϵ is sampled from a standard normal distribution, representing a random direction in the latent space.
(4) The adjusted direction $\mathbf{v} = UL^{-\frac{1}{2}}\epsilon$ is computed, which ensures that the step taken is consistent with the local geometry of the space, as dictated by the Riemannian metric.
(5) The current point is updated by taking a step in the direction of $\mathbf{v}$: $\mathbf{z_i} = \mathbf{z_i} + s\mathbf{v}$, scaled by the step size s.

For the Euclidean random walk, we simply move the current point $\mathbf{z_i}$ in the direction of the random vector ϵ, scaled by the step size. Figure 9 shows the results obtained from the Euclidean and Riemannian random walk. A key observation is the difference in behavior between random walks under the two metrics. Under the Euclidean metric, the walks tend to drift freely, which leads it to regions with little to no data support. This behavior contrasts with the Riemannian random walk, which tends to stay within the bounds

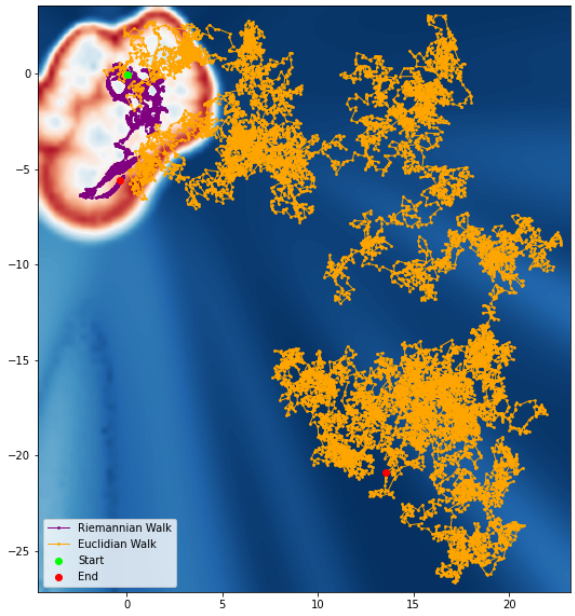


Figure 9: Random walk on the latent space under both the Riemannian and Euclidean metric with 10000 steps and a step size of 0.2.

of the data support. This is attributed to the variance term in the Riemannian metric, which effectively forms a "wall" around the data-rich regions, preventing the random walk from straying too far into data-sparse areas, which reflects the curvature of the latent space. An analogy can be drawn between this behavior and the motion of an entity in a physical landscape. If we imagine the latent space as a three-dimensional landscape of hills and valleys, the red areas representing regions of high variance are akin to steep, towering mountains, while the blue areas of low variance are like comfortable valleys or flat plains where the physical entity can move without strong constraints. Furthermore, these walls not only contain the random walk but also influence the shortest paths to align more closely with the data, creating a more meaningful exploration of the latent space.

4 DISCUSSION

The paper we studied [2] demonstrates that adapting concepts from [17] in the context of Variational Autoencoders (VAEs) addresses the identified problem: enabling the computation of meaningful geodesics in the latent space that corresponds accordingly with the data distribution in the input space. More broadly, it establishes a metric that provides insight into the data distribution of the latent space according to the input space. The use of Radial Basis Function Neural Networks for modeling the variance of the decoder effectively solves the issues encountered when using a conventional neural network. In particular, 8 supports our hypothesis about the conclusion of [15] stating that the learned manifold has little curvature. Using the VAE's variance output to construct the Riemannian metric leads to a relatively flat space in and out of the data support.

In our experiments, we used simple datasets and two-dimensional latent space, making it relatively easy to fine-tune the hyperparameters a and K_{rbf}. However, while it is true that the latent space of a VAE is typically of low dimension by design, in cases involving more complex data, it is not uncommon to encounter latent spaces with dimensions significantly higher than two. In such scenarios, determining and effectively interpreting the effect of these hyperparameters can become challenging due to the lack of a graphical representation. It is also worth noting that the use of the K-Means algorithm could be questioned in these instances due to the curse of dimensionality. To address this issue, one could consider identifying clusters through a projection onto a space of even lower dimension than the latent space.

The RBFNN is trained using a supervised learning approach. As explained in Section 2.3, to learn the values of the new variance, the RBFNN aims to learn from the variance values initially learned by the VAE. However, this variance learned by the VAE has significant flaws. The explanation of the RBFNN's training as a supervised learning approach in [2] was found to be somewhat unclear. Therefore, we aim to provide a clearer exposition of this aspect in our discussion. The initially learned decoder variance values are consistent with the data, as they have been learned to reconstruct the distribution of the input space. The errors stem from the variance values outside of the data. Therefore, during the training of the RBFNN, the goal is to achieve new variance values close to the previous ones in data-rich areas. The new variance values outside the data support are defined to address the issues described in Section 2.3 about extrapolating the variance values.

A possible extension of this work that could replace the RBFNN is the use of Neural fields [19]. Also known as coordinate-based neural networks or implicit neural representations, these neural networks are designed to learn continuous functions. In this context, the neural field would take latent space coordinates as input and output a variance estimation. For training, one could create a loss function that not only considers the difference between the model output and the estimated variance of the VAE, but also incorporates the distribution of data in the latent space for example by penalizing low variance estimations in data-sparse regions. Finally, the neural field would learn a refined variance estimation that aligns with the data distribution in the latent space. These neural networks are particularly effective for tasks involving high-dimensional data and complex geometries. Experimenting with this approach would clarify if neural fields could be a tangible alternative, and potentially overcome some limitations of RBFNNs in challenging settings.

In the original paper, the authors discuss the quality of the approximation of the Riemannian metric by establishing a Theorem (Theorem 2 in [2]), which makes reference to [17] for its proof. The theorem states that $\lim_{D\to\infty} \mathrm{Var}(\mathbf{M}_z) = 0$. However, a review of the latter source reveals that the theorem does not appear to be explicitly stated or elaborated upon in that article.

Finally, we find it important to keep in mind that this work does not aim to improve the performance of the reconstruction performed by μ_θ. Additionally, this method imposes constraints on the initial architecture of the VAE, requiring the activation functions of the VAE to be twice differentiable to utilize Theorem 1. Depending on the scenario, it may then be worthwhile to assess the trade-off between the importance of using activation functions that do not meet this condition and the benefits of incorporating this Riemannian metric. We have experimented with various non-differentiable activation functions such as ReLU and its variants to determine their impact on this metric and have not observed any pathological behaviors. It could therefore be interesting to test this in more complex cases to see if this theoretical constraint holds in practice.

5 RECENT RELATED WORK

In the landscape of generative models, the approach to latent space manipulation varies significantly, particularly when comparing Variational Autoencoders (VAEs) with other models like Generative Adversarial Networks (GANs) and Diffusion Models (DMs). While the method presented in [2] offers a novel perspective on manipulating the variance function in VAEs, its generalization to other models poses certain challenges due to inherent structural differences.

In DMs, a different approach is taken, as seen in [13]. This study introduces the use of a pullback metric, leveraging the known geometry of the input space to derive a corresponding metric in the latent space. Unlike the curvature-based method in VAEs, this approach does not rely on modifying the variance function but instead utilizes the input space geometry to influence the latent space. This method is particularly suited to DMs due to their unique generative process, which differs fundamentally from that of VAEs.

The issue of interpolation and manipulation in GANs is closely tied to their architectural design. [12] explores this by utilizing the StyleGAN architecture, as detailed in [9]. In StyleGAN, certain layers exhibit an approximately Euclidean geometry, in this case meaning the latent space closely mirrors the image space. This proximity allows for direct edits in the StyleGAN layers without the necessity of employing Riemannian geometry. The approach hinges on the unique architectural features of StyleGAN, where latent space manipulation is more straightforward due to its design, in contrast to the more complex latent space structures in VAEs.

A recent study in [8] is particularly relevant to the exploration of latent space geometry in generative models. This work investigates the latent spaces of models like GANs and VAEs, which are defined as a push-forward of a Gaussian measure by a continuous generator. The study focuses on how these models tend to output samples outside the target distribution when learning disconnected distributions. It explores the relationship between model performance and latent space geometry and provides a theoretical framework for understanding the latent space's geometry and proposes a truncation method to improve GAN performance by enforcing a simplicial cluster structure in the latent space.

6 CONCLUSION

To conclude, we have studied an adaptation of a Riemannian framework to the latent space of Variational Autoencoders. Introducing an approximation of a Riemannian metric to measure the curvature of the latent space according to the data manifold, shows the necessity of having a meaningful variance estimation by the generator. To this end, a Radial Basis Function neural network is trained to replace the standard variance estimation of the Variation Autoencoder, which permits better extrapolation of the variance outside the data support. We presented the theoretical aspects of this approach, showing that such method is sound, and experimented with the method in different settings. We studied the behavior of the Radial Basis Function neural network with different hyperparameters, which validated the theoretical analysis. Finally, we computed interpolations and studied the behavior of a random walk in the latent space showing that the approximated Riemannian metric provides a meaningful curvature in the latent space according to the input data manifold. This Riemannian formalism enhances the understanding and handling of the latent space of Variational Autoencoders. This showcases the potential of Riemannian geometry in improving generative models and offering an insightful perspective to future explorations in this field.

REFERENCES

[1] Georgios Arvanitidis. 2021. geometric_ml: Author's GitHub Repository. https://github.com/georgiosarvanitidis/geometric_ml. Accessed: 2023-12-07.

[2] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. 2018. Latent Space Oddity: on the Curvature of Deep Generative Models. arXiv:1710.11379 [stat.ML]

[3] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. 2015. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519* (2015).

[4] Nutan Chen, Alexej Klushyn, Richard Kurle, Xueyan Jiang, Justin Bayer, and Patrick Smagt. 2018. Metrics for deep generative models. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1540–1550.

[5] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.

[6] Kefan Dong and Tengyu Ma. 2022. First Steps Toward Understanding the Extrapolation of Nonlinear Models to Unseen Domains. arXiv:2211.11719 [cs.LG]

[7] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. 2013. Testing the Manifold Hypothesis. arXiv:1310.0425 [math.ST]

[8] Thibaut Issenhuth, Ugo Tanielian, Jérémie Mary, and David Picard. 2023. Unveiling the Latent Space Geometry of Push-Forward Generative Models. arXiv:2207.10541 [cs.LG]

[9] Tero Karras, Samuli Laine, and Timo Aila. 2018. A Style-Based Generator Architecture for Generative Adversarial Networks. *CoRR* abs/1812.04948 (2018). arXiv:1812.04948 http://arxiv.org/abs/1812.04948

[10] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[11] Diederik P Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. arXiv:1312.6114 [stat.ML]

[12] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

[13] Yong-Hyun Park, Mingi Kwon, Junghyo Jo, and Youngjung Uh. 2023. Unsupervised Discovery of Semantic Latent Directions in Diffusion Models. arXiv:2302.12469 [cs.CV]

[14] Qichao Que and Mikhail Belkin. 2016. Back to the future: Radial basis function networks revisited. In *Artificial intelligence and statistics*. PMLR, 1375–1383.

[15] Hang Shao, Abhishek Kumar, and P. Thomas Fletcher. 2017. The Riemannian Geometry of Deep Generative Models. arXiv:1711.08014 [cs.LG]

[16] Xinwei Shen and Nicolai Meinshausen. 2023. Engression: Extrapolation for Nonlinear Regression? arXiv:2307.00835 [stat.ME]

[17] Alessandra Tosi, Søren Hauberg, Alfredo Vellido, and Neil D. Lawrence. 2014. Metrics for Probabilistic Geometries. arXiv:1411.7432 [stat.ML]

[18] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv:cs.LG/1708.07747 [cs.LG]

[19] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. 2022. Neural Fields in Visual Computing and Beyond. arXiv:2111.11426 [cs.CV]

[20] Keyulu Xu, Mozhi Zhang, Jingling Li, Simon S. Du, Ken ichi Kawarabayashi, and Stefanie Jegelka. 2021. How Neural Networks Extrapolate: From Feedforward to Graph Neural Networks. arXiv:2009.11848 [cs.LG]