AN ANALYSIS ON DIMENSIONALITY AND ARCHITECTURE ON GENERATIVE MODELS

by

Ali Emre Gursu

B.S.,Management Information Systems, Boğaziçi University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computational Science and Engineering

Boğaziçi University

2023

ACKNOWLEDGEMENTS

I express my sincere gratitude to all those who supported and contributed to the completion of this project. My advisor, my family, my friends, and my colleagues help balance work and academic load in this tough journey.

First, I extend my heartfelt appreciation to my advisor, Bertan Badur, whose guidance, expertise, and unwavering support were invaluable throughout this journey. Your mentorship has been instrumental in shaping the direction and quality of this study. This opportunity and expertise would not have been possible without his assistance. I wish that I have not harmed his trust in me in any way during this journey. He was the source of my enthusiasm for following an academic degree. I heartfeltedly wished to prove worthy of his trust and support throughout my academic journey.

Secondly, I would like to thank my elder brother, Halit Murat Gursu, for his guidance throughout my life and especially in academic path. Your insights, feedback, and tireless efforts have enriched our research process and yielded meaningful results. His guidance was essential in correcting my path and helped shape my life in a more reasonable and knowledgeable way. His existence and seeing him pave paths for me throughout his life was one of the reasons I was able to follow and come to where I am today.

I am deeply thankful to my family for their constant encouragement and understanding during the phases demanding this project. Your love and encouragement have been my source of strength. My mother, Nuran Gursu, helped ease my tough workload and raised my self-trust at all times with her giving love and radiating smiles. My father, Halil Gursu, taught me to be hardworking at all times and always valued results more than excuses.

I extend my appreciation to my friends and colleagues for their encouragement, insightful discussions, and willingness to help when needed. Your camaraderie has made this journey both enjoyable and fulfilling. At tough times their support and making me feel communal was what helped me continue to pursue this tough path.

This work was made possible by the support and contributions of many, and I am deeply thankful to each and every one of you.

ABSTRACT

AN ANALYSIS ON DIMENSIONALITY AND ARCHITECTURE ON GENERATIVE MODELS

Deep generative models are powerful class of machine learning models. However, a significant amount of computing power and technical knowledge is required to conduct the training process. Even searching for hyperparameters requires a high computational cost. Moreover, there is still ongoing research on methods for evaluating generative models, and owing to the lack of a robust and consistent metric, there are limited comparisons between generative model architectures and algorithms. In this study, we attempted to compare two types of generative model architectures, Generative Adversarial Networks (GANs) and Real-valued Non-Volume-Preserving (NVP) flows, with synthetic datasets as well as with a well known image dataset MNIST. We evaluate their data capturing ability according to data dimensionality and variability. We propose an Minimum Description Length (MDL) based metric to examine the effect of model complexity which is measured as model's parameter count. We provide estimated Kullback-Leibler (KL) divergence and propsed MDL-based metric results.

Our findings indicate that NVP models have the capability to encode more data variability while utilizing fewer parameters when contrasted with GANs for lower dimensional datasets. The proposed MDL-based metric, facilitates selecting suitable architecture in terms of model complexity for a given dataset considering its variability and dimensionality.

Keywords: Generative Models, Generative Adversarial Networks, RealNVP, Deep Learning

ÖZET

ÜRETKEN MODELLERDE BOYUTSALLIK VE MIMARI ÜZERINE ANALIZ

Derin üretici modeller, güçlü bir makine öğrenme modeli sınıfıdır. Ancak eğitim sürecini yürütmek için bile önemli bir hesaplama gücü ve teknik bilgi gerekmektedir. Hiperparametreleri aramak dahi yüksek bir hesaplama maliyetini gerektirir. Dahası, üretici modelleri değerlendirmek için hala devam eden araştırmalar bulunmakta ve sağlam ve tutarlı bir metrik eksikliği nedeniyle üretici model mimarileri ve algoritmaları arasında sınırlı karşılaştırmalar bulunmaktadır. Bu çalışmada iki tür üretici model mimarisini -Üretken Karşıt Ağlar ve Gerçel Değerli Hacim Korumayan akış modellerisentetik veri kümeleri ve iyi bilinen bir görüntü veri kümesi olan MNIST'i kullanarak karşılaştırmaya çalıştık. Veri boyutluluğu ve değişkenliğine göre veri yakalama yeteneklerini değerlendirdik. Model karmaşıklığının -modelin parametre sayısı olarak ölçüldüğü durumda- etkisini incelemek için Minimum Tanım Uzunluğu (MDL) tabanlı bir metrik öneriyoruz. Tahmini Kullback-Leibler (KL) sapma ve önerilen MDL tabanlı metrik sonuçları sunuyoruz.

Bulgularımız, NVP modellerinin, daha düşük boyutlu veri kümeleri için GAN'larla karşılaştırıldığında daha az parametre kullanarak daha fazla veri değişkenliği kodlama yeteneğine sahip olduğunu göstermektedir. Önerilen MDL tabanlı metrik, veri değişkenliği ve boyutluluğunu dikkate alarak belirli bir veri kümesi için model karmaşıklığı açısından uygun mimariyi seçmeyi kolaylaştırır.

Anahtar Kelimeler: Üretken Modeller, Üretken Karşıt Ağlar, RealNVP, Sinir Ağları, Derin Öğrenme

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

2D	Two Dimensional
3D	Three Dimensional
4D	Four Dimensional
ANN	Artificial Neural Network
DGM	Deep Generative Models
EM	Earth Mover
FID	Frechet Inception Distance
GAN	Generative Adverserial Networks
GMM	Gaussian Mixture Model
IS	Inception Score
JS	Jensen Shannon divergence
KL	Kullback–Leibler divergence
KNN	Kth Nearest Neighbour
MDL	Minimum Description Length
MNIST	Modified National Institute of Standards and Technology
NND	Neural Net Divergence
NVP	Non Volume Preserving
PGM	Probabilistic Graphical Models
RGB	Red Green Blue
VAE	Variational Auto Encoder

1. INTRODUCTION

Deep generative models are powerful class of machine learning models that have revolutionized various domains, including computer vision and natural language processing. These models leverage the capabilities of deep neural networks to generate new data closely resembled the training data distribution. Deep generative models are based on unsupervised learning, which enables models to learn from unlabeled data without explicit guidance. Their taxonomy can be simplified into four groups: autoregressive, flow-based, latent variable, and energy-based [1]. Each model architecture offers unique advantages in addressing the challenges that arise within unsupervised learning. They have received considerable attention because of their capabilities in learning complex and high-dimensional distributions. To achieve this, deep generative models require extensive computing resources and technical expertise. Moreover, evaluation methods of trained models are an ongoing research area, and numerous metrics and estimates are used in this manner [2]. A metric should encapsulate information about the quality, diversity, and generalization of generative models [3]. Currently, there is no metric that captures all properties, and most studies visually inspect generated samples to assess models.

There are proposed measures, such as Inception Score (IS) [4] and Fréchet Inception Distance (FID) [5] which have shown promising results for practical applications. In particular, FID has been shown to be robust to image corruption, it correlates well with the visual fidelity of the samples, and can be computed using unlabeled data. Since they yield a one-dimensional score, they are unable to distinguish between different failure cases. It is also shown that most of the aforementioned metrics fail to capture the memorization property of model training [6]. Moreover, there is not many metrics that focus on generalization property of the models [2].

While these models achieve compelling results in specific domains, there is still no clear consensus on which architectures perform objectively better than others. There is

no criteria to assess which architecture to choose based on your specific dataset needs like variability, dimensionality and sharpness. This is partially due to the lack of robust and consistent metric, as well as limited comparisons which put all algorithms on equal footage, including the computational budget required to search for all hyperparameters [7]. We think this as a major blocking factor in this area of research. Due to its diverse use cases and advantages practicioners should be able to choose on generative architecture and complexity based on datasets' dimensionality and variability. It also makes novel architectures hard to assess.

In this work, we attempted to evaluate and compare model architectures on different datasets with different variabilities and dimensionalities to better understand the underlying advantages and assumptions of generative models, and propose a recommendation for which generative model to choose based on specific dataset settings. The main comparisons were performed with Generative Adversarial Networks (GANs) and Real-valued Non-Volume-Preserving (NVP) flow models with respect to the KL divergence and estimated Minimum Description Length (MDL) metric scores. We provide results for changing dimensionality and variability of different datasets to evaluate the model performance against complexity, which is measured as parameter count in our proposed MDL-based metric.

The remainder of this thesis is organized as follows. In Chapter 2, background and literature survey is presented. The background of generative models and those precisely used in this study are introduced and discussed in Section 2.1. In Section 2.2, we discuss how the evaluation of generative models is conducted with different metrics and how difficult it is to come up with a metric that can fit all problems. In Chapter 3, we present our methodology and explain the experimental settings and methods used. We continue with our experiments and results in Chapter 4. Finally, Chapter 5 concludes our findings and proposes future directions for this area of research, along with the limitations of the proposed MDL-based metric.

2. BACKGROUND

2.1. Generative Models

A generative model can be defined as a joint probability distribution for a random variate X. There are several types of generative models with different architectures. They can be mainly divided into two subcategories namely "classical" probabilistic graphical models (PGM) and deep generative models (DGM) [3]. In this study, we are interested in DGMs because they are becoming a leading direction in AI development. Their progressive interest stems from the development of Artifical Neural Networks (ANNs) and their increased computational power to model complex distributions.

Generative models, namely, modeling the underlying data distribution, can be used for various tasks and purposes. If we are to mention a few, they could be used for density estimation, imputation, generation, and structure discovery. Density estimation is vital for weighting data against the model probability distribution and has applications in outlier detection, data compression, and model comparison. It would also help assess uncertainty and contribute to discriminative models. Imputation is the process of dealing with missing values. Generative models can be used to predict missing values more accurately than other methods such as statistical imputation using mean or mode values. Another major factor contributing to the popularity of generative models is data generation. After capturing the underlying data distribution, generative models can be used to sample before unseen data. They are used in various tasks, such as text, audio, image, and video generation, as mentioned previously. A major instance of this phenomenon is deep fake, and research has focused on ways to circumvent it [8]. Generative models can also be used to discover underlying latent structures behind the data-generating distribution. Data representation is vital for the success of machine learning algorithms; therefore, learning latent representations is a major contributor to DGM adaptability [9].

The use of generative models has received increasing attention in recent years due to their versatility in performing diverse tasks such as text generation, image generation, video generation, image-to-image translation, captioning, and text-to-image generation [1]. The enhanced capabilities of generative models have resulted in significant interest from both academia and industry. Generative models are capable of semantic understanding, which is a critical aspect in decision-making as discriminative models lack this capability. The underlying criteria that create objects of interest are of greater interest to generative models than discriminative models, which primarily focus on finding a discriminative criterion to classify objects [10]. Although there is ongoing debate regarding the architecture and methods of generative models, they require vast computational resources and expertise to train. They have a wide range of applications but also face significant challenges, particularly in dealing with high-dimensional and complex data dependencies. Deep generative models can be categorized into four groups based on their architectures: autoregressive, flow-based, latent-variable, and energy-based. Each model architecture is based on different assumptions and has its own loss function and approach to modeling the underlying data distribution. Therefore, different criteria are considered during their training. The choice of generative model architecture is also influenced by factors such as the likelihood approach backed by Probability Theory or a distance-based metric backed by Information Theory, as well as the training behavior and sampling speed of the generative models.

In this investigation, we focus on two distinct types of generative models, namely GANs and NVPs. Specifically, we compare flow-based NVPs with vanilla GANs, which are both latent models that exhibit high sampling speeds but differ in the dimensionality of their latent spaces and their training approach. GANs are capable of utilizing lower-dimensional latent spaces and are trained using JS divergence, while NVPs require the latent space to have the same dimensionality as the data space in order to enforce bijective behavior and are trained using a likelihood approach.

2.1.1. Generative Adverserial Networks

Generative Adverserial Networks (GANs) introduced by Goodfellow [11] in 2014, train in an adverserial game theoretic manner that trains a discriminator and generator network jointly to capture the data distribution. The introduction proves that this algorithm converges to a local optimum through the Nash equilibrium. Conventional artificial neural network architectures are trained by optimizing a loss function using gradient descent. On the other hand, adverserial training plays a min-max game that involves more than one function for optimization and can result in collapse. Although gaining widely used in image generation, video generation, and image translation GANs have many caveats.

GANs have several caveats. They are difficult to train because their training is not stable [12, 13]. Secondly, because of the use of Jensen Shannon divergence as their loss function, they have the possibility of mode collapse [14]. Several solutions have been proposed to stabilize GAN training, such as using different learning rates for discriminator and generator networks and using different loss functions and regularization [5, 15–18]. Their introduction led to greater adaptability of GANs in various domains and different image-related tasks [19–22]. Their probability distributions are implicit; therefore, it is difficult to compare model performance with respect to the target distribution using likelihood based methods. They can learn any probability distribution from the samples if their architecture is correctly chosen and the training is stabilized [23]. Therefore, it is crucial to select the correct architecture for the problem at hand.

GAN training involves the challenge of discovering a Nash equilibrium within a complex, nonconvex game characterized by continuous and high-dimensional parameters. Traditional GAN training relies on gradient descent methods, primarily aimed at minimizing a cost function rather than specifically targeting the Nash equilibrium of the game. As previously mentioned, GAN training may encounter difficulties in achieving convergence [4]. Several different architectures in GANs have been proposed

to eliminate these problems, such as divergence-based GANs, which use the Wasserstein distance [24] rather than a trained discriminator to stabilize the training process. Other techniques include regularization, projections, and the use of multiple generators, discriminators, or both. The WGAN replaces the trained discriminator model with a critic function that calculates the Earth Mover distance between the data and sample distributions. Additionally, architectures such as VEEGAN [25] have been proposed to eliminate mode collapse by forcing an inverse mapping from the data distribution to the latent distribution, similar to flow-based models. However, detecting mode collapse in GANs is difficult and requires visual inspection. Currently, there is no unanimous agreement on which GAN algorithms are definitively superior to others. This lack of consensus is partly due to the lack of reliable and consistent metrics for assessing their performance, and there are limited comparisons that ensure similar settings for all algorithms, considering factors such as the computational resources required to explore various hyperparameters.

Generative Adverserial Nets' loss function is the discriminator's ability to separate syntetic data from training data, and the generator's ability to fool the discriminator. GAN loss function can be expressed as:

$$\mathcal{L}_{\mathrm{GAN}}(G, D) = \mathbb{E}_{x\sim p_{\mathrm{data}}(x)}[\log(D(x))] + \mathbb{E}_{z\sim p_{\mathrm{z}}(z)}[\log(1 - D(G(z)))]. \tag{2.1}$$

In this expression, $\log(D(x))$ is log probability that x came from the data rather than p_g. $G(z;\theta_g)$ is the differentiable function represented by a multilayer perceptron with parameters θ_g.

Several architectures and methods have been proposed for training Generative Adversarial Networks (GANs) on statistical samples. Some variants, such as multimodal GANs [26] and GMM GANs [27–29], have been developed to address the increased variability in GANs. Convolutional networks, such as StyleGAN [30] and BigGAN [31], are utilized for image data or even models in higher dimensional RGB images. In this study, we chose to implement the vanilla GAN due to its simplified structure.

In summary, Generative Adversarial Networks (GANs) have been developed to address various limitations associated with other generative models [32].

- GANs can generate samples in parallel, thereby avoiding the need for runtime proportional to the dimensionality of the data. This is a distinct advantage over the autoregressive models.
- The design of the generator function in a GANs is relatively unconstrained. This flexibility contrasts with flow-based models, where the generator must be invertible, and the latent code (z) must match the dimensionality of the data samples.
- GANs do not require the imposition of variational bounds, and specific model families compatible with the GAN framework are already recognized as universal approximators. Consequently, GANs are asymptotically consistent.

However, it is important to note that GANs have also introduced a new challenge; their training involves finding the Nash equilibrium of a game, which is a more complex problem than optimizing a conventional objective function. Therefore, selecting the correct architecture is vital for the problem under consideration [32].

2.1.2. Flow Based Models

Real-valued non-volume-preserving models [33] are flow-based models that use latent distributions and convert them to data samples with invertible functions. Real NVP constitutes the basis for current state-of-the-art flow-based models, such as GLOW [34]. Latent variable models are likelihood-based trained models that can directly parameterize probability distributions. On the other hand, Real NVP attempts to learn the mapping from the latent domain z to the data domain with invertible transformation; hence, the invertibility property ensures the prevention of mode collapse. Therefore, they can capture the variability of the distribution. However, measuring sample diversity in generative models is a difficult problem and is currently intractable [35,36]. The NVP introduces a class of bijective functions that can be used to evaluate density estimation in a tractable manner. It utilizes the power of change of the variable formula

to compute the log-likelihood of continuous data. One caveat that differs from other generative models is that the use of invertible transformation flow models requires latent and data dimensions to be of the same degree. Second, the Jacobian determinant computation should be efficiently calculated and differentiable.

Real NVP model function can be expressed as:

$$p_{\mathbf{x}}(x) = p_{\mathbf{z}}(z) \left| \det \left(\frac{\partial g(z)}{\partial \mathbf{z}^T} \right) \right|^{-1}. \tag{2.2}$$

In this expression, $g(z)$ is the generative network that maps latent space z to data space x. Because g is bijective, the model can be trained directly using the likelihood using the change of variable formula. The problem is then to find a class of bijective functions that have an easily computable Jacobian. Real NVP paper proposes of a method for coming up with such function as can be seen in Figure 2.1.

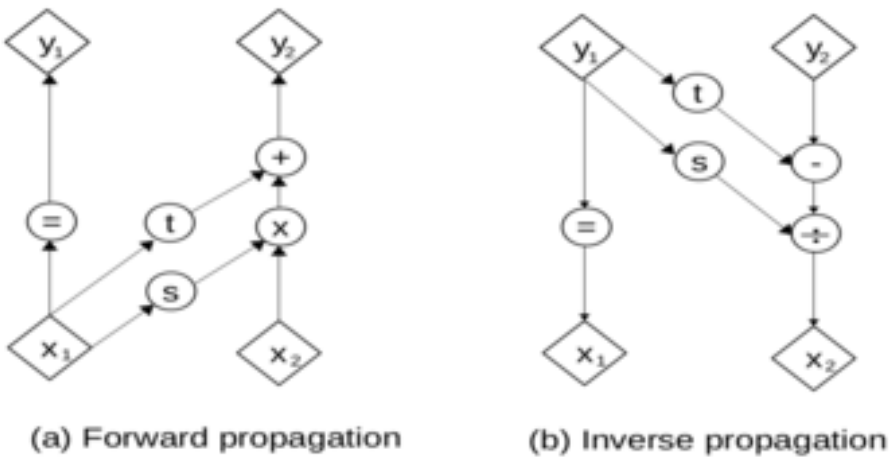

Figure 2.1. Computational graphs for forward and inverse propagation.

This figure can be expressed as

$$y_1 = x_1. \tag{2.3}$$

$$y_2 = x_2.e^{s(x_1)} + t(x_1). \tag{2.4}$$

In this equation x_1 and x_2 are the inputs of the networks and y_1 and y_2 are the outputs of the networks which will be the inputs for the next layers. This formulation

enables for easy calculation of determinant of the Jacobians. We can express this aforementioned formulation as

$$\frac{\partial y}{\partial \mathbf{x}^T} = \begin{bmatrix} \mathbf{I}_d & 0 \\ \frac{\partial y_{d:+1:D}}{\partial \mathbf{x_{1:d}}^T} & diag(exp[s(x_{1:d})] \end{bmatrix}. \tag{2.5}$$

In this equation, the Jacobian is independent of the choice of s() and t() therefore enables to capture complexity in these networks contrary to its simple nature in computation graph.

There are several architectures for the flow-based models. Models such as Integer Discrete Flows [37] were used for image modeling. In this study, we implemented RealNVP, which uses neural networks as scale and transition functions, while maintaining invertibility and tractability. We maintained RealNVP for all datasets and conducted all tests on the same architecture.

In summary, flow based models are characterized by the following attributes:

- The generation of samples can occur at a pace that is comparable to GANs, which allows for the efficient use of data.
- The latent space must have the same dimensionality as the data space, but it can be used for lossless compression since it provides an exact likelihood value.
- NVPs require a family of bijective functions for transformation, which can limit the choice of architecture.

2.2. Comparing Generative Models

A formal evaluation of the generative model architectures is of utmost importance. However, this task is not as straightforward as it may seem. The availability of an explicit probability distribution for the model depends on the architecture used, and in some cases, the underlying data distribution may not be present. For instance, in image or video generation tasks, the data manifold is high-dimensional and continuous, making it challenging to evaluate. In image-generation tasks, the performance of the

model is mostly evaluated by visually inspecting samples, and the task is to generate realistic samples. It is also crucial to capture the sample diversity and generalization properties of the model to generate diverse samples from a given distribution. Additionally, it is crucial to compare the same measures, such as in MNIST, which is an essential dataset in image generation tasks and consists of handwritten digits, which are grayscale, according to choice of the authors, some treat pixels as points in real vector space, others normalize it to 0-1 scale and some center it around the mean value of 0 and use this as a threshold for the pixel intensity [38]. Therefore, this might make the comparison harder to evaluate.

In evaluating generative models, determining a consistent and robust metric is crucial. We need a metric that can encompass the sample diversity, sample quality, and generalization capacity. Although no single metric exists that can fully meet these requirements [3], various metrics have been proposed for this purpose, each with its own strengths and weaknesses. One key consideration when comparing generative models is selecting the appropriate criteria to evaluate the models based on the data at hand. This differs greatly between domains; for instance, in image generation, FID and IS are used to compare and validate models; however, detection is mostly performed by visual inspection because divergence comparisons are difficult to conduct in multidimensional data, and ratio or distance metrics do not correctly represent the perception of generated data, such as blurry images. Additionally, likelihood-based comparisons may not accurately reflect human perception due to the informational value of background pixels being less important than other pixels, although they occur more frequently in the data. Another example of this phenomenon is the adversarial method, which has been proposed for image generation tasks and involves injecting a noise vector into image data that does not affect human perception but significantly reduces the discriminator's evaluation of the image [39]. Unfortunately, generative models can also be used to trade sample quality for spread using memorization. This behavior is mostly observed in GANs as mode collapse.Furthermore, it is also shown that most of the aforementioned metrics fail to capture the memorization property of the model training [6].

In addition to the aforementioned evaluation methods, certain studies have employed other methods to evaluate generative models, such as precision and recall metrics, statistical tests, and pretrained classifiers. Statistical tests have a longstanding history in determining whether two sets of samples originate from the same distribution, known as two-sample tests. These tests involve calculating a statistic from the data and comparing it to a predefined threshold. In the context of assessing implicit generative models, such as GANs, various statistics relying on classifiers [40] and Maximum Mean Discrepancy (MMD) [41] have been utilized.

However, it is important to note that, like other evaluation metrics for generative models, statistical tests have their own set of advantages and disadvantages. These tests tend to be computationally intensive and are not typically suitable for real-time monitoring of training progress [3]. Instead, they are more suitable for comparison with fully trained models. In contrast, the MMD estimate was used to train the MMDGAN [16] to replace the discriminator network.

Another approach for assessing generative models involves human evaluations. This entails presenting samples generated by the model alongside samples from the actual data distribution, and soliciting judgments from human assessors who compare the quality of these samples. Platforms, such as Amazon Mechanical Turk, are commonly used for this purpose. Human evaluation is a suitable metric, especially when the model's output is intended for artistic or human display purposes or when obtaining reliable automated metrics is challenging. However, human evaluation can be challenging to standardize, tough to automate, and entail significant costs or logistical complexities during setup.

A significant proportion of the aforementioned metrics concentrate on evaluating the quality and diversity of the generated samples but are inadequate in quantifying the overfitting of the model to the training data. In order to address this issue, a typical approach is to inspect the results visually. Specifically, a collection of samples is generated by the model, and for each sample, the K-nearest neighbors in the feature

space are identified using a pretrained classifier on the dataset. Although this method involves manual evaluation of samples, it offers a straightforward means of determining whether the model is merely memorizing the training data. Moreover, in order to measure the diversity of the generated samples, one approach is to approximate the range of the learned distribution by searching for similar samples within a large pool of samples. However, this method can be resource-intensive and often requires human judgement [42].

In this section, we present the aforementioned metrics in detail. KL divergence, which serves as the foundation of the GAN loss function, is utilized to measure the similarity between given probability distributions. The use of KL divergence is justified as it is an effective method for evaluating loss in cases where there are multiple feasible solutions to a given problem [32]. FID and IS are image-related metrics that require the use of a pretrained Inception Model for calculation. Finally, we introduce Occam's principle and MDL-based metrics, which evaluate the complexity of the model with respect to loss.

2.2.1. KL Divergence

Kullback-Leibler (KL) divergence, also known as relative entropy, is a measure of the dissimilarity between two probability distributions. It quantifies the information lost when one distribution is used to approximate another distribution. For two discrete probability distributions P and Q, Kullback-Leibler (KL) divergence is defined as:

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_i P(i) \log \left(\frac{P(i)}{Q(i)} \right). \tag{2.6}$$

The summation is switched to ingtegration for density functions. KL divergence has several properties that make it a useful tool in various fields such as information theory, machine learning, and statistics.

- Non-Negativity: The KL divergence is always non-negative, and it equals zero if and only if the two distributions are identical; that is, $P(i) = Q(i)$ for all i.

- Lack of Symmetry: KL divergence is not a true distance metric, because it is not symmetric. This property arises from the information gain perspective of KL divergence.
- Invariance: KL divergence is not invariant under changes in the probability scale. In other words, if P and Q are probability distributions, $D_{\mathrm{KL}}(P \parallel Q) \neq D_{\mathrm{KL}}(aP \parallel aQ)$ for any scalar $a \neq 1$.
- Additivity: For independent events, the KL divergence of the joint distribution factorizes into the sum of the individual KL divergences.

KL divergence has a wide range of applications in various fields, including statistical modeling, information retrieval, and generative models. It is commonly utilized as a loss function or measure of dissimilarity to compare probability distributions and evaluate the difference between model predictions and actual data. KL divergence provides a quantitative measure of disparity between probability distributions. However, it should be noted that KL divergence is not a distance, as it is asymmetric, and therefore cannot be used to measure the actual physical distance between two points. Additionally, calculating KL divergence can be challenging in high-dimensional data and is often impossible if the underlying distribution is not known. As a result, estimates of KL divergence are commonly used.

2.2.2. Frechet Inception Distance (FID) and Inception Score (IS)

The Frechet Inception Distance (FID) measures the similarity between the generated and real images based on their feature representations extracted from a pretrained Inception Network. It is calculated as the Wasserstein-2 distance between the multivariate Gaussian distributions formed by the means and covariances of the real and generated image features. The FID metric can be expressed as

$$\mathrm{FID}(P_{\mathrm{real}}, P_{\mathrm{gen}}) = \|\mu_{\mathrm{real}} - \mu_{\mathrm{gen}}\|^2 + \mathrm{Tr}(\Sigma_{\mathrm{real}} + \Sigma_{\mathrm{gen}} - 2(\Sigma_{\mathrm{real}}\Sigma_{\mathrm{gen}})^{1/2}), \tag{2.7}$$

where μ_{real} and μ_{gen} are the mean feature vectors, and Σ_{real} and Σ_{gen} are the covariance matrices of the real and generated image features, respectively. FID provides a quantitative measure of the quality and diversity of the generated images. A lower FID

value indicates a better similarity between the generated and real images. Therefore, the lower the FID score the better the model quality.

The Inception Score (IS) evaluates the quality of the generated images based on their diversity and how well they are classified by a pretrained Inception Network. It measures the quality and diversity of the generated samples by calculating the average entropy of class probabilities and their marginal entropy. The Inception score is expressed as

$$\mathrm{IS}(P_{\mathrm{gen}}) = \exp(\mathbb{E}_{x\sim P_{\mathrm{gen}}}[D_{\mathrm{KL}}(p(y|x)\|p(y))]). \tag{2.8}$$

In this expression $p(y|x)$ represents the class probabilities of the generated images and $p(y)$ is the marginal class distribution.A higher IS value indicates a better diversity and quality of the generated images.

Both FID and IS have been widely adopted in the evaluation of generative models in computer vision tasks. They provide valuable insights into the quality, diversity, and similarity of generated images compared with real images. Frechet Inception Distance (FID) and Inception Score (IS) are two important metrics for evaluating the performance of generative models in computer vision. They provide quantitative measures of the similarity, quality, and diversity of generated images, aiding the assessment and comparison of different generative models.

2.2.3. Minimum Description Length (MDL) and Overfitting

Occam's Razor, a principle formulated by William of Occam, advocates for the utilization of the most straightforward model that can account for a given dataset, among compatible hypotheses, as it provides a more aesthetically pleasing and empirical solution [43]. This principle is based on the notion that simpler models tend to provide more precise predictions that effectively capture the relevant data space around observed data points. Moreover, the utilization of complex models, which are dispersed thinly around the data space, increases the risk of overfitting. The application of this principle also aids practitioners in evaluating the likelihood of overfitting in

model selection. It additionally tends to exhibit a preference for models characterized by a greater diversity of samples.

Occam's Razor is not a rigid rule, but rather a heuristic, practical guideline for making decisions when faced with uncertainty. While simplicity is valued, it is important to balance it with the explanatory power of the hypothesis. Sometimes, a more complex model is required to account for all nuances of a phenomenon. Therefore, the explanatory power of a model and its related errors should be included in the process of model selection.

The Minimum Description Length (MDL) Principle is a relatively recent method for inductive inference that provides a generic solutions to the model selection problem. MDL is based on the following insights that any regularity in the data can be used to compress the data, i.e. to describe it using fewer symbols than the number required to describe the data.

The Neural Network Divergence metric is a key measurement used in the evaluation of neural network-based models, particularly in the realm of generative models. This metric assesses the dissimilarity or discrepancy between the data distribution and the distribution of generated samples produced by the neural network. It provides valuable insights into how well the model has learned to capture the underlying patterns and structures within the data. Lower divergence values indicate that the generated samples closely resemble the real data distribution, reflecting a higher quality and more faithful representation by the model. This metric plays a pivotal role in optimizing and fine-tuning neural-network architectures to enhance their generative capabilities and overall performance. However, this metric calculation requires a seperate neural network to be trained to estimate the divergence behaviour between generated samples and training samples. Moreover, it incorporates an additional test set for comparative analysis with the training set to comprehensively assess the potential presence of biases and overfitting in the model's performance.

In study [6], it is argued that divergence-based metrics fail to capture the overfitting phenomenon because we fail to capture the distribution, the likelihood is mostly intractable, and the measure is estimated via sampling, and therefore lacks an overfitting measure. To overcome this problem, they introduced a fix to the NND metric [44].

The Minimum Description Length (MDL) principle is a fundamental tool in machine learning that is concerned with the problem of overfitting. It minimizes the difference between the model complexity and the amount of information required to explain the training data. The MDL principle has been applied to the problem of architecture optimization in artificial neural networks in a heuristic manner, as evidenced by its successful application in [45]. This approach can be used to guide the selection of model architecture.

3. METHODOLOGY

Deep generative models have gained significant attention and play a pivotal role in various machine-learning applications, including computer vision and natural language processing. In this study, we performed a comprehensive analysis of deep generative models, focusing on their architecture and complexity. The results and insights obtained from this study shed light on several critical aspects of these models, which can guide future developments and applications in the field.

In this chapter, we will describe the problem that our study and proposed MDL-based metric aim to address. Following this, we will delve into the details of the models being compared, including their hyperparameters. We will then discuss the datasets used for this study, finalizing this chapter with a description of the metrics used to compare models and introduce our proposed MDL-based metric.

3.1. Motivation and Problem Definition

Generative model architectures have varying assumptions and, consequently, differ in their performance when applied to different datasets. Basic models, such as probabilistic PCA [46], are easier to employ for capturing linear dependencies among data distributions. Additionally, methods like expectation maximization [47] can be more practical to utilize for capturing data distributions like GMM with very few parameters and computing resources. However, deep generative models have been used to capture more complex dependencies. GANs are commonly used for generating realistic images, videos, and translating images. Flow-based models can capture both mixed and multivariate distributions. Thus, it is essential to select the appropriate architecture and model based on the specific data at hand. Training deep generative models requires significant computational resources and expertise, and there is currently a lack of consistent and robust performance metrics for these models. Furthermore, there are limited comparisons that evaluate models under equal computational conditions and

compare their performance. Therefore, we aim this study to form a basis for model complexity and data dimensionality comparison.

We categorize our contributions into research and practical implications. The research implications of this study are:

- We provide background on Generative Models and metrics for evaluating performance of Generative Models.
- We propose a MDL-based metric that encodes model quality with respect to model complexity.
- We provide the implementation of the proposed MDL-based metric for the evaluation and comparison of two types of generative models, namely GANs and RealNVPs based on their performance on their architecture and complexity with respect to variability and dimensionality of different datasets.

The practical implications of this study are as follows:

- We provide a method to help practitioners choose between different architectures and share comparisons of generative models based on the proposed MDL-based metric and KL divergences.
- We provide a method to guide practitioners to select generative models and model architectures taking into account variability and dimensionality of their datasets.
- We provide estimated KL results for performance of two types of generative models with different datasets.

3.2. Models

In this study, we utilized flow-based architectures (NVP) and Vanilla GANs to compare and contrast the impact of architecture and complexity on generative models, similar to the approach taken for VAE and GANs [40]. We searched for hyperparameters, such as learning rate and model architecture, based on the literature and used

them for the GANs and flow-based models for similar data-learning tasks. We also implemented different learning rate ratios for GAN training.

We experiment a set of generator learning rates in the interval 0.05 and 0.0001. For each value of the generator learning rate we varied the discriminator learning rate. All possible values , which are the cartesian product of a set of generator and discriminator learning rates, are experimented for each dataset. The model complexity is varied by experimenting different depths and for each depth varying the number of nodes.

The learning rate of the Real NVP model was searched within the range of 0.01 and 0.0001. Furthermore, the complexity of the NVP model was evaluated by varying the width of the coupling layer and the number of coupling layers, which subsequently impacted the number of trainable parameters for each dataset.

3.3. Datasets

We conducted an evaluation of the generative model's performance with respect to dimensionality and variability of datasets utilizing synthetic generated data distributions as well as MNIST image data. For synthetic data distributions dimensionality is experimented with only two dimensional and four dimensional datasets. This limitation stems from computational burden of the experiments. We theorized that the aforementioned model architectures can be categorized based on their performance in handling dimensionality and variability of the datasets. To test the capturability of the data, we utilized synthetic multivariate, mixed, and probabilistic distributions.

Generated datasets are formed by varying variability and noiseness measures. Ten distinct 2D datasets were employed, including Gaussian mixture models,clustered data, synthetic nonlinear data, and t-distribution. The datasets were selected based on similar studies [33, 48].

Our datasets consist of the following:

- Gaussian Mixture Model (GMM) Dataset: This dataset encompasses two multivariate Gaussians with mean vectors located at [-1,-1] and [1,0]. The respective covariance matrices are [0.05, 0], [0, 0.15], and [0.1, 0.2]. Additionally, an alternative version with increased variance is included, achieved by multiplying the covariance matrix by 5. The mixing probability values for the two Gaussians are 0.4 and 0.6.
- Blobs Dataset: Comprising three Isotropic Gaussians, this dataset is bounded within the range of [-10,-10] and [10,10]. An augmented more variable version is available, featuring Gaussian noise with a standard deviation of 0.5 to introduce higher variability.
- No Structure Dataset: This dataset is uniformly distributed within the rectangular area defined by the points [0,0] and [1,1].
- Noisy Circles Dataset: Characterized by large circles with an embedded smaller circle in a two-dimensional space, this dataset is further augmented with Gaussian noise featuring a standard deviation of 0.5 for increased variability.
- Noisy Moons Dataset: Comprising two interleaving half circles, this dataset is enhanced with Gaussian noise of standard deviation 0.5, providing increased variability in the dataset.
- T-Distribution Dataset: Featuring two multivariate T-distributions, this dataset is bounded within the range of [200,200] and [-200,-200].

Scipy python library was used to generate blobs,circles and moons datasets. We implemented Gaussian mixtures and t distribution in PyTorch.

The 2D datasets used in our study are presented in Figure 3.1 and Table 3.1 provides a summary of the datasets and their contributing variance factor, as well as the scale of each contributing factor.

Table 3.1. Characteristics of 2D datasets.

Dataset	Variance Factor	Variance Ratio
GMM low variance	Covariance	1
GMM high variance	Covariance	5
blobs	Gaussian Noise	1
blobs high std	Gaussian Noise	5
no structure	No Noise	0
noisy circles	Gaussian Noise	1
noisy circles high noise	Gaussian Noise	5
noisy moons	Gaussian Noise	1
noisy moons high noise	Gaussian Noise	5
t mixture	Covariance	1

Next, we expanded these 2D datasets to 4D by using nonlinear projections. Nonlinear projections were made by taking the inner product of the data vector with itself and the matrix product with a positive definite matrix as xAx^T where A is a positive definite matrix and x is the 2D data vector. Positive definite matrix A is generated by taking the inner product of any 2x2 matrix with a covariance matrix to make it strictly positive definite. This ensures positive definiteness of the transformation matrix A. The fourth dimension is added using the exponential of the square root of the inner product which is the third dimension as explained by prior. However, the nonlinear transformations resulted in increased variability of the t distribution, and we were unable to include it in the 4D dataset due to numerical instability. We increased the degree of freedom to reduce variability, but this still produced numerical instability for KL estimation. Further increasing the degree of freedom would lessen the characteristic of the t distribution.

We created a training dataset consisting of 10,000 samples for each of the ten 2D and 9 4D datasets. Additionally, we utilized a test dataset comprising of 2,000 samples for each of these 2D and 4D datasets respectively.

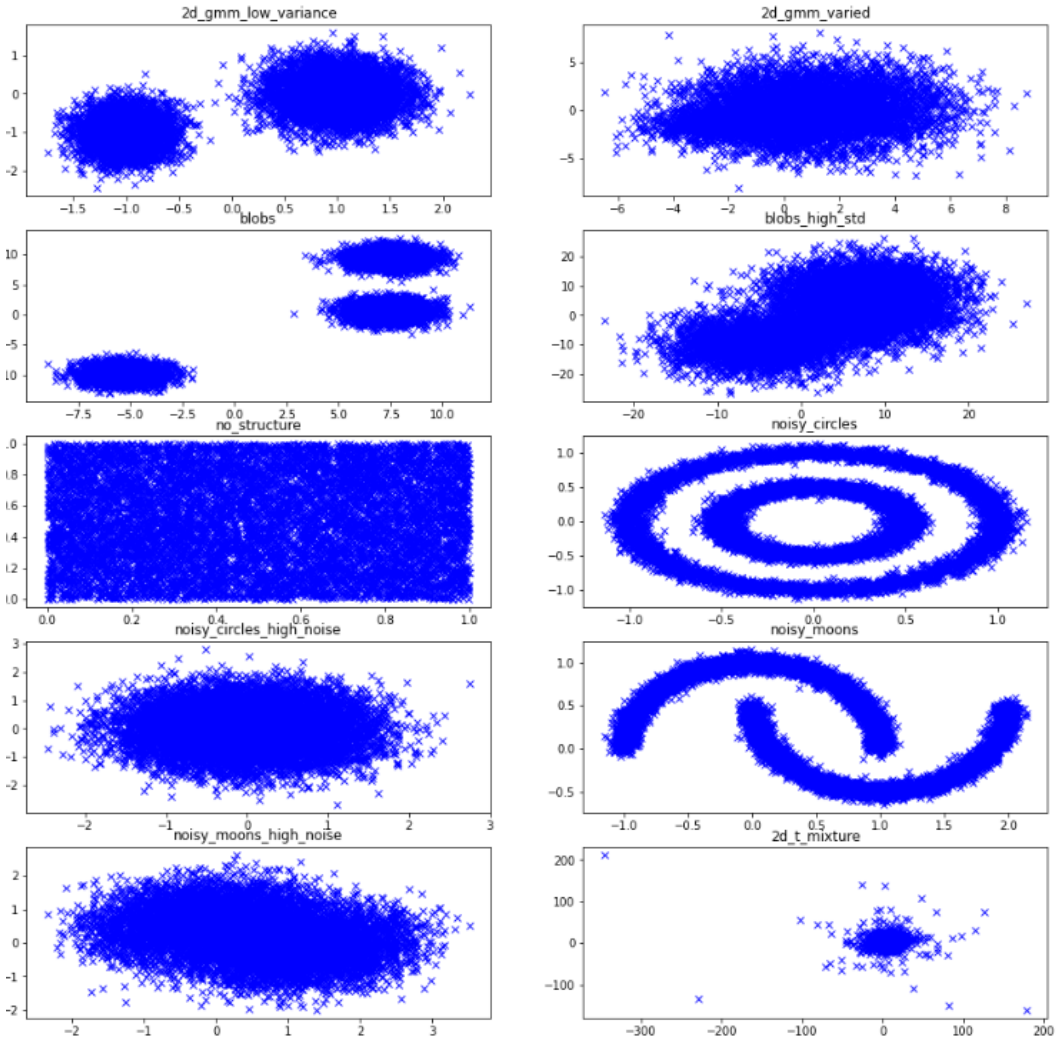

Figure 3.1. Our dataset.

We then added the MNIST dataset to compare the model performance in the image domain. The MNIST dataset comprises 28x28 grayscale handwritten digits, with 60,000 training samples and 10,000 test samples. To ensure consistency with other datasets, we normalized the pixel values, ranging from 0 to 255, to a range of -1 to 1. Additionally to evaluate the effect of dimensionality, we utilized a cropped variant of this dataset, 16x16, similar to the previous generated datasets. Therefore we conducted 2 tests on MNIST dataset, one with 28x28 images and the other with 16x16 images corresponding to high and low dimensionality respectively.

3.4. Metrics

We propose that Minimum Description Length (MDL)-based metric can be considered as a suitable tool for evaluating the intricacy of generative models in relation to the dimensionality of the data. This metric may prove to be a valuable asset to practitioners when selecting the most efficient architecture and complexity level for datasets of diverse dimensions.

We compared the performance of the model with the empirical KL implementation from [49]. The KL divergence between two distributions is defined as

$$D_{\mathrm{KL}}(P\|Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) \, dx, \tag{3.1}$$

in which probability distributions are defined as p(x) and q(x) respectively. Moreover, a kNN proposed estimator for KL is defined in [50] as

$$\hat{D_{\mathrm{KL}}}(P\|Q) = \frac{d}{N} \sum_{i} \log \frac{v_i}{\epsilon_i} + \log \frac{M}{N-1}, \tag{3.2}$$

in which ϵ_i is the distance between x_i and it's kth nearest neighbour in its own dataset, v_i is the distance between x_i and it's kth nearest neighbour in comparison dataset, d is the dimension and N and M are the sample counts of datasets respectively. KL is a ratio based metric that captures two distributions resemblence but due to being asymmetric it is not referred as a distance. To eliminate the effects of changing data dimensionalities, we utilized a ratio-based divergence estimator.

We used the MDL-based metric implementation for the ANNs based on [51] this metric is defined as

$$L = nD \log_2 E + \log_2 P + 0.5(P) \log_2 n, \tag{3.3}$$

in which E is the estimated KL divergence between the model samples and data samples, n is the sample size, D is the dimension and P is the parameter count. The original implementation had number of neurons and connections respectively which is switched to parameter count in our implementation. We used the number of parameters as the complexity measure and estimated KL divergence as the error rate for the Deep Generative Models.

It is important to note that both the MDL-based metric and the KL score are measured on a scale such that a lower value indicates a better outcome. Specifically, the MDL-based metric has a range from negative infinity to positive infinity due to its logarithmic definition, while the KL score is always positive and may occasionally result in minimal negative values due to numerical instability. The implementation we used is an estimate of KL therefore sometimes the estimate resulted in negative values which is outside of the range of KL which always non-negative. This was seen in too close divergence scores between generated and data samples. We took the absolute value for such cases to calculate MDL-based metric scores.

4. EXPERIMENTS AND RESULTS

We utilized synthetic generated data distributions and MNIST image data to evaluate the performance of the generative model with respect to dimensionality and variability by employing empirical KL divergence and MDL-based metrics [49, 50]. Our architecture is based on state-of-the-art architectures commonly used to train toy models.

We utilized PyTorch as our development environment, which is commonly utilized for academic research on deep neural networks. The models were trained on NVIDIA 3060 TI GPUs. The model architecture was trained and evaluated separately for each dataset, and compared against other architectures to ensure optimal performance.

In this chapter, we will describe the experimental setup that was utilized in our research. Following this, we will present the results of our investigations, commencing with the results pertaining to the 2D datasets and subsequently proceeding to the 4D datasets. Finally, we will share the results obtained from both the MNIST datasets and lastly provide an analysis of our findings.

4.1. Experimental Setup

We began our training with 2D datasets and subsequently proceeded to 4D datasets, eventually moving on to the MNIST dataset. Empirical KL results and MDL-based metrics are presented in the tables, and visual plots for the data and model-generated data are provided in the plots. We chose not to include plots for the MNIST data due to its higher dimensionality. We present only test scores for all datasets.

The hyperparameters and model architectures used for training GANs in our study are presented in Tables 4.1 and 4.2. 15 generator to discriminator learning rate

ratios are experimented which are shown in Table 4.1. Three generator learning rate levels are experimented(0.01,0.001,0.0001). For each value of the learning rate level all possible learning rate ratios are test resulting in 45 different training setting for a given architecture for each dataset. For each learning rate level we first set the generator learning rate and create the different discriminator learning rates utilizing the learning rate ratio. For instance, if the learning rate level is 0.01 and generator to discriminator learning rate ratio is 2:3 the generator learning rate is 2 times 0.01 which equals 0.02 and discriminator learning rate is 3 times 0.01 which equal 0.03.

The model complexity in the GANs was also tested for different layer widths and depths as can be seen in Table 4.2, which changed the number of trainable parameters. In Table 4.2 Gen. layers presents layer sizes used in generator network whereas disc. layers presents the sizes utilized in the discriminator network. There were 11 different architectural settings in GANs. The total number of settings tested are cartesian product of all possible hyperparameter and architectural settings which resulted in 495 different training settings overall for each dataset.

The learning rate of the Real NVP model takes three distinct values (0.01,0.001 and 0.0001). Furthermore, the complexity of the NVP model was evaluated by varying the width of the coupling layer and the number of coupling layers, which subsequently impacted the number of trainable parameters. Similarly to GANs, all hyperparameters were tested using a cartesian product of all possible settings, resulting in the examination of 24 settings for each dataset for NVP models. The architectural settings are shown in Table 4.3.

The effect of learning rate on KL and MDL metric scores are not found to be significant as demonstrated by the Table A.1. Therefore we present the best models. We presented the results for best(lowest MDL score) models for each dataset and model architecture but all of our models and datasets can be accessed from the link https://gitlab.com/aemreg/dgm-analysis/-/tree/main. We presented the results of lowest five complexity architectures for 2D and 4D datasets and moved on to higher

complexity of highest five complexities for MNIST datasets.We presented the models that are trained for 5000 epochs. In total we have conducted 10.899 trainings, GANs 4950 for 2D trainings which comes from 495 tested architectures for 10 datasets, 240 trainings for NVPs for 2D datasets which comes from 24 settings tested against 10 datasets, 4D trainings are 4455 trainings for GANs which is 495 tested settings against 9 datasets.990 MNIST trainings were conducted for GANs and 48 were conducted for RealNVP. Each training setting takes approximately 1 hour.

Table 4.1. Generator to discriminator learning rate ratios for GANs.

Generator to Discriminator Learning Rate Ratio
1:1
1:2
1:3
1:4
1:5
2:2
2:3
2:4
2:5
3:3
3:4
3:5
4:4
4:5
5:5

All of the learning rate ratios mention in the Table 4.1 were implemented using a recursive loop and tested for every learning rate level. The results of these different learning rate ratios and their relative effects on KL divergence and MDL based metric are presented for low variability GMM dataset in Table A.1

Table 4.2. Architectures for GANs.

Gen. Layers	**Disc. Layers**	**Network Depth**	**Parameter Size**
7,13	11,29	4	955
14,26	22,58	4	3547
35,65	55,145	4	21163
70,130	110,290	4	83323
105,165	165,435	4	180153
256,512,1024	1024,512,256	6	2631171
3,5,7,13	5,7,11,29	8	1292
6,10,14,26	10,14,22,58	8	4869
15,25,35,65	25,35,55,145	8	29328
30,50,70,130	50,70,110,290	8	115853
45,75,105,195	75,105,165,435	8	259578

Table 4.3. Architecture for NVP.

Model Width	**Coupling Layer**	**Parameter Size**
2	4	156
5	8	1224
10	16	8048
20	32	57696
40	64	435392
50	80	840240
60	96	1440288
70	112	2273936

4.2. Results for 2D Data

We conducted experiments using both GANs and NVP models to evaluate their performance in capturing the dimensionality and variability of 2D datasets. The results, presented in Figures 4.1 and 4.2 comes from the best(lowest) MDL score having models

with respect to all architectures and all hyperparameters. They indicate that NVP models are better suited to capturing the diversity of the data space, as represented by the red dots, while GANs are more effective in capturing the mode and centroids of the data. However, when we assessed the models using the MDL-based metric, we found that although the higher complexity NVP models resulted in lower KL estimate scores (as shown in Tables 4.4 and 4.6), they exhibited higher MDL-based metric scores compared to lower sized models (as seen in Tables 4.5, 4.7). This implies that low complexity models are preffered for 2D datasets in general.

Table 4.4. 2D KL results for GANs.

	Number of Parameters				
Dataset	**1289**	**4863**	**29313**	**115823**	**259533**
GMM low variance	4.8075	0.3698	1.3443	6.2880	3.0758
GMM high variance	3.5409	0.1776	0.6500	0.3918	0.4178
blobs	8.9693	7.0576	8.9776	8.9549	8.8740
blobs high std	4.6665	0.4817	0.3997	0.4043	8.6174
no structure	4.8121	3.2452	5.9935	4.6154	9.4772
noisy circles	2.3288	1.3612	4.3552	5.6686	4.7792
noisy circles high noise	4.5013	3.5925	1.5086	2.6296	2.1970
noisy moons	3.9605	5.1847	6.7564	6.0880	5.7235
noisy moons high noise	4.2442	0.8125	1.0895	0.4090	3.9754
t mixture	1.6346	0.5941	2.7224	0.7070	3.3497

The lower parameter sizes result in lower MDL metric scores for 2D datasets, but different settings have the highest scores for specific datasets. An investigation into MDL metric scores for 2D datasets revealed that lower parameter sizes correspond to lower MDL scores. However, a detailed analysis showed that optimal performance is achieved with specific parameter configurations for each dataset. This behavior is further investigated through an inspection of generated samples obtained from models possessing the most favorable Minimum Description Length (MDL) metric scores.

Table 4.5. 2D MDL metric results for GANs.

	Number of Parameters				
Dataset	**1289**	**4863**	**29313**	**115823**	**259533**
GMM low variance	16155	20966	162524	645836	1429739
GMM high variance	14390	16736	158331	629818	1418219
blobs	19754	39374	173482	647877	1435854
blobs high std	15983	22493	155524	630000	1435685
no structure	16160	33501	171150	644052	1436233
noisy circles	11972	28488	169308	645238	1432283
noisy circles high noise	15775	34088	163189	640805	1427798
noisy moons	15036	37733	166142	645650	1431220
noisy moons high noise	15436	25510	161312	630067	1431220
t mixture	9929	23703	166596	633225	1430232

Table 4.6. 2D KL results for NVP.

	Number of Parameters				
Dataset	**156**	**1224**	**8048**	**57696**	**435392**
GMM low variance	6.7264	6.9698	7.0071	7.0022	6.9939
GMM high variance	3.2831	3.0676	2.7542	3.1593	2.9532
blobs	6.4655	2.3169	0.3724	0.0080	0.0391
blobs high std	4.7558	0.2581	0.0513	0.0322	0.0237
no structure	0.6475	0.1404	0.0270	0.1020	0.0268
noisy circles	0.4198	0.3673	0.1452	0.0855	0.0896
noisy circles high noise	0.0093	0.0017	0.0038	0.0778	0.0224
noisy moons	1.2629	0.8420	0.1756	0.1543	0.1513
noisy moons high noise	0.0008	0.0215	0.0279	0.0138	0.0241
t mixture	0.2290	0.0189	0.0111	0.0067	0.0076

Table 4.7. 2D MDL metric results for NVPs.

	Number of Parameters				
Dataset	**156**	**1224**	**8048**	**57696**	**435392**
GMM low variance	56040	64164	109659	439496	2948822
GMM high variance	35345	40484	82715	416532	2923945
blobs	54899	32893	24977	244127	2799153
blobs high std	46037	-30935	-32204	284186	2784664
no structure	-11497	-48499	-50683	317463	2788218
noisy circles	-23999	-20760	-2192	312375	2823088
noisy circles high noise	-134015	-176655	-44223	222723	2783091
noisy moons	7777	-18584	3297	329414	2838211
noisy moons high noise	-206117	-102689	-50124	280113	2785225
t mixture	-41489	-16168	-76268	238731	2751885

An examination of the MDL metric scores for 2D datasets disclosed that smaller parameter sizes lead to lower MDL scores which is intuitive. However, it was discovered as mentioned before that the best performance for each dataset is attained through different architecture settings, rather than just lower sizes as demonstrated in Table 4.7. These findings emphasize the importance of carefully tailoring model settings.

This comprehensive analysis uncovered that, in comparison to Generative Adversarial Networks (GANs), Real-valued Non-Volume-Preserving (NVP) models exhibit a superior capacity to capture a greater amount of information and variability across a majority of datasets, a characteristic attributed to their inherent structure; particularly noteworthy is the observation that distributions with increased diversity, such as the t-distribution, necessitate a higher number of parameters for accurate encoding. These insights emphasize the intricate interplay between dataset characteristics, model structures, and parameter sizes, the role of the underlying distributional properties in determining the optimal model complexity for effective data encoding and representation.

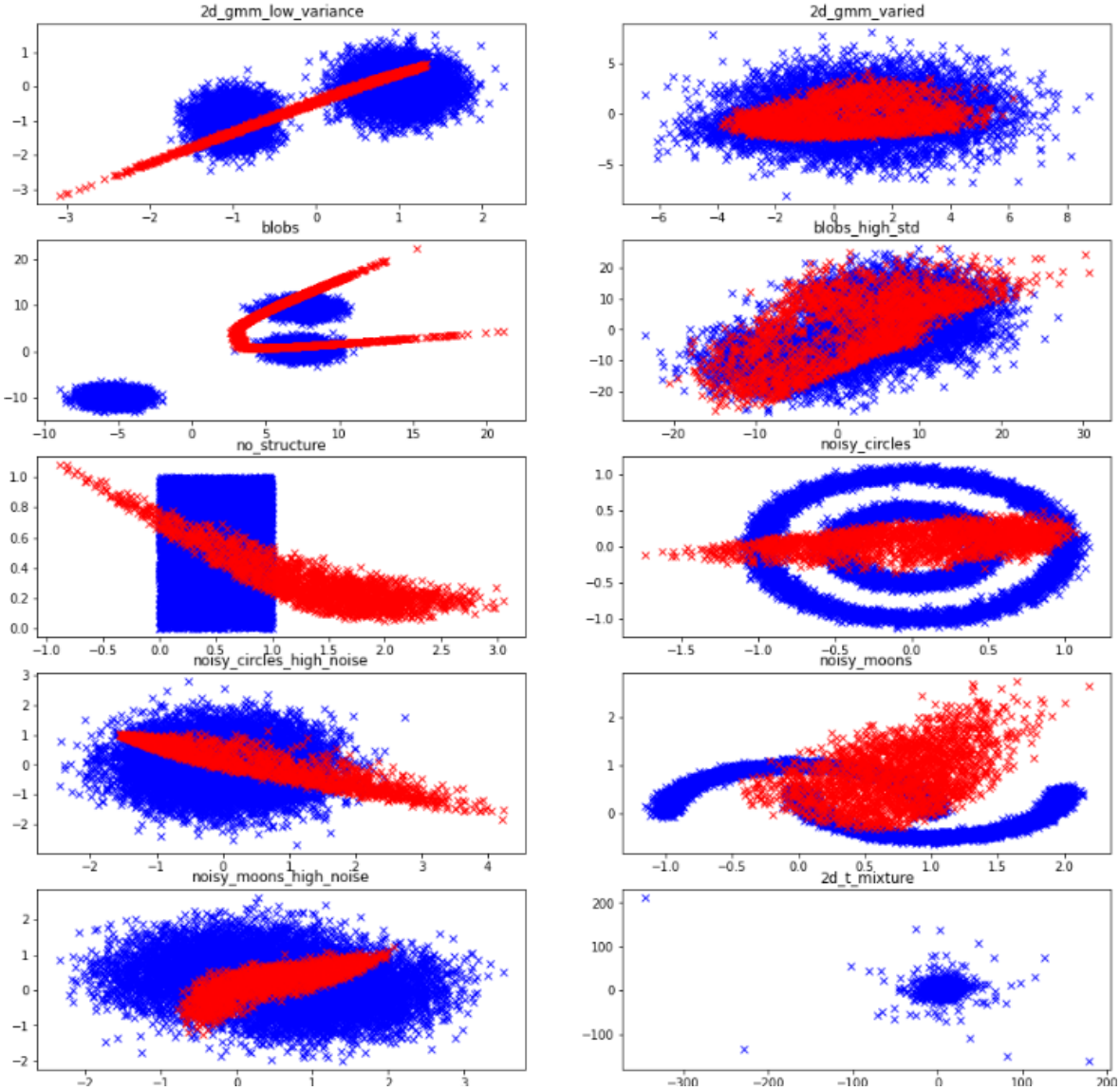

Figure 4.1. Best MDL 2D GAN model compared with test data. Red dots syntetic data, blue dots test data.

As depicted in Figure 4.1, GAN models tend to generate synthetic data that is predominantly situated around the centroids of the distributions, and oftentimes fail to recognize sparsity. Conversely, as illustrated in Figure 4.2, NVP models are more capable of accurately capturing diverse data compared to GANs. This outcome stems from the enforced bijective behavior ingrained within their training process, a distinctive characteristic that ensures a one-to-one mapping between input and output, thereby contributing to the model's heightened capability and effectiveness in capturing diverse data distributions.

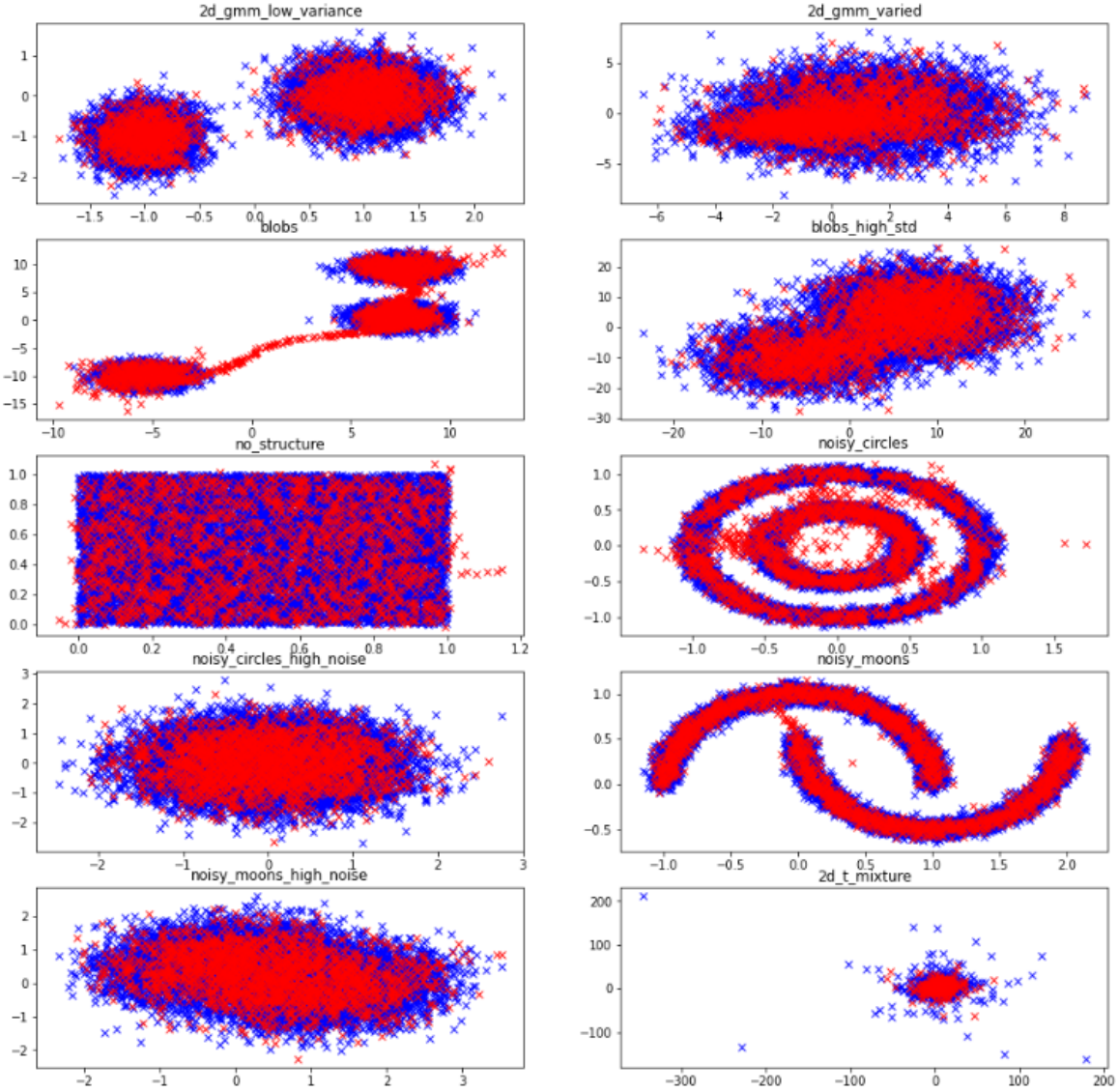

Figure 4.2. Best MDL 2D NVP model compared with test data. Red dots syntetic data, blue dots test data.

Our findings can be summarized as:

- NVP models outperform GANs on 7 out of 10 of our 2D datasets.
- Higher variability datasets like t-distribution requires more parameter complexity compared to other datasets.
- GANs have similar performance on most of the datasets in terms of MDL metric scores.

- NVP models have negative MDL scores in 7 datasets, which is a significantly lower value compared to GANs.
- GAN models are unable to capture the sparsity of the data, as evidenced by the visual inspection of generated samples.

4.3. Results for 4D Data

In our experimentation with the 4D nonlinearly transformed data, the settings were similar to 2D. We evaluated the KL divergence and MDL scores of the two models against different parameter sizes, which are presented in Tables 4.8, 4.9, 4.10, 4.11. We also visually inspected the sample diversity in Figures 4.3 and 4.4. These figures are also from best(lowest) MDL score models. The figures present first two dimensions for being able to compare performance with respect to 2D datasets, the other two dimensions are computed from these first two dimensions as explained in Chapter 3.

The increased dimensionality resulted in NVP and GAN models' divergence scores becoming closer. Nevertheless, the smaller parameter size of the NVP enabled it to encapsulate more variability than the GANs, leading to better MDL metric scores. Conversely, their performance in comparison to 2D datasets has not been comparable. Both models exhibited higher MDL scores due to the incorporation of increased variability and nonlinearity.

A notable observation has been made in the Generative Adversarial Networks (GANs): as the parameter count of a GAN rises, there is a corresponding decrease in the divergence between the generated data and the actual data. Although this behaviour is intuitive, it is crucial to note that this phenomenon does not inherently reflect a similar enhancement in the performance of Real-valued Non-Volume-Preserving (NVP) models, as GANs increased performance can be affected by the intricacies of their potentially unstable training process. Therefore, it can be asserted that Non Volume Preserving (NVP) models derive greater advantages from an augmentation in parameter count in contrast to Generative Adversarial Networks (GANs).

Table 4.8. 4D KL results for GAN.

	Number of Parameters				
Dataset	**1310**	**4903**	**29410**	**116015**	**259820**
GMM low variance	14.3060	14.4919	12.4332	11.6702	14.1737
GMM high variance	14.6079	12.8326	10.7207	10.7393	11.1305
blobs	27.1171	27.1160	27.1167	27.1165	27.1154
blobs high std	22.2728	22.2177	21.7148	21.5520	21.3928
no structure	20.0396	16.2933	14.0117	17.0768	14.2613
noisy circles	17.4766	14.0204	14.2782	13.8085	14.2798
noisy circles high noise	14.7359	12.4069	11.1141	10.1064	13.5165
noisy moons	19.7922	16.9080	16.9080	13.6204	13.2963
noisy moons high noise	16.0164	15.3354	13.1021	9.5474	9.9194

Table 4.9. 4D MDL metric results for GAN.

	Number of Parameters				
Dataset	**1310**	**4903**	**29410**	**116015**	**259820**
GMM low variance	37901	57752	190355	664472	1455184
GMM high variance	38142	56348	188645	663513	1452394
blobs	45281	64983	171930	674203	1462671
blobs high std	43010	62683	196791	671552	1459935
no structure	41791	59104	191735	668866	1455255
noisy circles	40211	57370	191952	666414	1455270
noisy circles high noise	38243	55959	189061	662812	1454636
noisy moons	41647	59531	192815	666256	1454446
noisy moons high noise	39204	58405	190960	662155	1451065

It has been previously noted that although the increase in number of parameters in GANS results in lower divergence scores, the magnitude of decrease is not significant enough to offset the magnitude of increase in parameters, leading to higher MDL metric scores.

Table 4.10. 4D KL results for NVP.

	Number of Parameters				
Dataset	**196**	**1400**	**8720**	**60320**	**445760**
2D GMM low variance	11.8100	7.8933	2.7143	2.6835	2.7622
2D GMM high variance	10.4531	8.9180	5.4641	3.8307	2.5694
blobs	27.0959	27.0983	27.0882	19.9618	4.6368
blobs high std	22.0769	22.1734	21.9028	11.8410	7.8969
no structure	9.1541	4.1409	2.4330	2.6361	5.4983
noisy circles	10.4411	7.4726	3.4346	4.0471	4.5822
noisy circles high noise	7.8454	7.2402	2.7887	1.7190	1.7736
noisy moons	11.8752	7.4966	3.6697	3.8606	12.2058
noisy moons high noise	8.1899	5.3561	2.8291	1.8407	2.3583

Table 4.11. 4D MDL metric results for NVP.

	Number of Parameters				
Dataset	**196**	**1400**	**8720**	**60320**	**445760**
2D GMM low variance	29577	31531	59348	342136	2455799
2D GMM high variance	28169	32940	67423	346244	2454964
blobs	39162	45767	85900	365297	2461777
blobs high std	36797	43452	83448	359269	2467923
no structure	26637	24086	58085	341931	2463744
noisy circles	28155	30899	62065	346879	2461641
noisy circles high noise	24857	30534	59660	336996	2450686
noisy moons	29641	30936	62829	346334	2472948
noisy moons high noise	25353	27056	59826	337785	2453974

It has been observed that as the dimensionality and variability of the data increased, the KL divergence and MDL metric scores of NVP models also increased. However, the KL divergence scores of NVP models showed a greater loss compared to GANs, with an increase in the number of parameters.

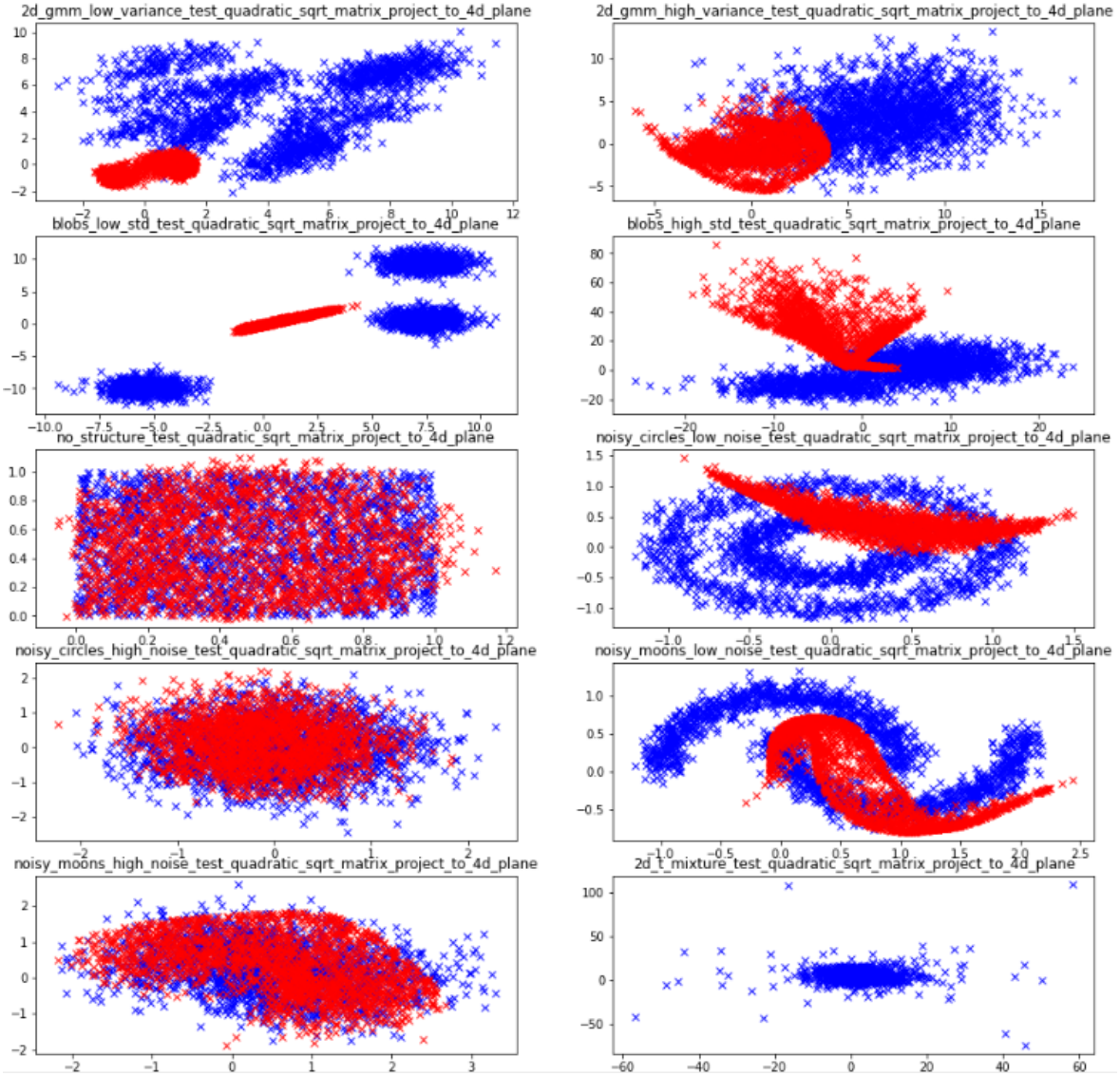

Figure 4.3. Trained 4D GAN model compared with test data. Red dots syntetic data, blue dots training data.

As depicted in Figure 4.3, it is evident that GAN models are incapable of accurately capturing sparsity in the majority of datasets. Conversely, as shown in Figure 4.4, NVP models are more proficient at capturing diversity in the noisy circles and noisy moons datasets. It can be observed that, as the dimensionality increased, the t-distribution exhibited a greater degree of variability, leading to the failure of both models to accurately capture the distribution. Therefore t-distribution was excluded from the comparative analysis, as previously expounded, and is absent from the result tables presented.

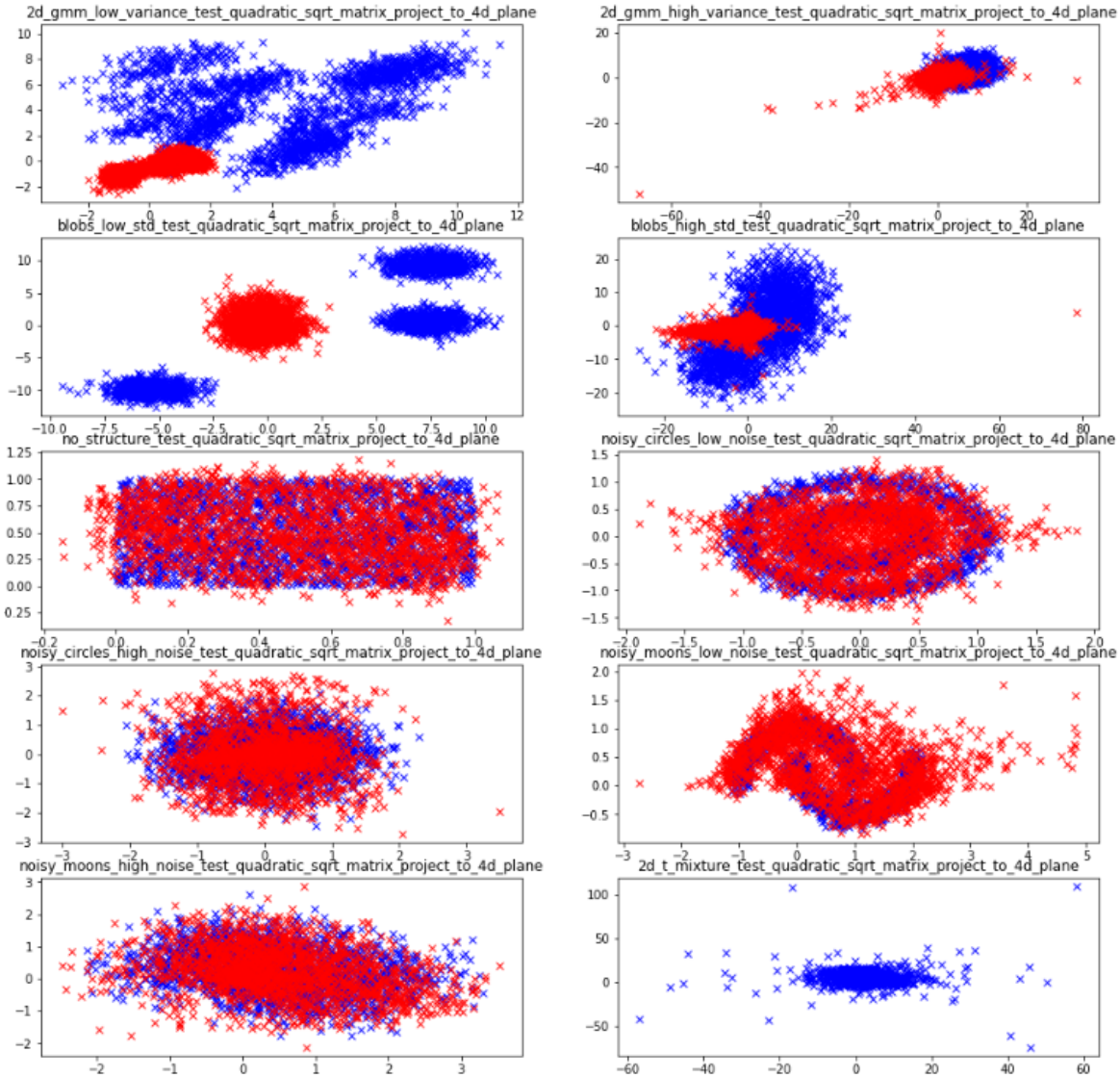

Figure 4.4. Trained 4D NVP model compared with test data. Red dots syntetic data, blue dots training data.

Our findings for 4D datasets can be summarized as:

- NVP models outperformed GANs on all ten of our 4D datasets, with the notable exception of performance degradation compared to 2D datasets.
- Both models MDL scores get closer to each other especially for the first two lowest parameter architectures, as well as their overall performance.
- GANs still have similar performance on most of the datasets in terms of MDL metric scores.

- GAN models continued to struggle with capturing the sparsity of the data as observed in the generated samples.
- Increased dimensionality and variability resulted in increased KL divergence and MDL metrics scores for both models.

4.4. Results for MNIST

In our experimentation with the MNIST, the settings were similar to prior datasets. We evaluated the KL divergence and MDL scores of the two models against different parameter sizes, which are presented in Tables 4.12, 4.13 and Tables 4.14, 4.15.

Table 4.12. MNIST KL results for GAN.

	Number of Parameters				
Dataset	**9943**	**41632**	**140207**	**295982**	**2953745**
MNIST16	346.7324	335.5670	319.8336	364.4218	247.3807
MNIST28	841.3870	832.7110	730.7866	617.1454	619.1794

Table 4.13. MNIST MDL results for GAN.

	Number of Parameters				
Dataset	**9943**	**41632**	**140207**	**295982**	**2953745**
MNIST16	129681668	129207842	128926039	133054417	141147971
MNIST28	456516815	457572303	4575723034	437487883	459700954

An intiutive observation emerged, indicating that with the escalation of data dimensionality, there was a proclivity towards favoring larger models in both Real-valued Non-Volume-Preserving (NVP) and Generative Adversarial Networks (GANs); nevertheless, the simultaneous increase in dimensionality and data size correspondingly led to a substantial elevation in Minimum Description Length (MDL) scores. This result is similarly seen in elevated Kullback-Leibler (KL) divergence scores, underscoring the consistency of the observed pattern across both evaluation metrics. An in-depth explo-

ration of this behavior is undertaken through the examination of the cropped variant of the MNIST dataset.

Table 4.14. MNIST KL results for NVP.

	Number of Parameters				
Dataset	**93392**	**390944**	**1752128**	**2892560**	**4390752**
MNIST16	160.0127	121.6242	63.9740	54.8563	45.1414
MNIST28	392.1704	272.4014	161.4538	152.9371	138.7165

Table 4.15. MNIST MDL metric results for NVP.

	Number of Parameters				
Dataset	**93392**	**390944**	**1752128**	**2892560**	**4390752**
MNIST16	113207787	109490439	106056492	111700058	119270894
MNIST28	407414619	389134790	380665813	398174302	418258452

Our findings for MNIST datasets can be summarized as:

- High capacity models were favoured for both GAN and NVP models.
- Increased dimensionality and data sizes lead to greater MDL and KL scores for both models.
- The architecture advised is the same for both MNIST28 and MNIST 16 datasets.

4.5. Discussion

One of the primary objectives of this study is to compare two prominent types of generative model architectures. NVP models were able to outperform GANs for almost all datasets with respect to KL and MDL-based metric scores. GANs had unstable behaviour which is attributed to GAN training as explained in Chapter 2. GAN models generally require a large sample size for effective training due to their implicit nature. Conversely, the NVP models displayed decreased KL scores for increased model

complexity across all training settings. This phenamenon is attributed to their stable training stemming from their loss function.

A crucial factor in our investigation was data dimensionality and variability, which significantly impacts the performance and scalability of the generative models evaluated. We analyzed the dimensionality of the datasets at two levels due to computational limition. We observed that increased variability and dimensionality resulted in elevated MDL and KL scores for both models. Furthermore, as the dataset variability increased, both models required greater variability to accommodate the additional complexity, as reflected in the KL and MDL based metric scores. NVP models were capable of capturing more variability compared to GANs as can be seen inspecting our low-dimensional datasets visually.

We conducted an evaluation of the abovementioned models and architecture settings utilizing various datasets with respect to KL and MDL-based metric scores. The KL estimate scores were employed to assess the data capturing ability of generative models. Furthermore, we introduced the MDL-based metric to evaluate the relationship between model complexity and data capturing ability. As stated in the literature, many existing metrics fail to capture the memorization property of model training, and there is not many metrics that focus on the generalization property of models. Therefore, our proposed MDL-based metric can resolve these issues. Our results demonstrate that our proposed MDL-based metric can serve to further investigate the architectural setting of generative models. Additionally, our findings suggest that our proposed MDL-based metric is consistent with the intuition that greater variability and dimensionality requires greater complexity.

The findings of this study have several practical implications for researchers and practitioners in the field of deep generative modeling. First, the choice between GANs and NVP flows should be made considering data dimensionality and variability. Second, the introduced MDL-based metric can serve as a valuable aid in model selection and hyperparameter tuning. By doing so, it encourages and facilitates the enhance-

ment of model development processes, making them more proficient in achieving their objectives.

In conclusion, this study contributes to our understanding of deep generative models and their applicability across diverse scenarios. By considering the model architecture and data dimensionality and variability factors, we provide insights for making more informed choices in designing and deploying generative models. In our study, we furnish methodologies to facilitate the comparison of the encoding efficiency among various deep generative models, offering valuable insights into their respective capacities for representing information and data patterns.

5. CONCLUSION

In this study, we investigate the architecture and performance of Deep Generative Models (GANs) and flow-based models. We compared their performance against synthetic datasets and the MNIST image dataset, considering the data dimensionality and variability with respect to their architectural complexity. We introduced an MDL-based metric to measure model performance with respect to model complexity. We performed several tests on two levels of data dimensionality and variability. The proposed MDL metric was tested using these datasets. Our experiments show that this metric can be used to estimate the generalization capacities with respect to the divergence behavior of generative models and can also aid practitioners in model selection. According to our results, NVP models perform better than GANs in encoding the variability of low-dimensional synthetic datasets with fewer parameters. Increased dimensionality of syntetic datasets from 2D to 4D makes both model's performance with respect to KL and MDL-based metric scores closer.

This study's findings have several research implications. We present information on Generative Models and metrics used to evaluate the performance of Generative Models. We introduce a metric based on Minimum Description Length (MDL) that assesses the performance of a model in relation to its complexity. We provide an implementation of the proposed MDL-based metric to compare and evaluate the performance of two types of generative models, namely GANs and RealNVPs, based on their architecture and complexity in relation to the variability and dimensionality of various datasets.

This study has several practical implications for practitioners. We provide a method that can assist practitioners in selecting between various architectures, by comparing generative models based on the proposed MDL-based metric and KL divergences. The method can also aid practitioners in choosing suitable generative models and model architectures that are well-suited for their datasets, considering the vari-

ability and dimensionality of their data. Additionally, we provide estimated KL results for the performance of two different types of generative models using various datasets.

The effect of dimensionality on capturing ability of models is investigated on only at two levels and for a specific type of nonlinear transformation. This is a limitation that requires further exploration in order to assess the effect of data dimensionality on the proposed MDL-based metric. Moreover, the model architecture and types of generative models utilized require further exploration.

We aim to improve the capability of the metric against different datasets and models. The current metric does not incorporate sample diversity. We also want to increase the model architectures included in the comparison and compare them with Generative Forests, which also have a low parameter size and can capture variability. The proposed MDL-based metric and MDL framework can be used to investigate dimensionality and overfitting behavior in Deep Generative Models.

REFERENCES

1. Tomczak, J. M., *Deep Generative Modeling*, Springer International Publishing, Eindhoven, 2022.

2. Borji, A., "Pros and Cons of GAN Evaluation Measures: New Developments", ArXiv:2103.09396 [cs.LG], 2021.

3. Murphy, K. P., *Probabilistic Machine Learning: Advanced Topics*, The MIT Press, Vancouver, 2023.

4. Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford and X. Chen, "Improved Techniques for Training GANs", ArXiv:1606.03498 [cs.LG], 2016.

5. Heusel, M., H. Ramsauer, T. Unterthiner, B. Nessler and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium", ArXiv:1706.08500 [cs.LG], 2018.

6. Thanh-Tung, H. and T. Tran, "Toward a Generalization Metric for Deep Generative Models", ArXiv:2011.00754 [cs.LG], 2011.

7. Lucic, M., K. Kurach, M. Michalski, S. Gelly and O. Bousquet, "Are GANs Created Equal? A Large-Scale Study", ArXiv:1711.10337 [stat.ML], 2018.

8. Appel, M. and F. Prietzel, "The Detection of Political Deepfakes", *Journal of Computer-Mediated Communication*, Vol. 27, No. 4, p. zmac008, 2022.

9. Bengio, Y., A. Courville and P. Vincent, "Representation Learning: A Review and New Perspectives", ArXiv:1206.5538 [cs.LG], 2014.

10. Balestriero, R., M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping,

Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun and M. Goldblum, "A Cookbook of Self-Supervised Learning", ArXiv:2304.12210 [cs.LG], 2023.

11. Goodfellow, I. J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Networks", ArXiv:1406.2661 [stat.ML], 2014.

12. Nie, W. and A. Patel, "Towards a Better Understanding and Regularization of GAN Training Dynamics", ArXiv:1806.09235 [stat.ML], 2019.

13. Huszár, F., "How (not) to Train Your Generative Model: Scheduled Sampling, Likelihood, Adversary?", ArXiv:1511.05101 [stat.ML], 2015.

14. Arjovsky, M. and L. Bottou, "Towards Principled Methods for Training Generative Adversarial Networks", ArXiv:1701.04862 [stat.ML], 2017.

15. Li, Y., K. Swersky and R. Zemel, "Generative Moment Matching Networks", ArXiv:1502.02761 [cs.LG], 2015.

16. Dziugaite, G. K., D. M. Roy and Z. Ghahramani, "Training Generative Neural Networks via Maximum Mean Discrepancy Optimization", ArXiv:1505.03906 [stat.ML], 2016.

17. Nowozin, S., B. Cseke and R. Tomioka, "f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization", ArXiv:1606.00709 [stat.ML], 2016.

18. Wu, Y., P. Zhou, A. G. Wilson, E. P. Xing and Z. Hu, "Improving GAN Training with Probability Ratio Clipping and Sample Reweighting", ArXiv:2006.06900 [cs.LG], 2020.

19. Shoshan, A., N. Bhonker, I. Kviatkovsky and G. Medioni, "GAN-Control: Explicitly Controllable GANs", ArXiv:2101.02477 [cs.CV], 2021.

20. Bau, D., H. Strobelt, W. Peebles, J. Wulff, B. Zhou, J.-Y. Zhu and A. Torralba, "Semantic Photo Manipulation with a Generative Image Prior", *ACM Transactions on Graphics*, Vol. 38, No. 4, pp. 1–11, 2019.

21. Chong, M. J., H.-Y. Lee and D. Forsyth, "StyleGAN of All Trades: Image Manipulation with Only Pretrained StyleGAN", ArXiv:2111.01619 [cs.CV], 2021.

22. Shaham, T. R., T. Dekel and T. Michaeli, "SinGAN: Learning a Generative Model from a Single Natural Image", ArXiv:1905.01164 [cs.CV], 2019.

23. Huang, J., Y. Jiao, Z. Li, S. Liu, Y. Wang and Y. Yang, "An Error Analysis of Generative Adversarial Networks for Learning Distributions", ArXiv:2105.13010 [cs.LG], 2022.

24. Arjovsky, M., S. Chintala and L. Bottou, "Wasserstein GAN", ArXiv:1701.07875 [stat.ML], 2017.

25. Srivastava, A., L. Valkov, C. Russell, M. U. Gutmann and C. Sutton, "VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning", ArXiv:1705.07761 [stat.ML], 2017.

26. Neyshabur, B., S. Bhojanapalli and A. Chakrabarti, "Stabilizing GAN Training with Multiple Random Projections", ArXiv:1705.07831 [cs.LG], 2018.

27. Farnia, F., W. Wang, S. Das and A. Jadbabaie, "GAT-GMM: Generative Adversarial Training for Gaussian Mixture Models", ArXiv:2006.10293 [cs.LG], 2020.

28. Huang, W., R. Y. Da Xu, S. Jiang, X. Liang and I. Oppermann, "GAN-based Gaussian Mixture Model Responsibility Learning", *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 3467–3474, Milano, Italy, 2021.

29. Richardson, E. and Y. Weiss, "On GANs and GMMs", ArXiv:1805.12462 [cs.CV], 2018.

30. Karras, T., S. Laine and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks", ArXiv:1812.04948 [cs.NE], 2019.

31. Brock, A., J. Donahue and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis", ArXiv:1809.11096 [cs.LG], 2019.

32. Goodfellow, I., "NIPS 2016 Tutorial: Generative Adversarial Networks", ArXiv:1701.00160 [cs.LG], 2017.

33. Dinh, L., J. Sohl-Dickstein and S. Bengio, "Density Estimation Using Real NVP", ArXiv:1605.08803 [cs.LG], 2017.

34. Kingma, D. P. and P. Dhariwal, "Glow: Generative Flow with Invertible 1x1 Convolutions", ArXiv:1807.03039 [stat.ML], 2018.

35. Theis, L., A. van den Oord and M. Bethge, "A Note On the Evaluation of Generative Models", ArXiv:1511.01844 [stat.ML], 2016.

36. Im, D. J., C. D. Kim, H. Jiang and R. Memisevic, "Generating Images with Recurrent Adversarial Networks", ArXiv:1602.05110 [cs.LG], 2016.

37. Hoogeboom, E., J. W. T. Peters, R. van den Berg and M. Welling, "Integer Discrete Flows and Lossless Compression", ArXiv:1905.07376 [cs], 2019.

38. Goodfellow, I. J., Y. Bengio and A. Courville, *Deep Learning*, MIT Press, Cambridge, 2016.

39. Su, J., D. V. Vargas and K. Sakurai, "One Pixel Attack for Fooling Deep Neural Networks", *IEEE Transactions on Evolutionary Computation*, Vol. 23, No. 5, pp. 828–841, 2019.

40. Sajjadi, M. S. M., O. Bachem, M. Lucic, O. Bousquet and S. Gelly, "Assessing Generative Models via Precision and Recall", ArXiv:1806.00035 [stat.ML], 2018.

41. Liu, F., W. Xu, J. Lu, G. Zhang, A. Gretton and D. J. Sutherland, "Learning Deep Kernels for Non-Parametric Two-Sample Tests", ArXiv:2002.09116 [stat.ML], 2021.

42. Arora, S. and Y. Zhang, "Do GANs Actually Learn the Distribution? An Empirical Study", ArXiv:1706.08224 [cs.LG], 2017.

43. MacKay, D. J. C., *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, Cambridge, 2005.

44. Arora, S., R. Ge, Y. Liang, T. Ma and Y. Zhang, "Generalization and Equilibrium in Generative Adversarial Nets (GANs)", *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 2017.

45. Molkov, Y. I., D. N. Mukhin, E. M. Loskutov, A. M. Feigin and G. A. Fidelin, "Using the Minimum Description Length Principle for Global Reconstruction of Dynamic Systems from Noisy Time Series", *Physical Review E*, Vol. 80, No. 4, p. 046207, 2009.

46. Tipping, M. E. and C. M. Bishop, "Probabilistic Principal Component Analysis", *Journal of The Royal Statistical Society Series B-statistical Methodology*, Vol. 61, No. 3, pp. 611–622, 1999.

47. Bishop, C. M., *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, Cambridge, 2007.

48. Zaheer, M. and B. Póczos, "Connoisseur : Can GANs Learn Simple 1D Parametric Distributions?", 2018, `https://chunliangli.github.io/docs/dltp17gan.pdf`, accessed on November 21, 2022.

49. Wang, Q., S. R. Kulkarni and S. Verdu, "Divergence Estimation for Multidimensional Densities via k-Nearest-Neighbor Distances", *IEEE Transactions on Information Theory*, Vol. 55, No. 5, pp. 2392–2405, 2009.

50. Zhao, P. and L. Lai, "Minimax Optimal Estimation of KL Divergence for Continuous Distributions", *IEEE Transactions on Information Theory*, Vol. 66, No. 12, pp. 7787–7811, 2020.

51. Potapov, A. and M. Peterson, "A Representational MDL Framework for Improving Learning Power of Neural Network Formalisms", *Artificial Intelligence Applications and Innovations*, Springer, Halkidiki, Greece, 2012.

APPENDIX A: TABLES

The presented results are for the GAN architecture with depth 4 and parameter size of 955. For instance G0.01D0.01 stands for the generator learning rate of 0.01 and discriminator learning rate of 0.01 all settings follow the same design respectively. All presented models are trained against 2D GMM high variance dataset.

Table A.1. KL & MDL results for different learning rates for a GAN.

Learning Rates	**KL**	**MDL**
G0.01D0.01	7.4222	64191
G0.01D0.02	6.1992	58996
G0.01D0.03	8.9568	69614
G0.01D0.04	7.0606	62750
G0.01D0.05	7.5893	64834
G0.02D0.02	9.1110	70107
G0.02D0.03	7.9361	66123
G0.02D0.04	8.0710	66609
G0.02D0.05	8.2307	69144
G0.03D0.03	7.3932	67459
G0.03D0.04	6.6637	61081
G0.03D0.05	5.7949	57050
G0.04D0.04	5.9660	57890
G0.04D0.05	7.9562	66196
G0.05D0.05	5.3813	54913
G0.001D0.001	7.4272	64211
G0.001D0.002	7.1731	63206
G0.001D0.003	7.2116	63361
G0.001D0.004	7.2708	63597
G0.001D0.005	7.4708	64380

Table A.1. KL & MDL results for different learning rates for a GAN (cont.).

Learning Rates	**KL**	**MDL**
G0.002D0.002	7.0076	62533
G0.002D0.003	7.5312	64612
G0.002D0.004	7.4416	64267
G0.002D0.005	7.5995	64872
G0.003D0.003	7.4382	64254
G0.003D0.004	7.6969	65240
G0.003D0.005	7.3586	63943
G0.004D0.004	6.7789	61575
G0.004D0.005	7.6560	65086
G0.005D0.005	7.4601	64338
G0.0001D0.0001	7.7191	65323
G0.0001D0.0002	7.2411	63478
G0.0001D0.0003	7.0184	62577
G0.0001D0.0004	7.1381	63065
G0.0001D0.0005	7.0402	62666
G0.0002D0.0002	7.1993	63311
G0.0002D0.0003	7.1349	63052
G0.0002D0.0004	6.7452	61432
G0.0002D0.0005	7.0926	62880
G0.0003D0.0003	7.2162	63379
G0.0003D0.0004	7.1565	63139
G0.0003D0.0005	6.9609	62340
G0.0004D0.0004	7.2442	63491
G0.0004D0.0005	7.2616	63560
G0.0005D0.0005	7.4601	64338
G0.0001D0.0001	7.7191	65323
G0.0001D0.0002	7.2411	63478

Table A.1. KL & MDL results for different learning rates for a GAN (cont.).

Learning Rates	**KL**	**MDL**
G0.0001D0.0003	7.0184	62577
G0.0001D0.0004	7.1381	63065
G0.0001D0.0005	7.0402	62666
G0.0002D0.0002	7.1993	63311
G0.0002D0.0003	7.1349	63052
G0.0002D0.0004	6.7452	61432
G0.0002D0.0005	7.0926	62880
G0.0003D0.0003	7.2162	63379
G0.0003D0.0004	7.1565	63139
G0.0003D0.0005	6.9609	62340
G0.0004D0.0004	7.2442	63491
G0.0004D0.0005	7.2616	63560
G0.0005D0.0005	7.4601	64338