Modelling High-Dimensional Data with Likelihood-Based Generative Models

BOND-TAYLOR, SAMUEL,EDWARD

Modelling High-Dimensional Data with Likelihood-Based Generative Models

Sam Bond-Taylor

A Thesis presented for the degree of
Doctor of Philosophy

Department of Computer Science
Durham University
United Kingdom
March 2024

Abstract

Deep generative models are a class of techniques that use neural networks to form a probabilistic model of training data so that new samples from the data distribution can be generated. This type of learning method necessitates greater level of understanding of the underlying data than common supervised learning approaches. As such, generative models have proven to be useful for much more than simply generation, with applications including, but far from limited to, segmentation, semantic correspondence, classification, image translation, representation learning, solving inverse and ill-posed problems, disentanglement, and out of distribution detection.

This thesis explores how to efficiently scale deep generative models to very high dimensional data in order to aid application to cases where fine details are crucial to capture, as well as data such as 3D models and video. A comprehensive review is carried out, from fundamentals to recent state-of-the-art approaches, to understand the properties of different classes of approaches, diagnose what makes scaling difficult, and explore how techniques can be combined to trade off speed, quality, and diversity. Following this, three new generative modelling approaches are introduced, each with a different angle to enable greater scaling. The first approach enables greater scaling by representing samples as continuous functions thereby allowing arbitrary resolution data to be modelled; this is made possible by using latent conditional neural networks to directly map coordinates to content. The second approach compresses data to a highly informative discrete space then models this space with a powerful unconstrained discrete diffusion process, thereby improving sample quality over the first method while allowing faster sampling and better scaling than comparable discrete methods. The final approach extends diffusion models to infinite dimensional spaces, combining the advantages of the first two approaches to allow diverse, high quality samples, at arbitrary resolutions. Based on these findings and current research, the thesis closes by discussing the state of generative models, future research directions, and ethical considerations of the field.

Declaration

The work in this thesis is based on research carried out at the Department of Computer Science, Durham University, United Kingdom. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Dr. Chris Willcocks, for his invaluable guidance, support, and mentorship throughout my PhD. His dedication, insightful discussions, and unwavering belief in me have been instrumental in shaping this work. I am truly grateful for the freedom he granted me to explore ideas, and for his amazing and valuable feedback that consistently challenged me to think deeper and push the boundaries of my research.

I would also like to extend my heartfelt thanks to all of my co-authors, in particular Peter Hessey and Abril Corona-Figueroa, for their significant contributions and collaborative efforts which have enriched this work immensely.

I am grateful to the members of my thesis committee, Dr. Amir Atapour-Abarghouei and Professor Haiping Lu, for their valuable time examining this thesis.

My sincere appreciation goes to Dr. Robert Powell and all those who maintain and support Durham's NVIDIA CUDA Centre cluster, without which this research would not have been possible.

Much of the work in this thesis was conducted during the COVID-19 pandemic, marked by lockdowns and uncertainties. I would be remiss not to acknowledge the solace and inspiration I found in the beautiful County Durham countryside and the North Pennines, where many ideas for the research conducted in this thesis took shape.

Finally, I am eternally grateful to my family and friends for their unwavering support, and encouragement throughout this time. To my parents and grandparents, thank you for your unwavering belief in me. Last but definitely not least, to my partner, Becky, words cannot express my gratitude for your endless support, patience, and understanding.

Contents

List of Figures

List of Tables

List of Symbols

BERT	Bidirectional Encoder Representations from Transformers
CLIP	Contrastive Language-Image Pre-Training
DDIM	Denoising Diffusion Implicit Model
DDPM	Denoising Diffusion Probabilistic Model
EBM	Energy-Based Model
ELBO	Evidence Lower Bound. Also known as the Variational Lower Bound
ELU	Exponential Linear Unit
FID	Fréchet Inception Distance
FID_{CLIP}	Fréchet Inception Distance using a CLIP feature extractor
FNO	Fourier Neural Operator
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
GPT	Generative Pre-trained Transformer
JS-Divergence	Jensen-Shannon Divergence
KL-Divergence (D_{KL})	Kullback-Leibler Divergence
LPIPS	Learned Perceptual Image Patch Similarity

LSTM	Long Short Term Memory
MAP	Maximum a Posteriori
MLM	Masked Language Model
MLP	Multilayer Perception
MCMC	Monte-Carlo Markov-Chain
$\mathcal{N}(\mu, \Sigma)$	Normal distribution with mean μ and covariance Σ
NLL	Negative Log Likelihood
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PRDC	Precision Recall Density Coverage
RAE	Regularised Autoencoder
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SDE	Stochastic Differential Equation
SN	Spectral Norm
SIREN	Sinusoidal Representation Network
VAE	Variational Autoencoder
VQ-GAN	Vector Quantized Generative Adversarial Network
VQ-VAE	Vector Quantized Variational Autoencoder
VRAM	Video Random Access Memory
WGAN	Wasserstein Generative Adversarial Network

CHAPTER 1

Introduction

The deep learning boom began around 2012 when a number of deep neural networks trained with supervised learning made substantial strides; particularly of note is AlexNet [197], a convolutional neural network which competed in a large scale image classification challenge [303], achieving an impressive error rate 10% lower than the runner up which used feature engineering. Key to AlexNet's success was the depth of the model, accelerated by a fast GPU implementation, together with the large scale of the training dataset consisting of millions of images across thousands of classes.

Since then, major advances have been made in this direction, accelerated by the development of fast deep learning libraries such as PyTorch [274] which feature automatic differentiation tools, greatly simplifying the development of deep neural networks. Nonetheless, large quantities of labelled data are key to the success of many these approaches and collecting such data is incredibly labour intensive. Often this data is gathered through Amazon Mechanical Turk, a website which employs contractors to perform such tasks; many ethical issues have been raised against this process particularly due to the abuse of workers rights [99]. Additionally, in many cases, such as medical imaging, manual labelling requires experts who are often

too busy to do this at scale. Supervised learning approaches also tend to be quite fragile, with out-of-distribution inputs giving unexpected outputs, leading to severe vulnerability to malicious attacks [282].

Deep generative models are a class of techniques that use deep neural networks to form a probabilistic model of the training data so that new samples from the training distribution can be generated. Unlike supervised learning, generative models can be trained on unlabelled data, which is much more readily available particularly due the large scale of content on the internet. Additionally, generative models are encouraged to properly understand the underlying data in order to effectively model it, making them much less vulnerable to attack than supervised approaches. Learning to create new data with generative models gives rise to a number of potential applications including representation learning [32], stress testing [275], disentanglement [33], and density estimation [193].

At this point in time there are numerous signs suggesting that we are at the start of another deep learning boom led not by large collections of manually labelled datasets, but instead by considerably larger collections of minimally structured or unlabelled data powered by deep generative models. Such approaches can already be witnessed, implemented into major applications; for instance, Microsoft and Google have added text-generation models into their search engines which allow natural language interaction and are remarkably effective. Similarly, large image generation models have been made created capable of synthesising incredibly realistic images from text descriptions, disrupting major industries [289; 298]. Major advances have also been made in the medical domain including protein generation [301] and medical report generation [157; 246]. These models are not without flaws; however, with them known to store training data [36] so it is important to consider the ethics of such models, particularly the confidentiality aspects of medical research.

Further adding to the allure of generative models is that they are the basis of some biologically plausible learning algorithms such as predictive coding networks [243]. These learn to predict incoming sensory signals using only local information, thereby permitting interesting properties such as loops, which are not possible in standard backpropagation based learning.

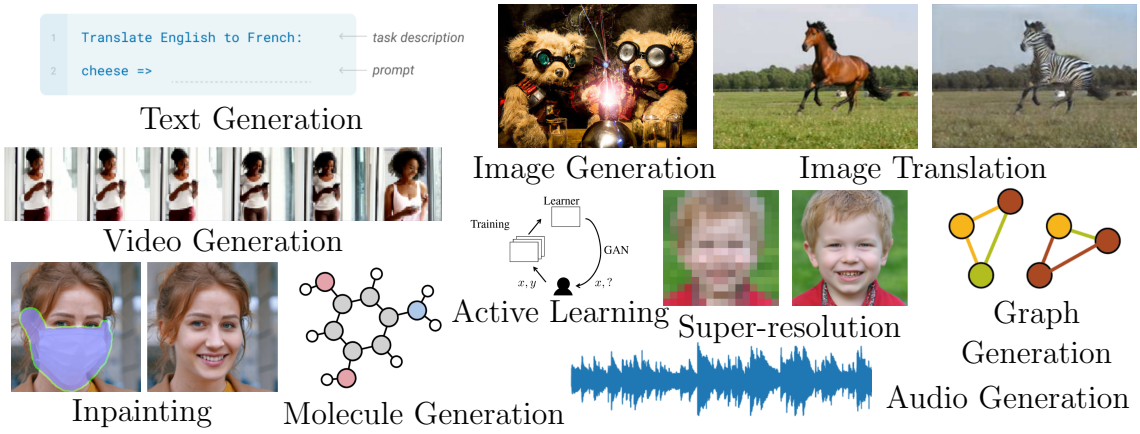


Figure 1.1: Deep Generative Modelling Applications.

Despite the recent success of deep generative models, all current approaches make trade-offs in order to meet specified requirements. This thesis aims to contribute to this growing area of research by exploring how deep generative models can be scaled to exceptionally high-dimensional data whilst minimising the impact on other trade-offs, or even improving them as well. A particular focus is made on image data reflecting the predominance in literature and readily available datasets; however, concepts are relevant across modalities. Each major chapter in this thesis introduces a different approach for high-resolution data generation, each addressing weaknesses associated with the previous.

1.1 Motivation

Deep generative models have been used for a large number of applications (see Fig. 1.1), from directly modelling modalities such as text [30], images [182], videos [137], audio [100], molecules [147], and graphs [365], as well as cross-modal generation such text-to-image [298], text-to-speech [242], image-to-image [408], super-resolution [305], and in-painting [230], to more indirect applications such as reinforcement learning [212], active learning [406], expanding training data [104], and representation learning [397].

By operating on higher resolution data, more information is available and inputs can be brought closer to that of the real world. For example, generative models been shown to be effective for weather forecasting where a deep neural network is

used to predict future weather patterns given only a short context window of the previous weather [290]; in this situation, using high resolution radar is crucial to its success. To effectively model audio, many approaches are applied as close to the raw waveform as possible (48kHz and above is considered high resolution); dimension therefore very quickly grows as the audio length increases making this a difficult challenge. Applying generative models to medical tasks [185], it is useful work as close to the resolution of the used scanners as possible to prevent information loss, to for instance, prevent missing early diagnoses.

There are a variety of different types of generative models, currently each of which make different trade offs; some of key properties the field of generative modelling strives for include:

- **Sample Quality:** High sample quality, is naturally a desirable attribute; however, some approaches sacrifice this for other properties discussed below. Effectively quantifying quality is a difficult task.
- **Mode Coverage:** When some of the training distribution is not modelled, a generative model is said to mode collapse. For a model with limited representation ability, it could make sense to trade this off, improving sample quality at the expense of introducing mode collapse. Conversely, if mode coverage is more important, sample quality could be reduced.
- **Training/Sampling Time:** Longer training and sampling times make models less energy efficient and limit usage in low latency applications. Typically, higher sample quality requires more training and sampling time, so these can be traded off based on time constraints.
- **Training/Sampling Stability:** Some generative models suffer from instability which can make them difficult to train and/or sample from.
- **Useful Representations:** The ability to extract useful representations from generative models gives rise to some downstream applications that can utilise generative models' ability to understand the data.

Generative modelling research aims to improve each of these properties as much as possible. Unfortunately, when working with higher dimensional data many of these issues can be amplified. Realistic samples are much more difficult to generate

at high dimensions, both because distributions are more complex therefore models need to scale accordingly, and because as humans we are more susceptible to noticing artefacts in data closer to that witnessed in the real world. Architectural problems affect most models, with scaling meaning slower run-times and making global information passing more difficult. Furthermore, iterative generative models generally have sampling times that grow with dimension, whether linear or exponential in nature, these times can quickly grow out of hand. High dimensions can also directly impact stability, for instance, the difference between real and generated data can be greater at the start of training thereby leading to instability.

1.2 Contributions

The main contributions of this thesis are as follows:

- A comprehensive review of deep generative models covering the main classes: energy and score-based, variational autoencoders, generative adversarial networks, autoregressive models, and normalizing flows. Comparing and contrasting while reviewing state-of-the-art advances. (Chapter 2)

- A latent-based generative model that utilises empirical Bayes to approximate the posterior rather than using an explicit encoding network. This approach is applied to decoding functions parameterised by implicit representation networks, allowing them to function at arbitrarily high resolutions, unlike traditional encoder/hypernetwork-based approaches. (Chapter 3)

- An approach for fast, high-quality and high-resolution data generation that uses discrete diffusion models to model vector-quantized image representations. In contrast to prior autoregressive approaches, using a discrete diffusion model allows for significantly faster sampling and therefore better scaling to higher resolutions, while also improving sample quality. The bidirectional nature of this approach also allows high quality samples to be generated at resolutions exceeding that of the original training data. (Chapter 4)

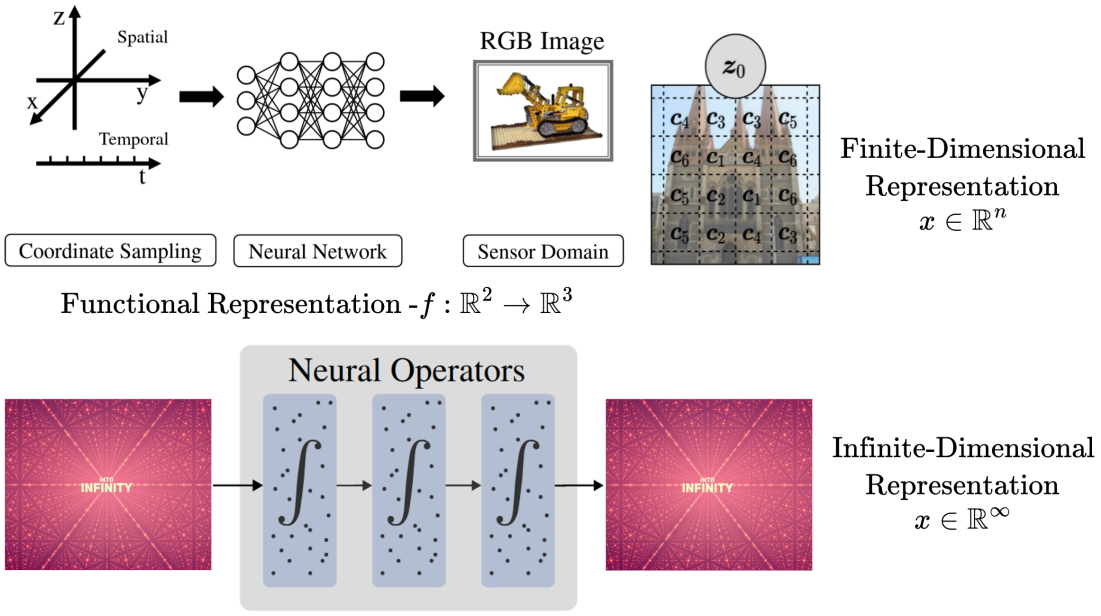


Figure 1.2: Summary of the data modalities used in this thesis. Chapter 3 uses a functional representation of data; Chapter 4 uses a finite-dimensional representation of data; and Chapter 5 uses an infinite-dimensional representation of data.

- A generative diffusion model defined in an infinite-dimensional Hilbert space which theoretically allows for infinite resolution data to be efficiently modelled. It is shown that a subset of the Neural Operators framework can be used to parameterise the denoising function allowing samples to be generated at arbitrary resolutions, provide speedup over finite-dimensional models, while substantially outperforming prior functional generative models. (Chapter 5)

Fig. 1.2 visualises the different approaches of interpreting the same data taken in each chapter.

1.3 Publications

The work contained within this thesis has been previously published in the following peer-reviewed publications by the author, and is used in the chapters as indicated below:

- **Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models**, S.

Bond-Taylor, A. Leach, Y. Long and C. G. Willcocks, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 11, pp. 7327-7347, 2022. (Contributing to Chapter 2).

- **Gradient Origin Networks**, S. Bond-Taylor and C. G. Willcocks, International Conference on Learning Representations, 2021. (Contributing to Chapter 3).

- **Unleashing Transformers: Parallel Token Prediction with Discrete Absorbing Diffusion for Fast High-Resolution Image Generation from Vector-Quantized Codes**, S. Bond-Taylor, P. Hessey, H. Sasaki, T. P. Breckon and C. G. Willcocks, European Conference on Computer Vision (pp. 170-188), 2022. (Contributing to Chapter 4).

- **∞-Diff: Infinite Resolution Diffusion with Subsampled Mollified States**, S. Bond-Taylor, C. G. Willcocks, International Conference on Learning Representations, 2024. (Contributing to Chapter 5).

As well as the above papers, the following works have been published during the period of research for this thesis; however, these publications do not fit into the narrative of this thesis and have not been included in the text.

- **RadEdit: stress-testing biomedical vision models via diffusion image editing**, F. Pérez-Garcaía, S. Bond-Taylor, P. P. Sanchez, B. van Breugel, D. C. Castro, H. Sharma, V. Salvatelli, Maria T.A. Wetscherek, H. Richardson, M. P. Lungren, A. Nori, J. Alvarez-Valle, O. Oktay, M. Ilse, European Conference on Computer Vision, 2024. (Contributions: conducted exploratory research that helped lead to this paper, conducted some experiments and heavily involved in paper writing).

- **Unaligned 2D to 3D Translation with Conditional Vector-Quantized Code Diffusion using Transformers**, A. Corona-Figueroa, S. Bond-Taylor, N. Bhowmik, Y. Gaus, T. P. Breckon, H. P.H. Shum, C. G. Willcocks, International Conference on Computer Vision, 2023. (Contributions: contributed

to conceptual discussions, developed much of the codebase, conducted initial exploratory experiments, and contributed to paper writing).

- **MedNeRF: Medical Neural Radiance Fields for Reconstructing 3D-aware CT-Projections from a Single X-ray**, A. Corona-Figueroa, J. Frawley, S. Bond-Taylor, S. Bethapudi, H. P.H. Shum and C. G. Willcocks, IEEE Engineering in Medicine and Biology Conference, 2022. (Contributions: contributed to conceptual discussions and conducted exploratory experiments).

- **Shape Tracing: An Extension of Sphere Tracing for 3D Non-Convex Collision in Protein Docking**, A. Leach, L. S.P. Rudden, S. Bond-Taylor, J. C. Brigham, M. T. Degiacomi and C. G. Willcocks, IEEE International Conference on Bioinformatics and Bioengineering, 2020. (Contributions: contributed to conceptual discussions).

1.4 Thesis Scope and Structure

Major parts of this thesis are based on the development and training of neural networks in order to address problems with prior work. As such, this thesis is limited by the availability of GPU hardware; all training was performed on single GPUs (primarily the NVIDIA RTX 2080Ti). This limits the scale of models in terms of parameters, training time, and sampling speed, in turn affecting what datasets can reasonably be used. Much of this thesis pushes the boundaries of what has been possible with single GPUs, training models achieving state-of-the-art quantitative scores with a fraction of the hardware of comparable approaches.

This thesis addresses problems associated with generative modelling frameworks, with neural networks used as general function approximators, and not on deep learning itself. As such, knowledge of neural networks and deep learning is assumed but is not essential. There are a number of excellent reviews of deep learning such as those written by Goodfellow et al. [102] and LeCun et al. [209].

In order to develop methods that scale to exceptionally high dimensions while also attempting to maintain or improve other factors such as sample quality and

speed, a variety of types of generative model are used in this thesis. Chapter 2 thoroughly reviews these different models types. In particular, this chapter covers energy-based models, diffusion models, variational autoencoders, generative adversarial networks, autoregressive models, normalizing flows, in addition to numerous hybrid approaches. These techniques, old and new, are compared and contrasted so as to explain the modelling decisions behind each respective techniques.

Chapter 3 explores how implicit representation networks, which represent images as spatially continuous entities by mapping coordinates to pixel values, can be used to parameterise generative models while maintaining their ability to be trained with data sampled at arbitrary coordinates. This is achieved by using an empirical Bayes approach to approximate the posterior of a latent-based model parameterised by an implicit representation network.

Chapter 4 addresses the problems associated with using autoregressive models for modelling vector-quantized image representations: they are sequential therefore slow and their unidirectional nature limits representation ability. This chapter proposes using a discrete diffusion model, allowing tokens to be predicted in parallel for significantly faster image generation while also improving sample quality, obtaining state-of-the-art results on various datasets in terms of precision/recall metrics.

Chapter 5 extends diffusion models to operate in an infinite dimensional Hilbert space, allowing arbitrary resolution data to be sampled. During training, random subsets of coordinates are sampled, and the model learns to denoise the content at those locations. Non-local integral operators which map between Hilbert spaces are used to parameterise the denoising function, with a novel multi-scale architecture proposed able to scale to very high-resolution datasets.

Chapter 6 concludes the thesis, summarising the findings of the previous chapters, discussing their strengths and weaknesses, discusses potential directions for future work, and considers ethical considerations of the field.

1.5 Reproducibility

Implementation details such as specific architecture details and optimiser hyper-parameters not discussed within each chapter are are provided in the appendix. Accompanying open source code is available for each chapter on Github:

- Chapter 3: `https://github.com/cwkx/GON`
- Chapter 4: `https://github.com/samb-t/unleashing-transformers`
- Chapter 5: `https://github.com/samb-t/infty-diff`

CHAPTER 2

Literature Review

The central idea of generative modelling stems around training a generative model whose samples $\tilde{\boldsymbol{x}} \sim p_\theta(\tilde{\boldsymbol{x}})$ come from the same distribution as the training data distribution, $\boldsymbol{x} \sim p_d(\boldsymbol{x})$. Early neural generative models, energy-based models achieved this by defining an energy function on data points proportional to likelihood; however, these struggled to scale to complex high dimensional data such as natural images, and require Markov Chain Monte Carlo (MCMC) sampling during both training and inference, a slow iterative process. In recent years there has been renewed interest in generative models driven by the advent of large freely available datasets as well as advances in both general deep learning architectures and generative models, breaking new ground in terms of visual fidelity and sampling speed. In many cases, this has been achieved using latent variables $\boldsymbol{z}$ which are easy to sample from and/or calculate the density of, instead learning $p(\boldsymbol{x}, \boldsymbol{z})$; this requires marginalisation over the unobserved latent variables, however in general, this is intractable. Generative models therefore typically make trade-offs in execution time, architecture, or optimise proxy functions. Choosing what to optimise for has implications for sample quality, with direct likelihood optimisation often leading to worse sample quality than alternatives.

Table 2.1: Comparison between deep generative models.

Method	Train Speed	Sample Speed	Num. Params.	Resolution Scaling	Free-form Jacobian	Exact Density	FID	NLL (in BPD)
Generative Adversarial Networks								
DCGAN [283]	★★★★★	★★★★★	★★★★★	★★★☆☆	✓	✗	37.11	-
ProGAN [178]	★★★★☆	★★★★★	★★★★☆	★★★★★	✓	✗	15.52	-
BigGAN [29]	★★★★☆	★★★★★	★★★★★	★★★★☆	✓	✗	14.73	-
StyleGAN2 + ADA [180]	★☆☆☆☆	★★★★★	★★★☆☆	★★★★★	✓	✗	2.42	-
Energy Based Models								
IGEBM [75]	★★★☆☆	★★☆☆☆	★★★★★	★★★☆☆	✓	✗	37.9	-
Denoising Diffusion [135]	★★★★★	★★☆☆☆	★★★☆☆	★★★★☆	✓	(✓)	3.17	≤ 3.75
DDPM++ Continuous [329]	★★★★★	★★☆☆☆	★★★☆☆	★★★★☆	✓	(✓)	2.20	-
Flow Contrastive (EBM) [92]	★☆☆☆☆	★★☆☆☆	★★☆☆☆	★★☆☆☆	✓	✗	37.30	≈ 3.27
VAEBM [379]	★★☆☆☆	★★★★☆	★☆☆☆☆	★★★★☆	✓	✗	12.19	-
Variational Autoencoders								
Convolutional VAE [190]	★★★★★	★★★★★	★★★★★	★★☆☆☆	✓	(✓)	106.37	≤ 4.54
Variational Lossy AE [46]	★★★☆☆	★☆☆☆☆	★☆☆☆☆	★★☆☆☆	✗	(✓)	-	≤ 2.95
VQ-VAE [291; 362]	★★☆☆☆	★★★☆☆	★☆☆☆☆	★★★★★	✗	(✓)	-	≤ 4.67
VD-VAE [48]	★☆☆☆☆	★★★★★	★★★☆☆	★★★★★	✓	(✓)	-	≤ 2.87
Autoregressive Models								
PixelRNN [361]	★★☆☆☆	★☆☆☆☆	★☆☆☆☆	★★★☆☆	✗	✓	-	3.00
Gated PixelCNN [360]	★★★☆☆	★☆☆☆☆	★☆☆☆☆	★★★☆☆	✗	✓	65.93	3.03
PixelIQN [268]	★★★☆☆	★☆☆☆☆	★☆☆☆☆	★★★☆☆	✗	✓	49.46	-
Sparse Trans. + DistAug [49; 171]	★★★★☆	★☆☆☆☆	★★★☆☆	★★★☆☆	✗	✓	14.74	2.66
Normalizing Flows								
RealNVP [69]	★★☆☆☆	★★★★★	★★★★★	★★★☆☆	✗	✓	-	3.49
GLOW [191]	★☆☆☆☆	★★★★★	★★☆☆☆	★★★★☆	✗	✓	45.99	3.35
FFJORD [107]	★☆☆☆☆	★★★☆☆	★★★★★	★★☆☆☆	✓	(✓)	-	3.40
Residual Flow [42]	★★☆☆☆	★★★★☆	★★★★☆	★★★★☆	✓	(✓)	46.37	3.28

This chapter provides a comprehensive overview of generative modelling, including advances old and new, comparing and contrasting so as to explain the modelling decisions behind each respective technique. A specific focus on image models is taken reflecting the topics addressed in this thesis; however, concepts are often relevant across modalities. In particular, this chapter covers energy-models (Sec. 2.1), unnormalised density models; diffusion models (Sec. 2.2), reversing a predefined process that destroys information by adding noise; variational autoencoders (Sec. 2.3), variational approximation of a latent-based model's posterior; generative adversarial networks (Sec. 2.4), two models set in a mini-max game; autoregressive models (Sec. 2.5); model data decomposed as a product of conditional probabilities; and normalizing flows (Sec. 2.6), exact likelihood models using invertible transformations. This breakdown is defined to closely match the typical divisions within research; however, numerous hybrid approaches exist that blur these lines, these are discussed in the most relevant section or both where suitable.

An overview of the differences between a collection of popular approaches is provided in Tab. 2.1 which contrasts a diverse array of techniques. For the column "Exact Density", ✓ represents tractable densities, (✓) approximate densities, and ✗ intractable densities. On a number of properties assessed, a star system is used to allow easy comparisons, with rules defined in Tab. 2.2 based on CIFAR-10. Ranking measures such as training speed in days can be considered anecdotal since it is dependent on the year and compute available, with GPUs regularly being released that are more powerful and have more memory. Nevertheless, this allows a comparison based on properties such as stability and convergence rates which cannot be easily judged, for instance, by simply looking at number of function evaluations per iteration.

Table 2.2: Rules for the star ratings in Tab. 2.1. AR is autoregressive sampling, and MCMC is Markov-Chain Monte-Carlo sampling.

	1 Star	2 Stars	3 Stars	4 Stars	5 Stars
Training	>5 days	≤5 days	≤2 days	≤1 days	$\leq\frac{1}{2}$ day
Sampling	AR	MCMC	Middle	≤20 steps	1 step
Params	>120M	≤120M	≤60M	≤30M	≤10M
Resolution	<32	32	64 or 128	256 or 512	≥1024

2.1 Energy-Based Models

Energy-based models (EBMs) [208] are based on the observation that any probability density function $p(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{R}^D$ can be expressed in terms of an energy function $E(\boldsymbol{x})\colon \mathbb{R}^D \to \mathbb{R}$ which associates realistic points with low values and unrealistic points with high values

$$p(\boldsymbol{x}) = \frac{e^{-E(\boldsymbol{x})}}{\int_{\tilde{\boldsymbol{x}} \in \mathcal{X}} e^{-E(\tilde{\boldsymbol{x}})}}. \tag{2.1}$$

Modelling data in such a way offers a number of perks, namely the simplicity and stability associating with training a single model; utilising a shared set of features thereby minimising required parameters; and the lack of any prior assumptions eliminates related bottlenecks [75]. Despite these benefits, scaling to high dimensional data is difficult; however, recent advances have made substantial strides.

A key issue with EBMs is how to optimise them; since the denominator in Eqn. 2.1 is intractable for most models, a popular proxy objective is contrastive divergence where energy values of data samples are 'pushed' down, while samples from the energy distribution are 'pushed' up. Formally, the gradient of the negative log-likelihood loss $\mathcal{L}(\theta) = \mathbb{E}_{\boldsymbol{x} \sim p_d}[-\ln p_\theta(\boldsymbol{x})]$ has been shown to approximately demonstrate the following property [37; 332],

$$\nabla_\theta \mathcal{L} = \mathbb{E}_{\boldsymbol{x}^+ \sim p_d}[\nabla_\theta E_\theta(\boldsymbol{x}^+)] - \mathbb{E}_{\boldsymbol{x}^- \sim p_\theta}[\nabla_\theta E_\theta(\boldsymbol{x}^-)], \tag{2.2}$$

where $\boldsymbol{x}^- \sim p_\theta$ is a sample from the EBM found through a Markov Chain Monte Carlo (MCMC) generating procedure.

2.1.1 Early Energy-Based Models

Before moving to recent advances, we start with some of the earliest neural generative models.

Boltzmann Machines

A Boltzmann machine [131] is a fully connected undirected network of binary neurons (Fig. 2.1a) that are turned on with probability determined by a weighted sum

(a) Boltzmann machine. (b) Restricted Boltzmann machine.

Figure 2.1: Restricted Boltzmann machines have restricted architectures to allow faster sampling than Boltzmann machines.

of their inputs i.e. for some state s_i, $p(s_i = 1) = \sigma(\sum_j w_{i,j} s_j)$. The neurons can be divided into visible $\boldsymbol{v} \in \{0,1\}^D$ units, those which are set by inputs to the model, and hidden $\boldsymbol{h} \in \{0,1\}^P$ units, all other neurons. The energy of the state $\{\boldsymbol{v}, \boldsymbol{h}\}$ is defined (without biases for succinctness) as

$$E_\theta(\boldsymbol{v}, \boldsymbol{h}) = -\frac{1}{2}\boldsymbol{v}^T \boldsymbol{L}\boldsymbol{v} - \frac{1}{2}\boldsymbol{h}^T \boldsymbol{J}\boldsymbol{h} - \frac{1}{2}\boldsymbol{v}^T \boldsymbol{W}\boldsymbol{h}, \tag{2.3}$$

where $\boldsymbol{W}$, $\boldsymbol{L}$, and $\boldsymbol{J}$ are symmetrical learned weight matrices. In order to train Boltzmann machines via contrastive divergence, equilibrium states are found via Gibbs sampling; however, this takes an exponential amount of time in the number of hidden units making scaling impractical.

Restricted Boltzmann Machines

Many of the issues associated with Boltzmann machines can be overcome by restricting their connectivity. One approach, known as the restricted Boltzmann machine (RBM) [132] is to remove connections between units in the same group (Fig. 2.1b), allowing exact calculation of hidden units. Although obtaining negative samples still requires Gibbs sampling, it can be parallelised and in practice a single step is sufficient if $\boldsymbol{v}$ is initially sampled from the dataset [132].

By stacking RBMs, using features from lower down as inputs for the next layer, more powerful functions can be learned; these models are known as deep belief networks [133]. Training an entire model at once is intractable so instead they are trained greedily layer by layer, composing densities thus improving the approximation of $p(\boldsymbol{v})$.

2.1.2 Deep EBMs via Contrastive Divergence

To train more powerful architectures through contrastive divergence, one must be able to efficiently sample from p_θ. Specifically, we would like to model high dimensional data using an energy function with a deep neural network, taking advantage of recent advances in discriminative models [388]. MCMC methods such as random walk and Gibbs sampling [133], when applied to high dimensional data, have long mixing times, making them impractical. A number of recent approaches [75; 381] have advocated the use of stochastic gradient Langevin dynamics [297; 375] which permits sampling through the following iterative process,

$$\boldsymbol{x}_0 \sim p_0(\boldsymbol{x}), \qquad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \frac{\alpha}{2}\frac{\partial E_\theta(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i} + \boldsymbol{\epsilon}, \tag{2.4}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha \boldsymbol{I})$, $p_0(\boldsymbol{x})$ is typically a uniform distribution over the input domain and α is the step size. As the number of updates $N \to \infty$ and $\alpha \to 0$, the distribution of samples converges to p_θ [375]; however, α and $\boldsymbol{\epsilon}$ are often tweaked independently to speed up training.

While Langevin MCMC is more practical than other approaches, sampling still requires a large number of steps. One solution is to use persistent contrastive divergence [75; 340] where a replay buffer stores previously generated samples that are randomly reset to noise; this allows samples to be continually refined with a relatively small number of steps while maintaining diversity. Short-run MCMC [260] which samples using as few as 100 update steps from noise has also been used to train deep EBMs; however, since the number of steps is so small, samples are not truly from the correct probability density. Nevertheless, there are other advantages such as allowing image interpolation and reconstruction (since short-run MCMC does not mix) [259]. Other approaches include initialising MCMC chains with data points [381] and samples from an implicit generative model [380], as well as adversarially training an implicit generative model, mitigating mode collapse somewhat by maximising its entropy [111; 187; 201]. Improved/augmented MCMC samplers with neural networks can also improve the efficiency of sampling [110; 139; 213; 324; 341].

One application of EBMs of this form comes by using standard classifier ar-

chitectures, $f_\theta \colon \mathbb{R}^D \to \mathbb{R}^K$, which map data points to logits used by a softmax function to compute $p_\theta(y|\boldsymbol{x})$. By marginalising out y, these logits can be used to define an energy model that can be simultaneously trained as both a generative and classification model [109],

$$p_\theta(\boldsymbol{x}) = \sum_y p_\theta(\boldsymbol{x}, y) = \frac{\sum_y \exp(f_\theta(\boldsymbol{x}[y]))}{Z(\theta)}, \tag{2.5a}$$

$$E_\theta(\boldsymbol{x}) = -\ln \sum_y \exp(f_\theta(\boldsymbol{x}[y])). \tag{2.5b}$$

2.1.3 Correcting Implicit Generative Models

While EBMs offer powerful representation ability due to unnormalized likelihoods, they can suffer from high variance training, long training and sampling times, and struggle to support the entire data space. In this section, a number of hybrid approaches are discussed which address these issues.

Exponential Tilting

To eliminate the need for an EBM to support the entire space, an EBM can instead be used to correct samples from an implicit generative network, simplifying the function to learn and allowing easier sampling. This procedure, referred to as exponentially tilting an implicit model, is defined as

$$p_{\theta,\phi}(\boldsymbol{x}) = \frac{1}{Z_{\theta,\phi}} q_\phi(\boldsymbol{x}) e^{-E_\theta(\boldsymbol{x})}. \tag{2.6}$$

By parameterising $q_\phi(\boldsymbol{x})$ as a latent variable model such as a normalizing flow [6; 262] or VAE generator [379], MCMC sampling can be performed in the latent space rather than the data space. Since the latent space is much simpler, and often uni-modal, MCMC mixes much more effectively. This limits the freedom of the model, however, leading some to jointly sample in latent and data space [6; 379].

Noise Contrastive Estimation

Noise contrastive estimation [83; 121] transforms EBM training into a classification problem using a noise distribution $q_\phi(\boldsymbol{x})$ by optimising the loss function,

$$\mathbb{E}_{p_d}\left[\ln \frac{p_\theta(\boldsymbol{x})}{p_\theta(\boldsymbol{x}) + q_\phi(\boldsymbol{x})}\right] + \mathbb{E}_{q_\phi}\left[\ln \frac{q_\phi(\boldsymbol{x})}{p_\theta(\boldsymbol{x}) + q_\phi(\boldsymbol{x})}\right], \tag{2.7}$$

where $p_\theta(\boldsymbol{x}) = e^{E_\theta(\boldsymbol{x})-c}$. This approach can be used to train a correction via exponential tilting [262], but can also be used to directly train an EBM and normalizing flow [92]. Eqn. 2.7 is equivalent to GAN Equation 2.31; however, training formulations differ, with noise contrastive estimation explicitly modelling likelihood ratios.

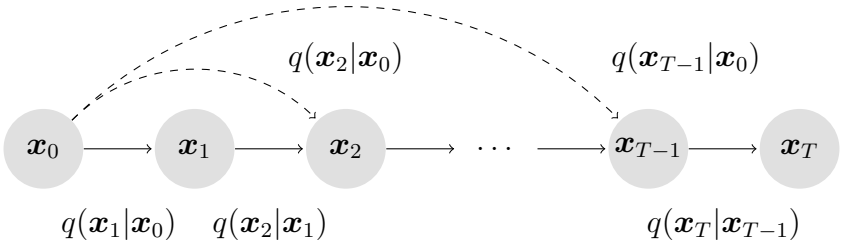
2.1.4 Alternative Training Objectives

As aforementioned, energy models trained with contrastive divergence approximately maximises the likelihood of the data; likelihood however does not correlate directly with sample quality [339]. Training EBMs with arbitrary f-divergences is possible, yielding improved FID scores [386].

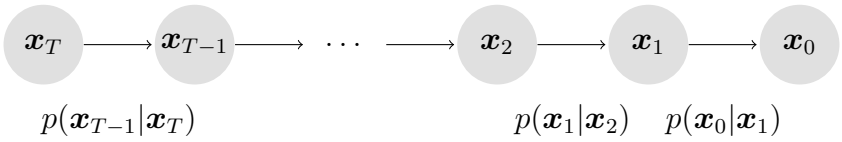
Since score estimates have high variance, the Stein discrepancy has been proposed as an alternative objective, requiring no sampling and more closely correlating with likelihood [108]. A middle ground between denoising score matching and contrastive divergence is diffusion recovery likelihood [20] which can be optimised via a sequence of denoising EBMs conditioned on increasingly noisy samples of the data, the conditional distributions being much easier to MCMC sample from than typical EBMs [93].

2.2 Diffusion Models

While Langevin MCMC has helped EBMs to scale to high dimensional data, training times are still slow due to the need to sample from the model distribution, additionally, the finite nature of the sampling process means that samples can be arbitrarily far away from the model's distribution [108]. Closely related are denoising diffusion probabilitic models (DDPM) (Fig. 2.2) [1; 20; 135; 323] for which the generative



(a) Diffusion Model Forward Process.



(b) Diffusion Model Reverse Process.

(c) Example Gaussian Diffusion Process.

Figure 2.2: Diffusion models consist of a forward process (a) which gradually maps data $\boldsymbol{x}_0$ to noise $\boldsymbol{x}_T$ via a large number of transitions $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}$ (solid lines). Typically $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ (dashed lines) can be represented in closed form, simplifing training. Samples can be generated by learning the reverse of this process, mapping noise to data (b). For Gaussian diffusion models (c), transition distributions are Gaussian.

process, referred to as the reverse process, is defined as a fixed length Markov chain over a sequence of latent variables $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$,

$$p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t), \tag{2.8}$$

$p(\boldsymbol{x}_T)$ is a fixed prior distribution that can be easily sampled from, and $p(\boldsymbol{x}_0)$ is the data distribution. From the perspective of EBMs, this process can be seen as the explicit sampling process (Eqn. 2.4) over a fixed number of steps T. In order to optimise this process, an approximate posterior $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)$ is also defined, thereby

allowing the variational bound (ELBO) on the likelihood $p_\theta(\boldsymbol{x}_0)$ to be calculated,

$$q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}), \tag{2.9}$$

$$\mathbb{E}[\log p_\theta(\boldsymbol{x}_0)] \geq \mathbb{E}_q \left[\log \frac{p_\theta(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)}\right] = \mathbb{E}_q \left[\log p(\boldsymbol{x}_T) + \sum_{t\geq 1} \log \frac{p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})}\right] \tag{2.10}$$

$$= \mathbb{E}_q \Big[D_{KL}(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \,\|\, p(\boldsymbol{x}_T)) + \sum_{t>1} D_{KL}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \,\|\, p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)) - \log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\Big]. \tag{2.11}$$

For long Markov chains, calculating the ELBO at every training step is impractical. However, because the ELBO consists of a sum of independent terms, it can be approximated by Monte Carlo sampling components, $\mathcal{L}_{t-1} = D_{KL}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \,\|\, p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))$. In many cases, diffusion models are defined such that $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ can be represented in closed form and can be easily sampled, simplifying this process.

2.2.1 Gaussian Diffusion Models

One of the most common types of DDPM are Gaussian diffusion models [135; 323], where the transition distributions are Gaussian and the loss $\mathcal{L}_{t-1}$ is a weighted mean squared error.

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_\theta(\boldsymbol{x}_t, t), \sigma_t^2 \boldsymbol{I}), \tag{2.12}$$

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \tag{2.13}$$

$$\mathcal{L}_{t-1} = \mathbb{E}_q \left[\frac{1}{\sigma_t^2} \|\tilde{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \mu_\theta(\boldsymbol{x}_t, t)\|_2^2\right] + C. \tag{2.14}$$

In other words, the reverse process is trained to gradually remove Gaussian noise, until realistic data is formed. The variance schedule $\beta_1, \ldots, \beta_T$ is defined so that each β_t is small but $q(\boldsymbol{x}_T|\boldsymbol{x}_0) \approx \mathcal{N}(\boldsymbol{x}_T; \mathbf{0}, \boldsymbol{I})$. $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ can be represented in closed form as $q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I})$ where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.

To improve sample quality, Ho et al. [135] proposed reweighting the ELBO $\mathcal{L}^{\text{simple}}$, down-weighting loss terms that correspond to low time steps encouraging the model to focus on denoising more noisy inputs. Additionally, by reparameterising

transition distributions, the noise $\boldsymbol{\epsilon}$ can be predicted instead of the mean $\boldsymbol{\mu}$,

$$\mathcal{L}_{t-1}^{\text{simple}} = \mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_0(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}), t\|_2^2\right] + C. \tag{2.15}$$

2.2.2 Connection with Score Matching

A different perspective to diffusion models can be found by studying score matching [158] which is based on the idea of minimising the difference between the derivatives of the data and model's log-density functions, i.e. the score function $\nabla_{\boldsymbol{x}} \ln p(\boldsymbol{x})$. If the score function is known, then it can be learned by minimising the Fisher divergence between p_d and p_θ,

$$\mathcal{L} = \frac{1}{2}\,\mathbb{E}_{p_d(\boldsymbol{x})}[\big\|s_\theta(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \ln p(\boldsymbol{x})\big\|_2^2], \tag{2.16}$$

however, the score function of data is usually not available. Various methods exist to estimate the score function including spectral approximation [318], sliced score matching [328], finite difference score matching [269], and notably denoising score matching [368] which allows the score to be approximated using corrupted data samples $q(\tilde{\boldsymbol{x}}|\boldsymbol{x})$. In particular, when $q = \mathcal{N}(\tilde{\boldsymbol{x}}|\boldsymbol{x}, \sigma^2\boldsymbol{I})$, Eqn. 2.16 simplifies to

$$\mathcal{L} = \frac{1}{2}\,\mathbb{E}_{p_d(\boldsymbol{x})}\,\mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathcal{N}(\boldsymbol{x},\sigma^2\boldsymbol{I})}\left[\left\|s_\theta(\tilde{\boldsymbol{x}}) + \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\sigma^2}\right\|_2^2\right]. \tag{2.17}$$

That is, s_θ learns to estimate the noise thereby allowing it to be used as a generative model [312; 326]. Since the Langevin update step uses $\nabla_{\boldsymbol{x}} \ln p(\boldsymbol{x})$ it is possible to sample from a score matching model using Langevin dynamics [354]. This is only possible, however, when trained over a large variety of noise levels so that $\tilde{\boldsymbol{x}}$ covers the whole space. It is clear that this loss is equivalent to the loss used in Gaussian diffusion models Eqn. 2.15, similarly, with one Langevin step per noise level, each step is similar to sampling a DDPM transition distribution Eqn. 2.12. This connection led to the development of predictor-corrector samplers [329] where Markov transitions can be followed by Langevin update steps to correct the marginal distribution of the sample.

2.2.3 Continuous Time Gaussian Diffusion

Another way of defining a diffusion process is with a stochastic differential equation [329]; here the diffusion process $\{\boldsymbol{x}(t)\}_{t=1}^{T}$ is continuous in time t. Similar to the discrete case, at time $t = 0$, $\boldsymbol{x}(0) \sim p_0$, then noise is slowly added until time $t = T$ where $\boldsymbol{x}(T) \sim p_T$ which contains no information about $\boldsymbol{x}(0)$,

$$\mathrm{d}\boldsymbol{x} = f(\boldsymbol{x}, t)\,\mathrm{d}t + g(t)\,\mathrm{d}\boldsymbol{w}, \tag{2.18}$$

where $\boldsymbol{w}$ is the standard Wiener process (Brownian motion), f is a function that defines the drift of the SDE, and g is a function that defines the diffusion coefficient. The reverse of this SDE, which allows samples to be generated, can be represented as another SDE in terms of the score function [4],

$$\mathrm{d}\boldsymbol{x} = [f(\boldsymbol{x}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})]\,\mathrm{d}t + g(t)\,\mathrm{d}\bar{\boldsymbol{w}}, \tag{2.19}$$

where $\bar{\boldsymbol{w}}$ is the standard Wiener process when time flows backwards. In the same manner as the discrete case, score-matching (such as in Eqn. 2.17) can be used to approximate the score function.

When f and g take the following forms, the continuous time generalisation of the DDPM forward process presented in Section 2.2.1 can be represented as

$$\mathrm{d}\boldsymbol{x} = -\frac{1}{2}\beta(t)\boldsymbol{x}\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{w}. \tag{2.20}$$

with $\beta(t)$ representing the continuous time analogue of β_t.

2.2.4 Diffusion in Discrete State Spaces

Discrete diffusion models [9; 145] constrain the state space so that $\boldsymbol{x}_t$ is a discrete random variable falling into one of K categories. As such, the forward process can be represented as categorical distributions $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathrm{Cat}(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_{t-1}\boldsymbol{Q}_t)$ for one-hot $\boldsymbol{x}_{t-1}$ where $\boldsymbol{Q}_t$ is a matrix denoting the probabilities of moving to each successive state. $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ can be expressed as $q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathrm{Cat}(\boldsymbol{x}_t; \boldsymbol{p} = \overline{\boldsymbol{Q}}_t)$ where

$\overline{\boldsymbol{Q}}_t = \boldsymbol{x}_0\boldsymbol{Q}_1\boldsymbol{Q}_2\cdots\boldsymbol{Q}_t$, therefore scaling is simple if $\overline{\boldsymbol{Q}}_t$ can be expressed in closed form. Transition processes include moving states with some low uniform probability [145], moving to nearby states with some probability based on similarity or distance, and masking inputs similar to generative MLMs.

2.2.5 Speeding up Sampling

Sampling from score-based models requires a large number of steps leading to various techniques being developed to reduce this. A simple approach is to skip steps at inference: cosine schedules [256] spend more time where larger visual changes are made reducing the impact of skipping; another approach is to use dynamic programming to find what steps should be taken to minimise ELBO based on a computation budget [373]. For the continuous time interpretation, more efficient numerical solvers can be used, reducing the number of steps required [169]. Denoising diffusion implicit models (DDIMs) [325] transform diffusion models into deterministic models allowing fewer steps to yield the same quality, replacing the sampling steps with

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}}\cdot\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t). \tag{2.21}$$

Also of interest are different formulations of diffusion models that among other features offer the potential for faster sampling, particularly of note are the recent advances in Schrödinger bridges and Optimal Transport [44; 62] and Poisson Flow [383] where the forward process instead maps to a high dimensional hemisphere.

2.3 Variational Autoencoders

One of the key problems associated with energy-based models is that sampling is not straightforward and mixing can require a significant amount of time. To circumvent this issue, it would be beneficial to explicitly sample from the data distribution with a single network pass.

To this end, suppose we have a latent based model $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ with prior $p_\theta(\boldsymbol{z})$ and posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$; unfortunately optimising this model through maximum like-

(a) VAE computational graph. (b) VAEs encode data to a compressed latent space resulting in imperfect reconstructions e.g. blur.

Figure 2.3: Variational autoencoder with a normally distributed prior. $\boldsymbol{\epsilon}$ is sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

lihood is intractable due to the integral in $p_\theta(\boldsymbol{x}) = \int_{\boldsymbol{z}} p_\theta(\boldsymbol{x}|\boldsymbol{z})p_\theta(\boldsymbol{z})d\boldsymbol{z}$. Instead, variational inference allows this problem to be reframed as an optimisation problem by introducing an approximation of the true intractable posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \arg\min_q D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x}))$ that allows a tractable bound on $p_\theta(\boldsymbol{x})$ to be formed. In particular, variational autoencoders amortize the inference process, that is, approximate $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ using a feedforward inference network allowing scaling to large datasets (Fig. 2.3) [190; 294]. From the definition of KL divergence we get

$$\begin{aligned} D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\ln \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}\right] \\ &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln q_\phi(\boldsymbol{z}|\boldsymbol{x})] - \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{z}, \boldsymbol{x})] + \ln p_\theta(\boldsymbol{x}), \end{aligned} \tag{2.22}$$

which can be rearranged to find an alternative definition for $p_\theta(\boldsymbol{x})$ that does not require the knowledge of $p_\theta(\boldsymbol{z}|\boldsymbol{x})$

$$\begin{aligned} \ln p_\theta(\boldsymbol{x}) &= D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) - \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln q_\phi(\boldsymbol{z}|\boldsymbol{x})] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{z}, \boldsymbol{x})] \\ &\geq -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln q_\phi(\boldsymbol{z}|\boldsymbol{x})] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{z}, \boldsymbol{x})] \\ &= -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln q_\phi(\boldsymbol{z}|\boldsymbol{x})] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{z})] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{x}|\boldsymbol{z})] \\ &= -D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{x}|\boldsymbol{z})] \\ &\equiv \mathcal{L}(\theta, \phi; \boldsymbol{x}), \end{aligned} \tag{2.23}$$

where $\mathcal{L}$ is known as the evidence lower bound (ELBO) [170]. To optimise this bound with respect to θ and ϕ, gradients must be backpropagated through the stochastic sampling process $\tilde{\boldsymbol{z}} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})$. This is permitted by reparameterizing $\tilde{\boldsymbol{z}}$ using a differentiable function $g_\phi(\boldsymbol{\epsilon}, \boldsymbol{x})$ of a noise variable $\boldsymbol{\epsilon}$: $\tilde{\boldsymbol{z}} = g_\phi(\boldsymbol{\epsilon}, \boldsymbol{x})$ with $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$

[190; 294].

Monte Carlo gradient estimators can be used to approximate the expectations; however, this yields very high variance making it impractical. Alternatively, if $D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}))$ can be integrated analytically then the variance is manageable. A prior with such a property needs to be simple enough to sample from but also sufficiently flexible to match the true posterior; a common choice is a normally distributed prior with diagonal covariance, $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\boldsymbol{I})$ with $\tilde{\boldsymbol{z}} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. In this case, the loss simplifies to *

$$\tilde{\mathcal{L}}_{VAE}(\theta, \phi; \boldsymbol{x}) \simeq \frac{1}{2}\sum_{j=1}^{J}\left(1 + \ln((\sigma^{(j)})^2) - (\mu^{(j)})^2 - (\sigma^{(j)})^2\right) + \frac{1}{L}\sum_{l=1}^{L}\ln p_\theta(\boldsymbol{x}|\tilde{\boldsymbol{z}}_l). \quad (2.24)$$

Despite success on small scale datasets, when applied to more complex datasets such as natural images, samples tend to be unrealistic and blurry [74]. This blurriness has been attributed to the maximum likelihood objective itself and MSE reconstruction loss; however, there is evidence that limited approximation of the true posterior is the root cause [400]; with MSE causing highly non-Gaussian posteriors. As such, the Gaussian posterior implies an overly simple model which, when unable to perfectly fit, maps multiple data points to the same encoding leading to averaging.

There are a number of other issues associated with limited posterior approximation, namely under-estimation of the variance of the posterior, resulting in poor predictions, and biases in the MAP estimates of model parameters [352]. Additionally, amortized inference leads to an amortization gap, the difference in ELBO for the amortized posterior and optimal approximate posterior [58]. Increasing the capacity of the encoder and decoder can reduce this gap by improving the posterior approximation and better fitting the choice of approximation respectively. Other proposed improvements include combining with adversarial training [152; 205; 236], improving the ELBO [31], as well as using different regularisation such as Wasserstein distance [342].

Reweighting the ELBO by multiplying D_{KL} with an extra hyperparameter β allows the capacity of the latent representation to be altered. When $\beta > 1$ a more

disentangled representation is learned where each latent unit is responsible for a single generative factor [130]. This approach has been generalised, allowing more precise states in the compression-representation trade-off to be targeted [3].

2.3.1 Beyond Simple Priors

One approach to improve variational bounds and increase sample quality is to improve the priors used for instance by careful selection to the task or by increasing its complexity [140]. Complex priors can be learned by warping simple distributions and inducing variational dependencies between the latent variables: variational Gaussian processes permit this by forming an infinite ensemble of mean-field distributions [346]; EBMs and score matching can be used to model flexible priors [269; 357]; normalizing flows (see Section 2.6) transform distributions through a series of invertible parameterised functions [21; 107; 149; 192; 293; 308].

By rewriting the VAE training objective to have two regularisation terms [236] , one on each line,

$$\begin{aligned}\mathcal{L}(\theta, \phi; \boldsymbol{x}) = \mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}[\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\ln p_\theta(\boldsymbol{x}|\boldsymbol{z})]] \\ + \mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}[\mathbb{H}[q_\phi(\boldsymbol{z}|\boldsymbol{x})]] - \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z})}[p_\theta(\boldsymbol{z})],\end{aligned} \tag{2.25}$$

the latter of which is the cross entropy between the aggregate posterior and the prior, the prior can be defined as the aggregate posterior (Fig. 2.4), thus obtaining a rich multi-modal latent representation that combats inactive latent variables. Since the true aggregate posterior is intractable, VampPrior [344] approximates it for a set of pseudo-inputs, tensors with the same shape as data points learned during training. Exemplar VAEs [263] scale this approach up, using the full training set to approximate the aggregate posterior, by approximating the prior using k-nearest-neighbours. Alternatively, the aggregate posterior can be approximated with a learned prior; this has been achieved with a learned rejection sampling procedure that transforms a base distribution [15].

In some instances, it can be helpful to compress data to discrete latent representations [28; 172]; however, gradients through discrete sampling procedures are

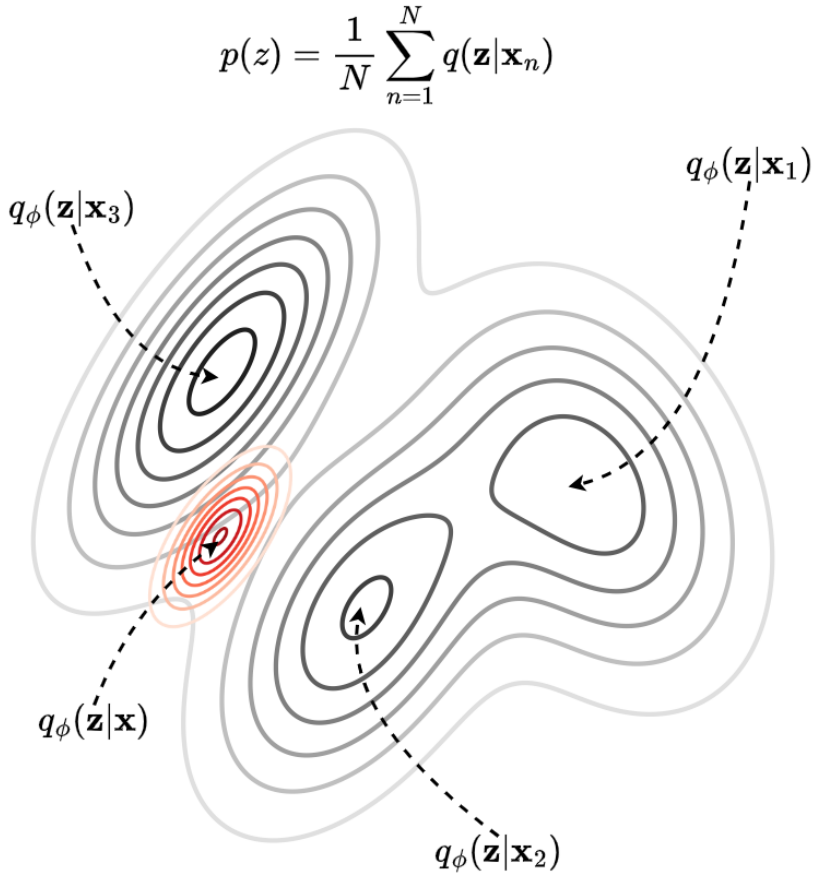


Figure 2.4: The prior $p(\boldsymbol{z})$ of a VAE can be defined as the aggregate posterior, $p(\boldsymbol{z}) = \frac{1}{N}\sum_{n=1}^{N} q(\boldsymbol{z}|\boldsymbol{x}_n)$, indicated by grey contour lines. Directly using the dataset $\{\boldsymbol{x}_n\}$ is impractical however, due to overfitting and computational complexity. To address this, rather than directly using the dataset, VampPrior [344] instead uses pseudo-inputs, while Exemplar VAEs [263] use a k-nearest-neighbours approximation.

ill-defined. The Gumbel-Softmax/Concrete distribution is a differentiable continuous approximation of a categorical distribution containing a temperature coefficient that converges to a discrete distribution in the limit [163; 234].

Alternatively, it has been argued that simple Gaussian priors are not a hindrance. When the data of dimension d lies on a sub-manifold of dimension r and $r < d$ then global VAE optimum exist that do no recover the data distribution; however, when $r = d$, global optimums do recover the data distribution; as such, 2 stage VAEs that first map data to latents of dimension r then use a second VAE to correct the learned density can better capture the data [60].

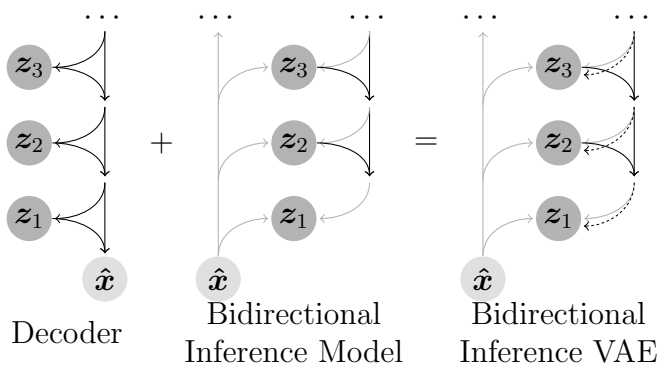


Figure 2.5: A hierarchical VAE with bidirectional inference [192].

Hierarchical VAEs

Hierarchical VAEs build complex priors with multiple levels of latent variables, each conditionally dependent on the last, forming dependencies depthwise though the network,

$$p_\theta(\boldsymbol{z}) = p_\theta(\boldsymbol{z}_0)p_\theta(\boldsymbol{z}_1|\boldsymbol{z}_0)\cdots p_\theta(\boldsymbol{z}_N|\boldsymbol{z}_{<N}), \tag{2.26a}$$

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = q_\phi(\boldsymbol{z}_0|\boldsymbol{x})q_\phi(\boldsymbol{z}_1|\boldsymbol{z}_0,\boldsymbol{x})\cdots q_\phi(\boldsymbol{z}_N|\boldsymbol{z}_{<N},\boldsymbol{x}). \tag{2.26b}$$

Ladder VAEs [333] achieve this conditioning structure using a bidirectional inference network where a deterministic "bottom-up" pass generates features at various resolutions, then the latent variables are processed from top to bottom with the features shared (Fig. 2.5). Specifically, they model latents as normal distributions conditioned on the last latent,

$$p_\theta(\boldsymbol{z}_i|\boldsymbol{z}_{i-1}) = \mathcal{N}(\boldsymbol{z}_i|\mu_{p,i}(\boldsymbol{z}_{i-1}), \sigma^2_{p,i}(\boldsymbol{z}_{i-1})). \tag{2.27}$$

By introducing skip connections around the stochastic sampling process, latents can be conditioned on all previously sampled latents [192; 233; 356]. Such an architecture generalises autoregressive models; inferring latents in parallel allows for significantly fewer steps compared to typical autoregressive models since many latents are statistically independent and allows different latent levels to correspond to global/local details depending on their depth. It has been argued that a single level of latents is sufficient since Gibbs sampling performed on that level can recover

the data distribution [399]. Despite that, Gibbs sampling converges slowly, making hierarchical representations more efficient; in support of this, deeper hierarchical VAEs have been shown to improve likelihood, independent of capacity [48].

2.3.2 Regularised Autoencoders

Related to VAEs are regularised autoencoders (RAEs) which apply regularisation to the latent space of a deterministic autoencoder then subsequently train a density estimator on this space to obtain a complex prior [98]. Since the approximate posterior is a degenerate distribution, RAEs have little connection with variational inference.

Vector Quantized-Variational Autoencoders (VQ-VAE) [288; 362] learn a highly compressed discrete representation taking advantage of an information rich codebook to achieve extremely high compression rates compared to continuous representations. A convolutional encoder downsamples images $\boldsymbol{x}$ to a smaller spatial resolution, $E(\boldsymbol{x}) = \{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_L\} \in \mathbb{R}^{L \times D}$. A simple quantisation approach is to use the argmax operation which maps continuous encodings to their closest elements in a finite codebook of vectors [362]. Specifically, for a codebook $\mathcal{C} \in \mathbb{R}^{K \times D}$, where K is the number of discrete codes in the codebook and D is the dimension of each code, each $\boldsymbol{e}_i$ is mapped via a nearest-neighbour lookup onto a discrete codebook value, $\boldsymbol{c}_j \in \mathcal{C}$:

$$\boldsymbol{z}_q = \{\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_L\} \text{ , where } \boldsymbol{q}_i = \min_{\boldsymbol{c}_j \in \mathcal{C}} \|\boldsymbol{e}_i - \boldsymbol{c}_j\|. \tag{2.28}$$

As this operation is non-differentiable, the straight-through gradient estimator [18] is used to approximate gradients resulting in bias. The quantized latents are fed through a decoder $\hat{\boldsymbol{x}} = G(\boldsymbol{z}_q)$ to reconstruct the input based on a perceptual reconstruction loss [86; 396]; this process is trained by minimising the loss $\mathcal{L}_{\text{VQ}}$,

$$\mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{rec}} + \|\text{sg}[E(\boldsymbol{x})] - \boldsymbol{z}_q\|_2^2 + \beta \|\text{sg}[\boldsymbol{z}_q] - E(\boldsymbol{x})\|_2^2. \tag{2.29}$$

Typically, to allow sampling, autoregressive models (see Section 2.5) are used to learn the compressed discrete spaces since they are powerful density estimators and the substantially reduced input dimension means that sampling is possible in reasonable

times.

2.3.3 Data Modelling Distributions

Unlike energy-based models, VAEs must model an explicit density $p(\boldsymbol{x}|\boldsymbol{z})$. For efficient sampling, typically this distribution is decomposed as a product of independent simple distributions, allowing unrestricted architectures to be used to parameterise the chosen distributions. Common instances include modelling variables as Bernoulli [226], Gaussian [190], multinomial distributions, or as mixtures[310].

Autoregressive Decoders

To introduce dependencies between the output variables, numerous works have used powerful autoregressive networks [119]. While these approaches allow complex distributions to be learned, they increase the runtime and often suffer from posterior collapse since early in training the approximate posterior contains little knowledge about $\boldsymbol{x}$ meaning that it is easy to minimise D_{KL} which in turn reduces the gradient between the encoder and decoder making it difficult to escape this minima [28]; in fact, for a sufficiently powerful generative distribution, this can occur even at optimum solutions [46]. Various methods to prevent posterior collapse have been proposed: by restricting the autoregressive network's receptive field to a small window, it is forced to use latents to capture global structure [46]; a mutual information term can be added to the loss to encourage high correlation between $\boldsymbol{x}$ and $\boldsymbol{z}$ [401]; encouraging the posterior to be diverse by controlling its geometry to evenly covering the data space, redundancy is reduced and latents are encouraged to learn global structure [232].

2.3.4 Bridging Amortized and Stochastic Inference

While variational approaches offer substantial speedup over MCMC sampling, there is an inherent discrepancy between the true posterior and approximate posterior despite improvements in this field. To this end, a number of approaches have been proposed to find a middle ground, yielding improvements over amortized methods

with lower costs than MCMC. Semi-amortised VAEs [188] use an encoder network followed by stochastic gradient descent on latents to improve the ELBO; however, this still relies on an inference network. The inference network can be removed by assigning latent vectors to data points, then optimising them with Langevin dynamics or gradient descent, during training; although this allows fast training, convergence for unseen samples is not guaranteed and there is still a large discrepancy between the true posterior and latent approximations due to lag in optimisation [26; 125]. Short-run MCMC has also been applied however it has poor mixing properties [261].

VAEBMs offer a different perspective, rather than performing latent MCMC sampling based on the ELBO, they use an auxiliary energy-based model to correct blurry VAE samples, with MCMC sampling performed in both the data space and latent space. This setup is defined by $h_{\phi,\theta}(\boldsymbol{x}, \boldsymbol{z}) = \frac{1}{Z_{\phi,\theta}} p_\theta(\boldsymbol{z}) p_\theta(\boldsymbol{x}|\boldsymbol{z}) e^{-E_\phi(\boldsymbol{x})}$, where $p_\theta(\boldsymbol{z})p_\theta(\boldsymbol{x}|\boldsymbol{z})$ is the VAE, and $E_\phi(\boldsymbol{x})$ is the energy model. This, however, requires 2 stages of training to avoid calculating the gradient of the normalising constant $Z_{\phi,\theta}$, training only the VAE and fixing the VAE and training the EBM respectively.

2.4 Generative Adversarial Networks

Another approach at eliminating the Markov chains used in energy models is the generative adversarial network (GAN) (Fig. 2.6) [101]. GANs consist of two networks, a discriminator $D\colon \mathbb{R}^n \to [0, 1]$ which estimates the probability that a sample comes from the data distribution $\boldsymbol{x} \sim p_d(\boldsymbol{x})$, and a generator $G\colon \mathbb{R}^m \to \mathbb{R}^n$ which given a latent variable $\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})$, captures p_d by tricking the discriminator into thinking its samples are real. This is achieved through adversarial training of the networks: D is trained to correctly label training samples as real and samples from G as fake, while G is trained to minimise the probability that D classifies its samples as fake. This can be interpreted as D and G playing a mini-max game, as with prior work [316; 317], optimising the value function $V(G, D)$,

$$\min_G \max_D V(G, D) = \mathbb{E}_{\boldsymbol{x}\sim p_d(\boldsymbol{x})}[\ln D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\ln(1 - D(G(\boldsymbol{z})))]. \tag{2.30}$$

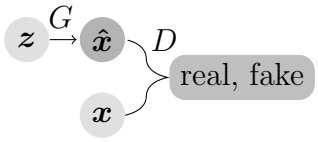


Figure 2.6: Generative adversarial networks set two networks in a game: D detects real from fake samples while G tricks D.

For a fixed G, the objective for D can be reformulated as

$$\begin{aligned}
&\max_D V(G,D) = \mathbb{E}_{\boldsymbol{x}\sim p_d}[\ln D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\ln(1-D(\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_d}\left[\ln \frac{p_d(\boldsymbol{x})}{p_d(\boldsymbol{x})+p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x}\sim p_g}\left[\ln \frac{p_g(\boldsymbol{x})}{p_d(\boldsymbol{x})+p_g(\boldsymbol{x})}\right] \\
&= D_{KL}(p_d \parallel \tfrac{1}{2}(p_d+p_g)) + D_{KL}(p_g \parallel \tfrac{1}{2}(p_d+p_g)) + C.
\end{aligned} \tag{2.31}$$

Therefore the loss is equivalent to the Jensen-Shannon divergence between the generative distribution p_g and the data distribution p_d and thus with sufficient capacity, the generator can recover the data distribution. The use of symmetric JS-divergence is well behaved when both distributions are small unlike the asymmetric KL-divergence used in maximum likelihood models. Additionally, it has been suggested that reverse KL-divergence, $D_{KL}(p_g||p_d)$, is a better measure for training generative models than normal KL-divergence, $D_{KL}(p_d||p_g)$, since it minimises $\mathbb{E}_{\boldsymbol{x}\sim p_g}[\ln p_d(\boldsymbol{x})]$ [156]; while reverse KL-divergence is not a viable objective function, JS-divergence is and behaves more like reverse KL-divergence than KL-divergence alone. With that said, JS-divergence is not perfect; if 0 mass is associated with a data sample in a maximum likelihood model, KL-divergence is driven to infinity, whereas this can happen with no consequence in a GAN.

2.4.1 Stabilising Training

The adversarial nature of GANs makes them notoriously difficult to train [7]; Nash equilibrium is hard to achieve [309] since non-cooperation cannot guarantee convergence, thus training often results in oscillations of increasing amplitude. As the discriminator improves, gradients passed to the generator vanish, accelerating this problem; on the other hand, if the discriminator remains poor, the generator does

not receive useful gradients. Another problem is mode collapse, where one network gets stuck in a bad local minima and only a small subset of the data distribution is learned. The discriminator can also jump between modes resulting in catastrophic forgetting, where previously learned knowledge is forgotten when learning something new [337]. This section explores proposed solutions to these problems.

Loss Functions

Since the cause of many of these issues can be linked with the use of JS-divergence, other loss functions have been proposed that minimise other statistical distances; in general, any f-divergence can be used to train GANs [264]. One notable example is the Wasserstein distance which intuitively indicates how much "mass" must be moved to transform one distribution into another. Wasserstein distance is defined formally in Eqn. 2.32a, which by the Kantorovich-Rubinstein duality is equivalent to Eqn. 2.32b [367]:

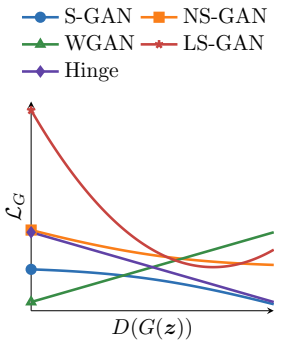
$$W(p_d, p_g) = \inf_{\gamma \in \prod(p_d, p_g)} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\gamma}[\|\boldsymbol{x} - \boldsymbol{y}\|], \tag{2.32a}$$

$$W(p_d, p_g) = \sup_{\|D\|_L \leq 1} \mathbb{E}_{\boldsymbol{x}\sim p_d}[D(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim p_g}[D(\boldsymbol{x})], \tag{2.32b}$$

where the supremum is taken over all 1-Lipschitz functions, that is, f such that for all x_1 and x_2, $\left\|f(x_1) - f(x_2)\right\|_2 \leq \|x_1 - x_2\|_2$. Optimising Wasserstein distance, as described in Tab. 2.7a, offers linear gradients thus eliminating the vanishing gradients problem (see Fig. 2.7b). Moreover, Wasserstein distance is also equivalent to minimising reverse KL-divergence [248], offers improved stability, and allows training to optimality. Numerous approaches to enforce 1-Lipschitz continuity have been proposed: weight clipping [8] invalidates gradients making optimisation difficult; applying a gradient penalty within the loss is heavily dependent on the support of the generative distribution and computation with finite samples makes application to the entire space intractable [118]; spectral normalisation (discussed below) applies global regularisation by estimating the singular values of parameters. Other popular loss functions include least squares GAN, hinge loss, energy-based GAN, and relativistic GAN (detailed in Tab. 2.7a). These can be visualised in Fig. 2.7b;

Name	Losses
NSGAN [101]	$\mathcal{L}_d = -\mathbb{E}[\ln(\sigma(D(\boldsymbol{x})))] - \mathbb{E}[\ln(1-\sigma(D(G(\boldsymbol{z}))))]$
	$\mathcal{L}_g = -\mathbb{E}[\ln(\sigma(D(G(\boldsymbol{z}))))]$
WGAN [8]	$\mathcal{L}_d = \mathbb{E}[D(\boldsymbol{x})] - \mathbb{E}[D(G(\boldsymbol{z}))]$
	$\mathcal{L}_g = \mathbb{E}[D(G(\boldsymbol{z}))]$
LSGAN [237]	$\mathcal{L}_d = \mathbb{E}[(D(\boldsymbol{x})-1)^2] + \mathbb{E}[D(G(\boldsymbol{z}))^2]$
	$\mathcal{L}_g = \mathbb{E}[(D(G(\boldsymbol{z}))-1)^2]$
Hinge [215]	$\mathcal{L}_d = \mathbb{E}[\min(0, D(\boldsymbol{x})-1)] - \mathbb{E}[\max(0, 1+D(G(\boldsymbol{z})))]$
	$\mathcal{L}_g = -\mathbb{E}[D(G(\boldsymbol{z}))]$
EBGAN [398]	$\mathcal{L}_d = D(\boldsymbol{x}) + \max(0, m - D(G(\boldsymbol{z})))$
	$\mathcal{L}_g = D(G(\boldsymbol{z}))$
RSGAN [168]	$\mathcal{L}_d = \mathbb{E}[\ln(\sigma(D(\boldsymbol{x}) - D(G(\boldsymbol{z}))))]$
	$\mathcal{L}_g = \mathbb{E}[\ln(\sigma(-D(G(\boldsymbol{z})) - D(\boldsymbol{x})))]$

(a) GAN losses.



(b) Generator loss functions.

Figure 2.7: A comparison of popular losses used to train GANs. (a) Respective losses for discriminator/generator. (b) Plots of generator losses with respect to discriminator output. Notably, NS-GAN's gradient disappears as discriminator gets better.

notably, least squares GAN (LS-GAN) penalises samples that lie far from the decision boundary, whether they are correctly classified or not; hinge loss is based on SVM separating hyperplanes that has the maximal margin between the two classes, it has been found to oscillate less than other losses.

The catastrophic forgetting problem can be mitigated by conditioning the GAN on class information, encouraging more stable representations [29; 245; 392]. Nevertheless, labelled data, if available, only covers limited abstractions. Self-supervision achieves the same goal by training the discriminator on an auxiliary classification task based solely on the unsupervised data. Proposed approaches are based on randomly rotating inputs to the discriminator, which learns to identify the angle rotated separately to the standard real/fake classification [45]. Extensions include training the discriminator to jointly determine rotation and real/fake to provide better feedback [349], and training the generator to trick the discriminator at both the real/fake and classification tasks [349]. A more explicit approach is to model the generator with a normalizing flow, avoiding collapse by jointly optimising the GAN and likelihood objectives [115].

Spectral Normalisation

Spectral normalisation [248] is a technique to make a function globally 1-Lipschitz utilising the observation that the Lipschitz constant of a linear function is its largest singular value (spectral norm). The spectral norm of a matrix $\boldsymbol{A}$ is

$$SN(\boldsymbol{A}) := \max_{\boldsymbol{h}:\boldsymbol{h}\neq\boldsymbol{0}} \frac{\|\boldsymbol{A}\boldsymbol{h}\|_2}{\|\boldsymbol{h}\|_2} = \max_{\|\boldsymbol{h}\|_2\leq 1} \|\boldsymbol{A}\boldsymbol{h}\|_2 \,, \tag{2.33}$$

thus a weight matrix $\boldsymbol{W}$ is normalised to be 1-Lipschitz by replacing the weights with $\boldsymbol{W}_{SN} := \frac{\boldsymbol{W}}{SN(\boldsymbol{W})}$. Rather than using singular value decomposition to compute the norm, the power iteration method is used; for randomly initialised vectors $\boldsymbol{v} \in \mathbb{R}^n$ and $\boldsymbol{u} \in \mathbb{R}^m$, the procedure is

$$\boldsymbol{u}_{t+1} = \boldsymbol{W}\boldsymbol{v}_t,\ \boldsymbol{v}_{t+1} = \boldsymbol{W}^T\boldsymbol{u}_{t+1},\ SN(\boldsymbol{W}) \approx \boldsymbol{u}^T\boldsymbol{W}\boldsymbol{v}. \tag{2.34}$$

Since weights change only marginally with each optimisation step, a single power iteration step per global optimisation step is sufficient to keep $\boldsymbol{v}$ and $\boldsymbol{u}$ close to their targets.

As aforementioned, enforcing the discriminator to be 1-Lipschitz is essential for WGANs; however, spectral normalisation has been found to dramatically improve sample quality and allow scaling to datasets with thousands of classes across a variety of loss functions [29; 248]. Spectral collapse, has been linked to discriminator overfitting when spectral norms of layers explode [29] as well as mode collapse when spectral norms fall in value significantly [221]. Additionally, regularising the discriminator in this manner helps balance the two networks, reducing the number of discriminator update steps required [29; 392].

Data Augmentation

Augmenting training data to increase the quantity of training data is often common practice; when training GANs the types of augmentations permitted are limited to more simple augmentations such as cropping and flipping to prevent the generator from creating undesired artefacts. Several approaches independently proposed

applying augmentations to all discriminator inputs, allowing more substantial augmentations to be used [180; 348; 402; 403]; the training procedure for a WGAN with augmentations is

$$\mathcal{L}_D = \mathbb{E}_{\boldsymbol{x}\sim p_d(\boldsymbol{x})}[D(T(\boldsymbol{x}))] - \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[D(T(G(\boldsymbol{z})))], \tag{2.35a}$$

$$\mathcal{L}_G = \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[D(T(G(\boldsymbol{z})))], \tag{2.35b}$$

where T is a random augmentation. These approaches have been shown to improve sample quality on equivalent architectures and stabilise training. Each work offers a different perspective on why augmentation is so effective: the increased quantity of training data in conjunction with the more difficult discrimination task prevents overfitting and in turn collapse [29], notably this applies even on very small datasets (100 samples); the nature of GAN training leads to the generated and data distributions having non-overlapping supports, complicating training [334], strong augmentations may cause these distributions to overlap further. If an augmentation is differentiable and represents an invertible transformation of the data space's distribution, then the JS-divergence is invariant, and the generator is guaranteed to not create augmented samples [180; 348].

Discriminator Driven Sampling

In order to improve sample quality and address overpowered discriminators, numerous works have taken inspiration from the connection between GANs and energy models [398]. Interpreting the discriminator of a Wasserstein GAN [8] as an energy-based model means samples from the generator can be used to initialise an MCMC sampling chain which converges to the density learned by the discriminator, correcting errors learned by the generator [254; 353]. This is similar to pure EBM approaches; however, training the two networks adversarially changes the dynamics. The slow convergence rates of high dimensional MCMC sampling has led others to instead sample in the latent space [40; 331].

GANs without Competition

Originally proposed as a proxy to measure GAN convergence [113], the duality gap is an upper bound on the JS-divergence that can be directly optimised [114], defined as

$$DG(D, G) = \max_{D'} V(G, D') - \min_{G'} V(G', D). \tag{2.36}$$

Cooperative training simplifies the optimisation procedure, avoiding oscillations. Each training step, however, requires optimising for D' and G' which slows down training and could suffer from vanishing gradients.

2.4.2 Architectures

Careful network design is a key component for stable GAN training. Scaling any deep neural network to high-resolution data is non-trivial due to vanishing gradients and high memory usage, but since the discriminator can classify high-resolution data more easily, GANs notably struggle [266].

Early approaches designed hierarchical architectures, dividing the learning procedure into more easily learnable chunks. LapGAN [64] builds a Laplacian pyramid such that at each layer, a GAN conditioned on the previous image resolution predicts a residual adding detail. Stacked GANs [154; 390] use two GANs trained successively: the first generates low-resolution samples, then the second upsamples and corrects the first, thus fewer GANs need to be trained. A related approach, progressive growing [178; 179], iteratively trains a single GAN at higher resolutions by adding layers to both the generator and discriminator upscaling the previous output, after the previous resolution converges. Training in this manner, however, not only takes a long time but leads to high frequency components being learned in the lower layers, resulting in shift artefacts [181].

Accordingly, a number of works have targeted a single GAN that can be trained end-to-end. DCGAN [283] introduced a fully convolutional architecture with batch normalisation [159] and ReLU/LeakyReLU activations. BigGAN [29] employ a number of tricks to scale to high resolutions including using very large mini-batches to reduce variation, spectral normalisation to discourage spectral collapse, and using

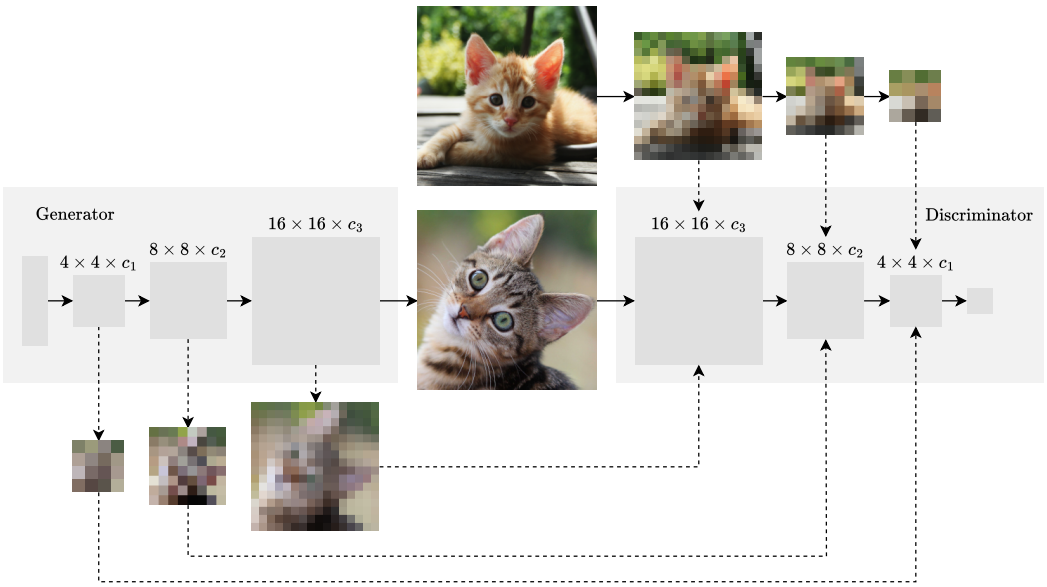


Figure 2.8: A GAN with skip connections between the generator and discriminator to improve gradient flow. Dashed lines are 1×1 convolutions for mapping the generator's activations to image channels (when discriminating generated images); and to inject low resolution image features into the discriminator (when discriminating real images) [175; 181; 358].

large datasets to prevent overfitting. Despite this, training collapse still occurs thus requiring early stopping. Another approach is to include skip connections between the generator and discriminator at each resolution, allowing gradients to flow through shorter paths to each layer, providing extra information to the generator (Fig. 2.8) [175; 181; 358]. By treating subsets of the generator's parameters as smaller generators, Anycost GANs extend this approach, allowing samples to be generated at multiple resolutions and speeds[218]. To learn long-range dependencies, GANs can be built with self-attention components [164; 364; 392]; however, full quadratic attention does not scale well to high dimensional data.

2.4.3 Training Speed

The mini-max nature of GAN training leads to slow convergence, if achieved at all. This problem has been exacerbated by numerous works as a byproduct of improving stability or sample quality. One such example is that by using very large mini-batches, reducing variance and covering more modes, sample quality can be improved

significantly; however, this comes at the cost of slower training [29]. Small-GAN [320] combats this by replacing large batches with small batches that approximate the shape of the larger batch using core set sampling [320], significantly improving the mode coverage and sample quality of GANs trained with small batches.

While strong discriminator regularisation stabilises training, it allows the generator to make small changes and trick the discriminator, making convergence very slow. Rob-GAN [223], include an adversarial attack step [235] that perturbs real images to trick the discriminator without altering the content inordinately, adapting the GAN objective into a min-max-min problem. This provides a weaker regularisation, enforcing small Lipschitz values locally rather than globally. This approach has been connected with the follow-the-ridge algorithm [372; 404], an optimisation approach for solving mini-max problems that reduces the optimisation path and converges to local mini-max points.

Another approach to improve training speed is to design more efficient architectures. Depthwise convolutions [50] apply separate convolutions to each channel of a tensor reducing the number of operations and hence also the runtime, have been found to have comparable quality to standard convolutions [255]. Lightweight GANs [220] achieve fast training using a number of tricks including small batch sizes, skip-layer excitation modules which provide efficient shortcut gradient flow, as well as using a self-supervised discriminator forcing good features to be learned.

2.5 Autoregressive Likelihood Models

Autoregressive generative models (Fig. 2.9) [19] are based on the chain rule of probability, where the probability of a variable that can be decomposed as $\boldsymbol{x} = x_1, \ldots, x_n$ is expressed as

$$p(\boldsymbol{x}) = p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1}). \tag{2.37}$$

As such, unlike GANs and energy models, it is possible to directly maximise the likelihood of the data by training a recurrent neural network to model $p(x_i | \boldsymbol{x}_{1:i-1})$

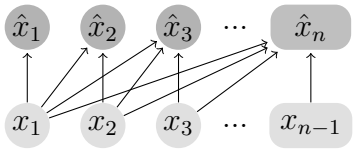


Figure 2.9: Autoregressive models decompose data points using the chain rule and learn conditional probabilities.

by minimising the negative log-likelihood,

$$-\ln p(\boldsymbol{x}) = -\sum_{i}^{n} \ln p(x_i|x_1, \ldots, x_{i-1}). \tag{2.38}$$

While autoregressive models are extremely powerful density estimators, sampling is inherently a sequential process and can be exceedingly slow on high dimensional data. Additionally, data must be decomposed into a fixed ordering; while the choice of ordering can be clear for some modalities (e.g. text and audio), it is not obvious for others such as images and can affect performance depending on the network architecture used.

2.5.1 Architectures

The majority of research is focused on improving network architectures to increase their receptive fields and memory, ensuring the network has access to all parts of the input to encourage consistency, as well as increasing the network capacity, allowing more complex distributions to be modelled.

Masked Multilayer Perceptrons

One approach to build autoregressive models is to mask the weights of simple multilayer perceptron (MLP) autoencoders so as to satisfy the autoregressive property. The neural autoregressive density estimator (NADE) [204], which can be viewed as a mean-field approximation of a restricted Boltzmann machine, achieves this for binary data by placing time-dependent masks on an MLP with one hidden layer. Specifically, at time step i, weights are masked so that the entire hidden state $\boldsymbol{h}_i$

and output $p(x_i|\boldsymbol{x}_{<i})$ are dependent only on $\boldsymbol{x}_{<i}$; formally this can be defined as

$$p(x_i = 1|\boldsymbol{x}_{<i}) = \sigma(b_i + (\boldsymbol{W}^T)_{i,\cdot}\boldsymbol{h}_i), \tag{2.39a}$$

$$\boldsymbol{h}_i = \sigma(\boldsymbol{c} + \boldsymbol{W}_{\cdot,<i}\boldsymbol{x}_{<i}), \tag{2.39b}$$

where $\boldsymbol{W}_{\cdot,<d}$ is the first $d-1$ columns of a shared weight matrix $\boldsymbol{W}$, and b_i and $\boldsymbol{c}$ are biases. The RNADE [355] generalises NADE to real valued data by instead modelling $p(x_i|\boldsymbol{x}_{<i})$ with mixture distributions parameterised by the network. An alternative masking procedure known as MADE [96] allows for parallel density estimation by placing a mask fixed over time on an MLP so that no connections exist between $p(x_i|\boldsymbol{x}_{<i})$ and $\boldsymbol{x}_{\geq i}$. Additionally, MADE is more readily vectorisable and does not suffer from neuron saturation since the number of inputs to all neurons is constant with respect to time.

Recurrent Neural Networks

A natural architecture to apply is that of standard recurrent neural networks (RNNs) such as LSTMs [138; 338; 361] and GRUs [53; 238] which model sequential data by tracking information in a hidden state. However, RNNs are known to forget information, limiting their receptive field thus preventing modelling of long range relationships. This can be improved by stacking RNNs that run at different frequencies allowing long data such as multiple seconds of audio to be modelled [53]. Nevertheless, their sequential nature means that training can be too slow for many tasks.

Causal Convolutions

An alternative approach is that of causal convolutions, which apply masked or shifted convolutions over a sequence [47; 310; 360]. When stacked, this only provides a receptive field linear with depth; however, by dilating the convolutions to skip values with some step the receptive field can be orders of magnitude higher.

Self-Attention

Neural attention is an approach which at each successive time step is able to select where it wishes to 'look' at previous time steps. This concept has been used to autoregressively 'draw' images onto a blank 'canvas' [112] in a manner similar to human drawing. More recently self-attention (known as Transformers when used in an encoder-decoder setup) [364] has made significant strides improving not only autoregressive models, but also other generative models due to its parallel nature, stable training, and ability to effectively learn long-distance dependencies. This is achieved using an attention scheme that can reference any previous input where an entirely independent process is used per time step so that there are no dependencies. Specifically, inputs are encoded as key-value pairs, where the values $\boldsymbol{V}$ represent the inputs, and the keys $\boldsymbol{K}$ act as an indexing method. At each time step a query q is made; taking the dot product of the queries and keys, a similarity vector is formed that describes which value vectors to access. This process can be expressed as

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V}, \tag{2.40}$$

where d_k is the key/query dimension and is used to normalise gradient magnitudes. Since the self-attention process contains no recurrence, positional information must be passed into the function. A simple effective method to achieve this is to add sinusoidal positional encodings which combine sine and cosine functions of different frequencies to encode positional information [364]; alternatively others use trainable positional embeddings [49].

The infinite receptive fields of attention provides a powerful tool for representing data; however, the attention matrix $\boldsymbol{Q}\boldsymbol{K}^T$ grows quadratically with data dimension, making scaling difficult. Approaches include scaling across large quantities of GPUs [30], interleaving attention between causal convolutions [47], attending over local regions [273], and using sparse attention patterns that provide global attention when multiple layers are stacked [49]. More recently, a number of linear transformers have been proposed whose memory and time footprints grow linearly with data dimension [51; 183; 371]. By approximating the softmax operation with a kernel function with

feature representation $\phi(\boldsymbol{x})$, the order of multiplications can be rearranged to

$$\big(\phi(\boldsymbol{Q})\phi(\boldsymbol{K})^T\big)\boldsymbol{V} = \phi(\boldsymbol{Q})\big(\phi(\boldsymbol{K})^T\boldsymbol{V}\big), \tag{2.41}$$

allowing $\phi(\boldsymbol{K})^T\boldsymbol{V}$ to be cached and used for each query.

Multiscale Architectures

Even with a linear autoregressive model, $O(N)$ for N pixels, scaling to high-resolution images grows quadratically with resolution. One multi-scale approach reduces this complexity to $O(\ln N)$ by successively upscaling images, making the assumption that when upscaling, each pixel is dependent only on its adjacent area and the previous resolution image, allowing scaling to high resolutions [292]. To avoid making independence assumptions, [241] partition images in an interleaving pattern so that sub-images are the same size and capture global structure. Sub-images are generated autoregressively pixel-wise and are conditioned on previously generated sub-images; while this reduces the memory required, sampling times are still slow.

2.5.2 Data Modelling Decisions

When generating text, output variables are often modelled using a multinomial distribution since tokens are discrete and are in general unrelated. However, this modelling assumption can cause complications or be infeasible in other cases such as 16-bit audio modelling, in which magnitude would not be intrinsically modelled and 65,536 output neurons would be required. Solutions proposed include:

- Applying μ-law, a logarithmic companding algorithm which takes advantage of human perception of sound, then quantizing to 8-bit values [359].
- First predicting the first 8-bits, then predicting the second 8-bits conditioned on the first.
- Modelling output probabilities using a mixture of logistic distributions (MoL) has the benefits of providing more useful gradients and allowing intensities never seen to still be sampled [310].

Nevertheless, these assumptions restrict the expressiveness of the network, for instance, MoLs struggle to model high frequency signals as found in raw image data; a simple solution in this case is to add Gaussian noise, reducing the Lipschitz constant of the data distribution [240]. This restriction can be removed at the expense of less efficient sampling by learning an autoregressive energy model, for instance, by approximating normalising constants [252] or through score matching [239]. Alternatively, quantile regression, which minimises Wasserstein distance, can be used to learn an approximation of the inverse cumulative distribution [268].

When modelling images, many works use "raster scan" ordering [310; 360; 361] where pixels are estimated row by row. Alternatives have been proposed such as "zig-zag" ordering [47] which allows pixels to depend on previously sampled pixels to the left and above, providing more relevant context. Another factor when modelling images is how to factorise sub-pixels. While it is possible to treat them as independent variables, this adds additional complexity. Alternatively, it is possible to instead condition on whole pixels, and output joint distributions in a single step [310].

2.6 Normalizing Flows

While training autoregressive models through maximum likelihood offers plenty of benefits including stable training, density estimation, and a useful validation metric, the slow sampling speed and poor scaling properties handicaps them significantly. Normalizing flows are a technique that also allows exact likelihood calculation while being efficiently parallelisable as well as offering a useful latent space for downstream tasks. Consider an invertible, smooth function $f\colon \mathbb{R}^d \to \mathbb{R}^d$; by applying this transformation to a random variable $\boldsymbol{x} \sim p(\boldsymbol{x})$, then the distribution of the resulting random variable $\boldsymbol{y} = f(\boldsymbol{x})$ can be determined through the change of variables rule (and application of the chain rule),

$$p(\boldsymbol{y}) = p(\boldsymbol{x}) \left| \det \frac{\partial f^{-1}}{\partial \boldsymbol{y}} \right| = p(\boldsymbol{x}) \left| \det \frac{\partial f}{\partial \boldsymbol{x}} \right|^{-1}. \tag{2.42}$$

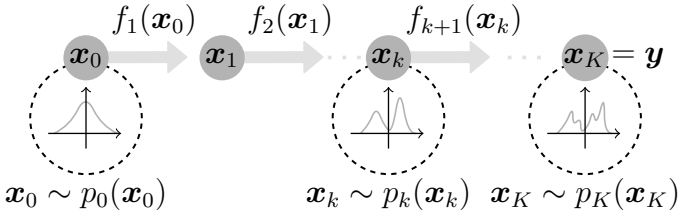


Figure 2.10: Normalizing flows build complex distributions by mapping a simple distribution through invertible functions.

Consequently, arbitrarily complex densities can be constructed by composing simple maps and applying Eqn. 2.42 [366]. This chain is known as a normalizing flow [293] (see Fig. 2.10). The density $p_K(\boldsymbol{x}_K)$ obtained by successively transforming a random variable $\boldsymbol{x}_0$ with distribution p_0 through a chain of K transformations f_k can be defined as

$$\boldsymbol{x}_K = f_K \circ \cdots \circ f_2 \circ f_1(\boldsymbol{x}_0), \tag{2.43a}$$

$$\ln p_K(\boldsymbol{x}_K) = \ln p_0(\boldsymbol{x}_0) - \sum_{k=1}^{K} \ln \left| \det \frac{\partial f_k}{\partial \boldsymbol{x}_{k-1}} \right|. \tag{2.43b}$$

Each transformation therefore must be sufficiently expressive while being easily invertible and have an efficient way to compute Jacobian determinant. While restrictive, there have been a number of works which have introduced more powerful invertible functions (see Tab. 2.3). Nevertheless, normalizing flow models are typically less parameter efficient than other generative models.

One disadvantage of requiring transformations to be invertible is that the input dimension must be equal to the output dimension which makes deep models inefficient and difficult to train. A popular solution to this is to use a multi-scale architecture [69; 191] (see Fig. 2.11) which divides the process into a number of stages, at the end of each half of the remaining units are factored out and treated immediately as outputs. This allows latent variables to sequentially represent course to fine features and permits deeper architectures.

Table 2.3: Normalizing Flow Layers: $\odot$ represents elementwise multiplication, $\star_l$ represents a cross-correlation layer

Description	Function	Inverse Function	Log-Determinant
Low Rank			
Planar [293]	$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{u}h(\boldsymbol{w}^T\boldsymbol{z} + b)$ With $\boldsymbol{w} \in \mathbb{R}^D$, $\boldsymbol{u} \in \mathbb{R}^D$, $\boldsymbol{b} \in \mathbb{R}$	No closed form inverse	$\ln\lvert 1 + \boldsymbol{u}^T h'(\boldsymbol{w}^T\boldsymbol{z} + b)\boldsymbol{w}\rvert$
Sylvester [21; 126]	$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{U}h(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b})$	No closed form inverse	$\ln\det(\boldsymbol{I}_M + \text{diag}(h'(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b}))\boldsymbol{W}\boldsymbol{U}^T)$
Coupling/Autoregressive			
General Coupling	$\boldsymbol{y}^{(1:d)} = \boldsymbol{x}^{(1:d)}$ $\boldsymbol{y}^{(d+1:D)} = h(\boldsymbol{x}^{(d+1:D)}; f_\theta(\boldsymbol{x}^{(1:d)}))$	$\boldsymbol{x}^{(1:d)} = \boldsymbol{y}^{(1:d)}$ $\boldsymbol{x}^{(d+1:D)} = h^{-1}(\boldsymbol{y}^{(d+1:D)}; f_\theta(\boldsymbol{y}^{(1:d)}))$	$\ln\lvert\det \nabla_{\boldsymbol{x}^{(d+1:D)}} h\rvert$
MAF [270]	$y^{(t)} = h(x^{(t)}; f_\theta(\boldsymbol{x}^{(1:t-1)}))$	$x^{(t)} = h^{-1}(y^{(t)}; f_\theta(\boldsymbol{x}^{(1:t-1)}))$	$-\sum_{t=1}^{D} \ln\lvert\frac{\partial y^{(t)}}{\partial x^{(t)}}\rvert$
IAF [192]	$y^t = h(x^{(t)}; f_\theta(\boldsymbol{y}^{(1:t-1)}))$	$x^t = h^{-1}(y^{(t)}; f_\theta(\boldsymbol{y}^{(1:t-1)}))$	$\sum_{t=1}^{D} \ln\lvert\frac{\partial y^{(t)}}{\partial x^{(t)}}\rvert$
Affine Coupling [69]	$h(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{x} \odot \exp(\boldsymbol{\theta}_1) + \boldsymbol{\theta}_2$	$h^{-1}(\boldsymbol{y}; \boldsymbol{\theta}) = (\boldsymbol{y} - \boldsymbol{\theta}_2) \odot \exp(-\boldsymbol{\theta}_1)$	$\sum_{i=1}^{d} \theta_1^{(i)}$
Flow++ [134]	$h(\boldsymbol{x}; \boldsymbol{\theta}) = \exp(\boldsymbol{\theta}_1) \odot F(\boldsymbol{x}, \boldsymbol{\theta}_3) + \boldsymbol{\theta}_2$ where F is a monotone function.	Calculated through bisection search	$\sum_{i=1}^{d} \theta_1^{(i)} + \ln \frac{\partial F(\boldsymbol{x}, \boldsymbol{\theta}_3)_i}{\partial x_i}$
Spline Flows [249][82][81]	$h(\boldsymbol{x}; \boldsymbol{\theta}) = \text{Spline}(\boldsymbol{x}; \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are the spline's knots.	$h^{-1}(\boldsymbol{y}; \boldsymbol{\theta}) = \text{Spline}^{-1}(\boldsymbol{y}; \boldsymbol{\theta})$	Computed in closed-form as a product of quotient derivatives
B-NAF [63]	$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}^T$ for blocked weights: $\boldsymbol{W} = \exp(\tilde{\boldsymbol{W}}) \odot \boldsymbol{M}_d + \tilde{\boldsymbol{W}} \odot \boldsymbol{M}_o$ where $\boldsymbol{M}_d$ selects diagonal blocks and $\boldsymbol{M}_o$ selects off-diagonal blocks.	No closed form inverse	$\ln \sum_{i=1}^{d} \exp(\tilde{W}_{ii})$
Convolutions			
1x1 Convolution [191]	$h \times w \times c$ tensor $\boldsymbol{x}$ & $c \times c$ tensor $\boldsymbol{W}$ $\forall i, j : \boldsymbol{y}_{i,j} = \boldsymbol{W}\boldsymbol{x}_{i,j}$	$\forall i, j : \boldsymbol{x}_{i,j} = \boldsymbol{W}^{-1}\boldsymbol{y}_{i,j}$	$h \cdot w \cdot \ln\lvert\det \boldsymbol{W}\rvert$
Emerging Convolutions [142]	$\boldsymbol{k} = \boldsymbol{w}_1 \odot \boldsymbol{m}_1, \quad \boldsymbol{g} = \boldsymbol{w}_2 \odot \boldsymbol{m}_2$ $\boldsymbol{y} = \boldsymbol{k} \star_l (\boldsymbol{g} \star_l \boldsymbol{x})$	$\boldsymbol{z}_t = (\boldsymbol{y}_t - \sum_{i=t+1} G_{t,i}\boldsymbol{z}_i)/G_{t,t}$ $\boldsymbol{x}_t = (\boldsymbol{z}_t - \sum_{i=1}^{t-1} K_{t,i}\boldsymbol{x}_i)/K_{t,t}$	$\sum_c \ln\lvert\boldsymbol{k}_{c,c,m_y,m_x}\boldsymbol{g}_{c,c,m_y,m_x}\rvert$
Lipshitz Residual			
i-ResNet [17]	$\boldsymbol{y} = \boldsymbol{x} + f(\boldsymbol{x})$ where $\lVert f\rVert_L < 1$	$\boldsymbol{x}_1 = \boldsymbol{y}. \quad \boldsymbol{x}_{n+1} = \boldsymbol{y} - f(\boldsymbol{x}_n)$ converging at an exponential rate	$\text{tr}(\ln(\boldsymbol{I} + \nabla_{\boldsymbol{x}} f)) =$ $\sum_{k=1}^{\infty}(-1)^{k+1}\frac{\text{tr}((\nabla_{\boldsymbol{x}} f)^k)}{k}$

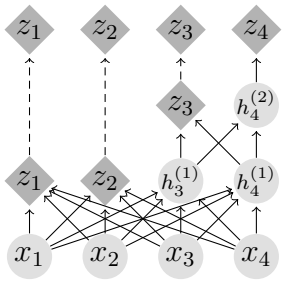


Figure 2.11: Factoring out variables at different scales allows normalizing flows to scale to high dimensional data.

2.6.1 Coupling and Autoregressive Layers

A simple way of building an expressive invertible function is the coupling flow [68], which divide inputs into two and applies a bijection h on one half parameterised by the other,

$$\boldsymbol{y}^{(1:d)} = \boldsymbol{x}^{(1:d)}, \tag{2.44a}$$

$$\boldsymbol{y}^{(d+1:D)} = h(\boldsymbol{x}^{(d+1:D)}; f_\theta(\boldsymbol{x}^{(1:d)})), \tag{2.44b}$$

here f can be arbitrarily complex i.e. a neural network. h tends to be selected as an elementwise function making the Jacobian triangular allowing efficient computation of the determinant, i.e. the product of elements on the diagonal.

Affine Coupling

A simple example of this is the affine coupling layer [69],

$$\boldsymbol{y}^{(d+1:D)} = \boldsymbol{x}^{(d+1:D)} \odot \exp(f_\sigma(\boldsymbol{x}^{(1:d)})) + f_\mu(\boldsymbol{x}^{(1:d)}), \tag{2.45}$$

which has a simple Jacobian determinant and can be trivially rearranged to obtain a definition of $\boldsymbol{x}^{(d+1:D)}$ in terms of $\boldsymbol{y}$, provided that the scaling coefficients are not 0. This simplicity, however, comes at the cost of expressivity; while stacking numerous such flows increases their expressivity, allowing them to learn representations of complex high dimensional data such as images [191], it is unknown whether multiple affine flows are universal approximators [271].

Monotone Functions

Another method of creating invertible functions that can be applied element-wise is to enforce monotonicity. One possibility to achieve this is to define h as an integral over a positive but otherwise unconstrained function g [374],

$$h(x_i; \boldsymbol{\theta}) = \int_0^{x_i} g_\phi(x; \boldsymbol{\theta}_1) dx + \theta_2, \tag{2.46}$$

however, this integration requires numerical approximation. Alternatively, by choosing g to be a function with a known integral solution, h can be efficiently evaluated. This has been accomplished using positive polynomials [162] and the CDF of a mixture of logits [134]. Both cases, however, don't have analytical inverses and have to be approximated iteratively with bisection search. Another option is to represent g as a monotonic spline: a piecewise function where each piece is easy to invert. As such, the inverse is as fast to evaluate as the forward pass. Linear and quadratic splines [249], cubic splines [82], and rational-quadratic splines [81] have been applied so far.

Autoregressive Flows

For a single coupling layer, a significant proportional of inputs remain unchanged. A more flexible generalisation of coupling layers is the autoregressive flow, or MAF [270],

$$y^{(t)} = h(x^{(t)}; f_\theta(\boldsymbol{x}^{(1:t-1)})). \tag{2.47}$$

Here f_θ can be arbitrarily complex, allowing the use of advances in autoregressive modelling (Section. 2.5), and h is a bijection as used for coupling layers. Some monotonic bijectors have been created specifically for autoregressive flows, namely Neural Autoregressive Flows (NAF) [150] and Block NAF [63]. Unlike coupling layers, a single autoregressive flow is a universal approximator.

Alternatively, an autoregressive flow can be conditioned on $\boldsymbol{y}^{(1:t-1)}$ rather than $\boldsymbol{x}^{(1:t-1)}$, this is known as an Inverse Autoregressive Flow, or IAF [192]. While coupling layers can be evaluated efficiently in both directions, MAF permits parallel density estimation but sequential sampling, and IAF permits parallel sampling but

sequential density estimation.

Probability Density Distillation

Inverse autoregressive flows [192] offer the ability to sample from an autoregressive model in parallel; however, training via maximum likelihood is inherently sequential making this infeasible for high dimensional data. Probability density distillation [363] has been proposed as a solution to this where a second pre-trained autoregressive network is used as a 'teacher' network while an IAF network is used as a 'student' and mimics the teacher's distribution by minimising the KL divergence between the two distributions:

$$D_{KL}(p_S||p_T) = H(p_S, p_T) - H(p_S), \tag{2.48}$$

where p_S and p_T are the student's and teacher's distributions respectively, $H(p_S, p_T)$ is the cross-entropy between p_S and p_T, and $H(p_S)$ is the entropy of p_S. Crucially, this never requires the student's inverse function to be used allowing it to be computed entirely in parallel.

2.6.2 Convolutional Flows

A considerable problem with coupling and autoregressive flows is the restricted triangular Jacobian, meaning that all inputs cannot interact with each other. Simple solutions involve fixed permutations on the output space such as reversing the order [68; 69]. A more general approach is to use a 1×1 convolution which is equivalent to a linear transformation applied across channels [191]. Numerous works have been proposed to generalise these to larger kernel sizes. A number of these apply variations on causal convolutions [359], including emerging convolutions [142] whose inverse is sequential, MaCow [231] which uses smaller conditional fields allowing more efficient sampling, and MintNet [327] which approximates the inverse using fixed-point iteration. Alternative approaches to causal masking involve imposing repeated (periodic) structure [174], however in general this is not a good assumption for image modelling, as well as representing convolutions as exponen-

tial matrix-vector products, $\exp(\boldsymbol{M})\boldsymbol{x}$, approximated implicitly with a power series, allowing otherwise unconstrained kernels [144].

2.6.3 Residual Flows

Residual networks [127] are a popular technique to build deep neural networks that alleviate the vanishing gradients problem. By restricting f_θ, invertible residual networks can be built by stacking blocks of the form

$$\boldsymbol{y} = \boldsymbol{x} + f_\theta(\boldsymbol{x}). \tag{2.49}$$

Matrix Determinant Lemma

If a function has a certain residual form, then its Jacobian determinant can be computed with the matrix determinant lemma [293]. A simple example is planar flow [293] which is equivalent to a 3 layer MLP with a single neuron bottleneck:

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{u}h(\boldsymbol{w}^T\boldsymbol{x} + b), \tag{2.50}$$

where $\boldsymbol{u}, \boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, and h is a differentiable non-linearity function. Planar flows are invertible provided some simple conditions are satisfied, however its inverse is difficult to compute making it only practical for density estimation tasks. A higher rank generalisation of the matrix determinant lemma has been applied to planar flows, known as Sylvester flows, removing the severe bottleneck thus allowing greater representation ability [21; 126].

Lipschitz Constrained

By restricting the Lipschitz constant of f_θ, $\|f_\theta\|_L < 1$, then this block is invertible [17]. The inverse, however, has no closed form definition but can be found through fixed-point iteration which by the Banach fixed-point theorem converges to a fixed unique solution at an exponential rate dependant on $\|f_\theta\|_L$. The authors originally proposed a biased approximation of the log determinant of the Jacobian as a power series where the Jacobian trace is approximated using Hutchkinson's trace estimator

(see Tab. 2.3), but an unbiased approximator known as a Russian roulette estimator has also been proposed [42]. Unlike coupling layers, residual flows have dense Jacobians, allowing interaction. Enforcing Lipschitz constraints has been achieved with convolutional networks [103; 221; 248] as well as self-attention [186].

Making strong Lipschitz assumptions severely restricts the class of functions learned; an N layer residual flow network is at most 2^N-Lipshitz. Implicit flows [227] bypass this by solving implicit equations of the form

$$F(\boldsymbol{x}, \boldsymbol{y}) = f_\theta(\boldsymbol{x}) - f_\phi(\boldsymbol{y}) + \boldsymbol{x} - \boldsymbol{y} = \boldsymbol{0}, \tag{2.51}$$

where both f_θ and f_ϕ both have Lipschitz constants less than 1. Both the forwards (solve for $\boldsymbol{y}$ given $\boldsymbol{x}$) and backwards (solve for $\boldsymbol{x}$ given $\boldsymbol{y}$) directions require solving a root finding problem similar to the inverse process of residual flows; indeed, an implicit flow is equivalent to the composition of a residual flow and the inverse of a residual flow. This allows them to model arbitrary Lipschitz transformations.

2.6.4 Surjective and Stochastic Layers

Restricting the class of functions available to those that are invertible introduces a number of practical problems related to the topology-preserving property of diffeomorphisms. For example, mapping a uni-modal distribution to a multi-modal distribution is extremely challenging, requiring a highly varying Jacobian [70]. By composing bijections with surjective or stochastic layers these topological constraints can be bypassed [258]. While the log-likelihood of stochastic layers can only be bounded by their ELBO, functions surjective in the inference direction permit exact likelihood evaluation even with altered dimensionality. Surjective transformations have the following likelihood contributions:

$$\mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x})}\left[\ln \frac{p(\boldsymbol{x}|\boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x})}\right], \tag{2.52}$$

where $p(\boldsymbol{x}|\boldsymbol{y})$ is deterministic for generative surjections, and $q(\boldsymbol{y}|\boldsymbol{x})$ is deterministic for inference surjections.

One approach to build a surjective layer is to augment the input space with additional dimensions allowing smoother transformation to be learned [41; 78; 151]; the inverse process, where some dimensions are factored out, is equivalent to a multi-scale architecture [69]. Another approach known as RAD [70] learns a partitioning of the data space into disjoint subsets $\{\mathcal{Y}_i\}_{i=1}^K$, and applies piece-wise bijections to each region $g_i \colon \mathcal{X} \to \mathcal{Y}_i, \forall i \in \{1, \dots, K\}$. The generative direction learns a classifier on $\mathcal{X}$, $i \sim p(i|\boldsymbol{x})$, allowing the inverse to be calculated as $\boldsymbol{y} = g_i(\boldsymbol{x})$. Similar to both of these approaches are CIFs [56] which consider a continuous partitioning of the data space via augmentation equivalent to an infinite mixture of normalizing flows. Other approaches include modelling finite mixtures of flows [77].

Some powerful stochastic layers have already been discussed in this survey, namely VAEs [190] and DDPMs [135]. Stochastic layers have been incorporated into normalizing flows by interleaving small energy models, sampled with MCMC, between bijectors [377].

2.6.5 Discrete Flows

The normalizing flow framework can be extended to discrete distributions, by restricting transformation functions to be discrete e.g. $f \colon \mathcal{X}^d \to \mathcal{X}^d$. Integer discrete flows (IDF) achieve this using additive coupling layers, rounding translation values to the nearest integer and approximating gradients with the straight-through estimator [143]; discrete flows [347] apply affine coupling layers in modulo space while also restricting the translation and scaling coefficients to a finite number of possible values. In this case the change of variables rule (Eqn. 2.42) simplifies to [143; 347]

$$p(\boldsymbol{x}) = p(f(\boldsymbol{x})). \tag{2.53}$$

Unlike the continuous case, there is no Jacobian determinant term; intuitively this term adjusts for volume changes. However, in a discrete space there is no volume. As such, there is no requirement for f to have an efficiently computable Jacobian determinant [347]. The absence of this term is restricting; however, discrete flows can only permute the values of $p(\boldsymbol{x})$, not change them i.e. a uniform base distribution

can only be mapped to another uniform distribution [271]. Nevertheless, this can be avoided by embedding the data into a space with more values than the data, making IDFs more flexible than discrete flows [22].

2.6.6 Continuous Time Flows

It is possible to consider a normalizing flow with an infinite number of steps that is defined instead by an ordinary differential equation specified by a Lipschitz continuous neural network f with parameters θ, that describes the transformation of a hidden state $\boldsymbol{x}(t) \in \mathbb{R}^D$ [43],

$$\frac{\partial \boldsymbol{x}(t)}{\partial t} = f(\boldsymbol{x}(t), t, \theta). \tag{2.54}$$

Starting from input noise $\boldsymbol{x}(t_0)$, an ODE solver can solve an initial value problem for some time t_1, at which data is defined, $\boldsymbol{x}(t_1)$. Modelling a transformation in this form has a number of advantages such as inherent invertibility by running the ODE solver backwards, parameter efficiency, and adaptive computation. However, it is not immediately clear how to train such a model through backpropagation. While it is possible to backpropagate directly through an ODE solver, this limits the choice of solvers to differentiable ones as well as requiring large amounts of memory. Instead, the authors apply the adjoint sensitivity method which instead solves a second, augmented ODE backwards in time and allows the use of a black box ODE solver. That is, to optimise a loss dependent on an ODE solver:

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}(t_1)) &= \mathcal{L}\bigg(\boldsymbol{x}(t_0) + \int_{t_0}^{t_1} f(\boldsymbol{z}(t), t, \theta) dt\bigg), \\ &= \mathcal{L}(\text{ODESolve}(\boldsymbol{x}(t_0), f, t_0, t_1, \theta)), \end{aligned} \tag{2.55}$$

the adjoint $\boldsymbol{a}(t) = \frac{\partial \mathcal{L}}{\partial \boldsymbol{x}(t)}$ can be used to calculate the derivative of loss with respect to the parameters in the form of another initial value problem [278],

$$\frac{\partial \mathcal{L}}{\partial \theta} = \int_{t_1}^{t_0} \Big(\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}(t)}\Big)^T \frac{\partial f(\boldsymbol{x}(t), t, \theta)}{\partial \theta} dt, \tag{2.56}$$

which can be efficiently evaluated by automatic differentiation at a time cost similar to evaluating f itself.

Despite the complexity of this transformation, the continuous change of variables rule is remarkably simple:

$$\frac{\partial \ln p(\boldsymbol{x}(t))}{\partial t} = -tr\left(\frac{\partial}{\partial \boldsymbol{x}(t)} f(\boldsymbol{x}(t), t, \theta)\right), \tag{2.57}$$

and can be computed using an ODE solver as well. The resulting continuous-time flow is known as FFJORD [107]. Since the length of the flow tends to infinity (an infinitesimal flow), the true posterior distribution can be recovered [293].

As previously mentioned, invertible functions suffer from topological problems; this is especially true for Neural ODEs since their continuous nature prevents trajectories from crossing. Similar to augmented normalizing flows [151], this can be solved by providing additional dimensions for the flow to traverse [78]. Specifically, a p-dimensional Euclidean space can be approximated by a Neural ODE in a $(2p + 1)$-dimensional space [391].

Regularising Trajectories

ODE solvers can require large numbers of network evaluations, notably when the ODE is stiff or the dynamics change quickly in time. By introducing regularisation, a simpler ODE can be learned, reducing the number of evaluations required. Specifically, all works here are inspired by optimal transport theory to encourage straight trajectories. Monge-Ampère Flow [395] and Potential Flow Generators [384] parameterise a potential function satisfying the Monge-Ampère equation [34; 366] with a neural network. RNODE [89] applies transport costs to FFJORD as well as regularising the Frobenius norm of the Jacobian, encouraging straight trajectories. By combining these approaches, OT-Flow [267] utilises the optimal transport derivation to derive an exact trace definition with cost similar to stochastic estimators.

2.7 Evaluation Metrics

A major problem when developing generative models is how to effectively evaluate and compare them. Qualitative comparison of random samples plays a large role in the majority of state-of-the-art works; however, it is subjective and time-consuming to compare many works. Calculating the log-likelihood on a separate validation set is popular for tractable likelihood models but comparison with implicit likelihood models is difficult and while it is a good measure of diversity, it does not correlate well with quality [339].

One approach to quantify sample quality is Inception Score (IS) [309] which takes a trained classifier and determines whether a sample has low label entropy, indicating that a meaningful class is likely, and whether the distribution of classes over a large number of samples has high entropy, indicating that a diverse range of images can be sampled. A perfect IS can be scored by a model that creates only one image per class [229] leading to the creation of Fréchet Inception Distance (FID) [128] which models the activations of a particular layer of a classifier as multivariate Gaussians for real and generated data, measuring the Fréchet distance between the two.

Limitations of the FID Metric

While FID is a popular choice for evaluating sample quality, it has been found to correlate well with image quality, and is efficient to calculate, it is not without flaws. It unrealistically approximates the data distribution as Gaussian in embedding space and is insensitive to the global structure of the data distribution [350]. Kernel Inception Distance (KID) [25] instead calculates the squared maximum mean discrepancy in feature space; however, pretrained features may not be sufficient to detect overfitting. Another approach is to train a neural network to distinguish between real and generated samples similar to the discriminator from a GAN; while this detects overfitting, it increases the complexity and time required to evaluate a model and is biased towards adversarial models [120]. Other approaches that address these issues [27] include PPL [179], which assesses sample consistency through latent interpo-

lations; IMD [350], which uses all moments making it sensitive to global structure; and MTD [14], which compares image manifolds.

Precision and Recall

Precision (P) and Recall (R) [306] approaches (Tab. 4.1), unlike FID, evaluate sample quality and diversity separately by quantifying the overlap between the data and sample distributions. Precision is the expected likelihood of fake samples lying on the data manifold and recall vice versa. These metrics are computed by approximating the data and sample manifolds as hyper-spheres around data and sample points respectively; manifold $\mathrm{m}(X_1, \ldots, X_N) = \bigcup_{i=1}^{N} B(X_i, \mathrm{NND}_k(X_i))$, where $B(x, r)$ is a hypersphere around x with radius r and NND_k is k^{th} nearest neighbour distance [202]. While modelling manifolds as hyperspheres is a flawed assumption, it is beneficial to evaluate on multiple metrics to obtain a more accurate representation of performance. Finally, Density (D) and Coverage (C) are modifications to Precision and Recall respectively that address manifold overestimation [250]. Formally, these metrics can be defined as,

$$\mathrm{P} = \frac{1}{M} \sum_{j=1}^{M} 1_{Y_j \in \mathrm{m}(X_1, \ldots, X_N)}, \quad \text{(2.58a)} \qquad \mathrm{D} = \frac{1}{kM} \sum_{j=1}^{M} \sum_{i=1}^{N} 1_{Y_j \in B(X_i, \mathrm{NND}_k(X_i))}, \quad \text{(2.58b)}$$

$$\mathrm{R} = \frac{1}{N} \sum_{i=1}^{N} 1_{X_i \in \mathrm{m}(Y_1, \ldots, Y_M)}, \quad \text{(2.58c)} \qquad \mathrm{C} = \frac{1}{N} \sum_{i=1}^{N} 1_{\exists j \mathrm{s.t} Y_j \in B(X_i, \mathrm{NND}_k(X_i))}. \quad \text{(2.58d)}$$

Challenges of evaluating speed

When developing generative models, there are two speed factors to consider: the time required to train the model, and the time required for sampling. Sometimes these factors can be traded-off against one another; for instance, by using more compute to distil diffusion models, sampling times can be reduced. In theory, the best method to compare times would be to simply time training/sampling from start to finish. However, this is problematic as GPUs improve in performance over time and receive more memory, this requires re-running previous models to compare against. This is also not trivial since code can be optimised to run on specific

hardware, research code can become depricated with library updates, and efficient GPU kernels such as Flash Attention [61] come into common use. As such, this can result in considerable effort being require to replicate old methods and provide fair comparison. For approaches which require multiple forward passes of the model during inference such as diffusion models and it can therefore make sense to compare inference time in terms of model calls.

2.8 Applications

In general, the definition of a generative model means that any technique can be used on any modality/task; however, some models are more suited for certain tasks. Standard autoregressive networks are popular for text/audio generation [30; 49; 359]; VAEs have been applied but posterior collapse is difficult to mitigate [11; 28]; GANs are more parameter efficient but struggle to model discrete data [257] and suffer from mode collapse [199]; some normalizing flows offer parallel synthesis, providing substantial speedup [280; 347; 411]. Video synthesis is more challenging due its exceptionally high dimensionality, typically approaches combine a latent-based implicit generative model to generate individual frames, with an autoregressive network used to predict future latents [11; 200; 210] similar to how world models are constructed in reinforcement learning [122; 123]. Modality conversion has been achieved using GANs [408], VAE-GANs [222], and DDPMs [313].

2.9 Datasets

A variety of datasets with different properties are commonly used to train and evaluate generation models. This section describes each dataset used within this thesis.

2.9.1 Low Resolution Datasets

The following low resolution datasets are used in this thesis to provide a spread from simple (MNIST) to complex (CIFAR-10).

MNIST [206]: a dataset of 70,000 greyscale images of handwritten digits (0-9), each 28×28 pixels.

Fashion MNIST [378]: similar in structure to MNIST, but instead of digits, containing 70,000 greyscale images of 10 types of clothing items (e.g. shoes and t-shirts), each 28×28 pixels.

Small NORB [207]: contains 194,400 images of 50 toy figurines under various lighting and pose conditions. It contains 50 toys across 5 categories (e.g. cars and planes).

COIL20 [251]: the Columbia Object Image Library dataset, with 1,440 greyscale images of 20 objects, each photographed at 72 different angles.

CelebA [225]: large-scale face attributes data with with more than 200,000 celebrity images, each annotated with 40 attribute labels and 178×218 pixels.

CIFAR-10 [196]: contains 60,000 32×32 colour images across 10 classes (e.g. animals and vehicles).

2.9.2 High Resolution Datasets

CelebA-HQ [178]: a high-quality subset of CelebA, consisting of 30,000 images at 1024×1024 resolution. Face datasets are challenging because humans are very sensitive to inconsistencies in facial representations, potentially leading to an "uncanny valley" effect.

FFHQ [179]: contains 70,000 high-resolution 1024×1024 images of human faces sourced from Flickr. This dataset has more diversity in terms of age, ethnicity, and image background than CelebA-HQ.

LSUN Bedroom [385]: part of the larger LSUN (Large-scale Scene Understanding) dataset. This subset contains around 3 million images of bedrooms, each at least 256×256 pixels.

LSUN Church [385]: another subset of the LSUN dataset, this subset containing around 126k images, each at least 256×256 pixels.

2.10 Conclusion

This chapter explored the major classes of generative models; while for a while GANs led the way in terms of sample quality, recent advances have led to other approaches, particularly diffusion models, being brought to par or even surpassing GANs, without the disadvantages such as unstable training and mode collapse associated with adversarial approaches.

Hybrids between different types of model offer a balance between extremes at the expense of additional complexity. The varied connections between these systems mean that advances in one field inevitably benefit others, for instance, improved variational bounds are beneficial for VAEs, diffusion models, and surjective flows, and the application of data augmentation has been found to offer benefits across numerous model classes without necessitating more powerful architectures.

When it comes to scaling models to high-dimensional data each type of generative model suffers from different issues which will be addressed in the following chapters. There are, however, some common themes in addressing this, particularly architecturally: attention allows long-range dependencies to be learned and recent advances in linear attention will aid scaling to even higher resolutions.

CHAPTER 3

Gradient Origin Networks

Implicit representation learning [272; 335], where a network is parameterised to represent data continuously rather than in discrete grid form, has seen a surge of interest due to the small number of parameters, speed of convergence, ability to model fine details, and represent irregularly sampled data. In particular, sinusoidal representation networks (SIRENs) [322] achieve impressive results, modelling many signals with high precision, thanks to their use of periodic activations paired with carefully initialised MLPs. So far, however, these models have been limited to modelling single data samples, or use an additional hypernetwork or meta learning [321] to estimate the weights of a simple implicit model, adding significant complexity.

A more scalable method to allow a generative function to represent a distribution of data points is to condition the function on a compressed latent space by encoding data with a convolutional network in a similar manner to Variational Autoencoders (VAEs, Figure 3.1a) [190]. When applied to implicit representation networks, however, this approach is problematic as convolutional networks assume that data lies on a fixed size grid thereby removing the benefits of parameterising data continuously. Variational approaches that approximate the posterior using gradient descent [219] and short run MCMC [261] respectively have been proposed, which do not suffer

(a) VAE (b) GON (c) Implicit GON (d) Variational GON

Figure 3.1: Gradient Origin Networks (GONs; b) use gradients (dashed lines) as encodings thus only a single network F is required, which can be an implicit representation network (c). Unlike VAEs (a) which use two networks, E and D, variational GONs (d) permit sampling with only one network.

from this limitation, but to obtain a latent vector for a sample they require iterative gradient updates significantly slowing training and sampling.

This chapter introduces Gradient Origin Networks (GONs), addressing these challenges,

- GONs are a new type of generative model (Figure 3.1b) that does not require encoders or hypernetworks. This is achieved by initialising latent points at the origin, then using the gradient of the log-likelihood of the data with respect to these points as the latent space. At inference, latent vectors can be obtained in a single step without requiring iteration.
- GONs are shown to have similar characteristics to convolutional autoencoders (AEs) and variational autoencoders using approximately half the parameters.
- We show that GONs can be applied to implicit representation networks (such as SIRENs) allowing a space of implicit functions to be learned with a simpler overall architecture.

3.1 Empirical Bayes

First, some background context is introduced that will be used to derive the proposed approach. Empirical Bayes [296; 311], for a random variable $\mathbf{z} \sim p_{\mathbf{z}}$ and particular observation $\mathbf{z}_0 \sim p_{\mathbf{z}_0}$, is a method that provides an estimator of $\mathbf{z}$ expressed purely in terms of $p(\mathbf{z}_0)$ that minimises the expected squared error. This estimator can be

written as a conditional mean:

$$\hat{\mathbf{z}}(\mathbf{z}_0) = \int \mathbf{z} p(\mathbf{z}|\mathbf{z}_0) d\mathbf{z} = \int \mathbf{z} \frac{p(\mathbf{z}, \mathbf{z}_0)}{p(\mathbf{z}_0)} d\mathbf{z}. \tag{3.1}$$

Of particular relevance is the case where $\mathbf{z}_0$ is a noisy observation of $\mathbf{z}$ with covariance $\boldsymbol{\Sigma}$. In this case $p(\mathbf{z}_0)$ can be represented by marginalising out $\mathbf{z}$:

$$p(\mathbf{z}_0) = \int \frac{1}{(2\pi)^{d/2} |\det(\boldsymbol{\Sigma})|^{1/2}} \exp\Big(-(\mathbf{z}_0 - \mathbf{z})^T \boldsymbol{\Sigma}^{-1} (\mathbf{z}_0 - \mathbf{z})/2 \Big) p(\mathbf{z}) d\mathbf{z}. \tag{3.2}$$

Differentiating this with respect to $\mathbf{z}_0$ and multiplying both sides by $\boldsymbol{\Sigma}$ gives:

$$\boldsymbol{\Sigma} \nabla_{\mathbf{z}_0} p(\mathbf{z}_0) = \int (\mathbf{z} - \mathbf{z}_0) p(\mathbf{z}, \mathbf{z}_0) d\mathbf{z} = \int \mathbf{z} p(\mathbf{z}, \mathbf{z}_0) d\mathbf{z} - \mathbf{z}_0 p(\mathbf{z}_0). \tag{3.3}$$

After dividing through by $p(\mathbf{z}_0)$ and combining with Eqn. 3.1 we obtain a closed form estimator of $\mathbf{z}$ [247] written in terms of the score function $\nabla \log p(\mathbf{z}_0)$ [158]:

$$\hat{\mathbf{z}}(\mathbf{z}_0) = \mathbf{z}_0 + \boldsymbol{\Sigma} \nabla_{\mathbf{z}_0} \log p(\mathbf{z}_0). \tag{3.4}$$

This optimal procedure is achieved in what can be interpreted as a single gradient descent step, with no knowledge of the prior $p(\mathbf{z})$. By rearranging Eqn. 3.4, a definition of $\nabla \log p(\mathbf{z}_0)$ can be derived; this can be used to train models that approximate the score function [326].

3.2 Method

Consider some dataset $\mathbf{x} \sim p_d$ of continuous or discrete signals $\mathbf{x} \in \mathbb{R}^m$, it is typical to assume that the data can be represented by low dimensional latent variables $\mathbf{z} \in \mathbb{R}^k$, which can be used by a generative neural network to reconstruct the data. These variables are often estimated through the use of a secondary encoding network that is trained concurrently with the generative network. An encoding network adds additional complexity (and parameters) to the model, it can be difficult to balance capacities of the two networks, and for complex hierarchical generative models designing a suitable architecture can be difficult. This has led some to instead

approximate latent variables by performing gradient descent on the generative network [26; 261]. While this addresses the aforementioned problems, it significantly increases the run time of the inference process, introduces additional hyperparameters to tune, and convergence is not guaranteed.

3.2.1 Gradient Origin Networks

We propose a generative model that consists only of a decoding network, using empirical Bayes to approximate the posterior in a single step. That is, for some data point $\mathbf{x}$ and latent variable $\mathbf{z} \sim p_{\mathbf{z}}$, we wish to find an approximation of $p(\mathbf{z}|\mathbf{x})$. Given some noisy observation $\mathbf{z}_0 = \mathbf{z} + \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ of $\mathbf{z}$ then empirical Bayes can be applied to approximate $\mathbf{z}$. Specifically, since we wish to approximate $\mathbf{z}$ conditioned on $\mathbf{x}$, we instead calculate $\hat{\mathbf{z}}_{\mathbf{x}}$, the least squares estimate of $p(\mathbf{z}|\mathbf{x})$.

$$\hat{\mathbf{z}}_{\mathbf{x}}(\mathbf{z}_0) = \int \mathbf{z} p(\mathbf{z}|\mathbf{z}_0, \mathbf{x}) d\mathbf{z} = \int \mathbf{z} \frac{p(\mathbf{z}_0, \mathbf{z}|\mathbf{x})}{p(\mathbf{z}_0|\mathbf{x})} d\mathbf{z}. \tag{3.5}$$

Through the definition of the probabilistic chain rule and by marginalising out $\mathbf{z}$, we can define $p(\mathbf{z}_0|\mathbf{x}) = \int p(\mathbf{z}_0|\mathbf{z}, \mathbf{x}) p(\mathbf{z}|\mathbf{x}) d\mathbf{z}$ which can be simplified to $\int p(\mathbf{z}_0|\mathbf{z}) p(\mathbf{z}|\mathbf{x}) d\mathbf{z}$ since $\mathbf{z}_0$ is dependent only on $\mathbf{z}$. Writing this out fully, we obtain:

$$p(\mathbf{z}_0|\mathbf{x}) = \int \frac{1}{(2\pi)^{d/2} |\det(\boldsymbol{\Sigma})|^{1/2}} \exp\Big(-(\mathbf{z}_0 - \mathbf{z})^T \boldsymbol{\Sigma}^{-1} (\mathbf{z}_0 - \mathbf{z})/2 \Big) p(\mathbf{z}|\mathbf{x}) d\mathbf{z}. \tag{3.6}$$

Differentiating with respect to $\mathbf{z}_0$ and multiplying both sides by $\boldsymbol{\Sigma}$ gives:

$$\boldsymbol{\Sigma} \nabla_{\mathbf{z}_0} p(\mathbf{z}_0|\mathbf{x}) = \int (\mathbf{z} - \mathbf{z}_0) p(\mathbf{z}_0|\mathbf{z}, \mathbf{x}) p(\mathbf{z}|\mathbf{x}) d\mathbf{z} = \int (\mathbf{z} - \mathbf{z}_0) p(\mathbf{z}_0, \mathbf{z}|\mathbf{x}) d\mathbf{z} \tag{3.7}$$

$$= \int \mathbf{z} p(\mathbf{z}_0, \mathbf{z}|\mathbf{x}) d\mathbf{z} - \mathbf{z}_0 p(\mathbf{z}_0|\mathbf{x}). \tag{3.8}$$

After dividing both sides by $p(\mathbf{z}_0|\mathbf{x})$ and combining with Eqn. 3.5 we get:

$$\boldsymbol{\Sigma} \frac{\nabla_{\mathbf{z}_0} p(\mathbf{z}_0|\mathbf{x})}{p(\mathbf{z}_0|\mathbf{x})} = \int \mathbf{z} \frac{p(\mathbf{z}_0, \mathbf{z}|\mathbf{x})}{p(\mathbf{z}_0|\mathbf{x})} d\mathbf{z} - \mathbf{z}_0 = \hat{\mathbf{z}}_{\mathbf{x}}(\mathbf{z}_0) - \mathbf{z}_0. \tag{3.9}$$

Finally, this can be rearranged to give the single step estimator of $\mathbf{z}$:

$$\hat{\mathbf{z}}_{\mathbf{x}}(\mathbf{z}_0) = \mathbf{z}_0 + \boldsymbol{\Sigma}\frac{\nabla_{\mathbf{z}_0} p(\mathbf{z}_0|\mathbf{x})}{p(\mathbf{z}_0|\mathbf{x})} = \mathbf{z}_0 + \boldsymbol{\Sigma}\nabla_{\mathbf{z}_0} \log p(\mathbf{z}_0|\mathbf{x}). \tag{3.10}$$

Using Bayes' rule, $\log p(\mathbf{z}_0|\mathbf{x})$ can be written as $\log p(\mathbf{z}_0|\mathbf{x}) = \log p(\mathbf{x}|\mathbf{z}_0) + \log p(\mathbf{z}_0) - \log p(\mathbf{x})$. Since $\log p(\mathbf{x})$ is a normalising constant that does not affect the gradient, and by setting $\boldsymbol{\Sigma} = \boldsymbol{I}_d$, we can rewrite Eqn. 3.9 in terms only of the decoding network and $p(\mathbf{z}_0)$:

$$\hat{\mathbf{z}}_{\mathbf{x}}(\mathbf{z}_0) = \mathbf{z}_0 + \nabla_{\mathbf{z}_0}\Big(\log p(\mathbf{x}|\mathbf{z}_0) + \log p(\mathbf{z}_0)\Big). \tag{3.11}$$

It still remains, however, how to construct a noisy estimate of $\mathbf{z}_0$ with no knowledge of $\mathbf{z}$. If we assume $\mathbf{z}$ follows a known distribution, then it is possible to develop reasonable estimates. For instance, if we assume $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \boldsymbol{I}_d)$ then we could sample from $p(\mathbf{z}_0) = \mathcal{N}(\mathbf{z}_0; \mathbf{0}, 2\boldsymbol{I}_d)$ however this could be far from the true distribution of $p(\mathbf{z}_0|\mathbf{z}) = \mathcal{N}(\mathbf{z}_0; \mathbf{z}, \boldsymbol{I}_d)$. Instead we propose initialising $\mathbf{z}_0$ at the origin since this is the distribution's mean. Initialising at a constant position decreases the input variation and thus simplifies the optimisation procedure. Naturally, how $p(\mathbf{x}|\mathbf{z})$ is modelled affects $\hat{\mathbf{z}}_{\mathbf{x}}$. While mean-field models result in $\hat{\mathbf{z}}_{\mathbf{x}}$ that are linear functions of $\mathbf{x}$, conditional autoregressive models, for instance, result in non-linear $\hat{\mathbf{z}}_{\mathbf{x}}$; multiple gradient steps also induce non-linearity; however, we show that a single step works well on high dimensional data suggesting that linear encoders, which normally do not scale to high dimensional data are effective in this case.

3.2.2 Autoencoding with GONs

Before exploring GONs as generative models, we discuss the case where the prior $p(\mathbf{z})$ is unknown; such a model is referred to as an autoencoder. As such, the distribution $p(\mathbf{z}_0|\mathbf{z})$ is also unknown thus it is again unclear how we can construct a noisy estimate of $\mathbf{z}$. By training a model end-to-end where $\mathbf{z}_0$ is chosen as the origin however, a prior is implicitly learned over $\mathbf{z}$ such that it is reachable from $\mathbf{z}_0$. Although $p(\mathbf{z})$ is unknown, we do not wish to impose a prior on $\mathbf{z}_0$; the term

which enforces this is in Eqn. 3.11 is $\log p(\mathbf{z}_0)$, so we can safely ignore this term and simply maximise the likelihood of the data given $\mathbf{z}_0$. Our estimator of $\mathbf{z}$ can therefore be defined simply as $\hat{\mathbf{z}}_{\mathbf{x}}(\mathbf{z}_0) = \mathbf{z}_0 + \nabla_{\mathbf{z}_0} \log p(\mathbf{x}|\mathbf{z}_0)$, which can otherwise be interpreted as a single gradient descent step on the conditional log-likelihood of the data. From this estimate, the data can be reconstructed by passing $\hat{\mathbf{z}}_{\mathbf{x}}$ through the decoder to parameterise $p(\mathbf{x}|\hat{\mathbf{z}}_{\mathbf{x}})$. This procedure can be viewed more explicitly when using a neural network $F\colon \mathbb{R}^k \to \mathbb{R}^m$ to output the mean of $p(\mathbf{x}|\hat{\mathbf{z}}_{\mathbf{x}})$ parameterised by a normal distribution; in this case the loss function is defined in terms of mean squared error loss $\mathcal{L}^{\text{MSE}}$:

$$G_{\mathbf{x}} = \mathcal{L}^{\text{MSE}}(\mathbf{x}, F(-\nabla_{\mathbf{z}_0}\mathcal{L}^{\text{MSE}}(\mathbf{x}, F(\mathbf{z}_0)))). \tag{3.12}$$

The gradient computation thereby plays a similar role to an encoder, while F can be viewed as a decoder, with the outer loss term determining the overall reconstruction quality. Using a single network to perform both roles has the advantage of simplifying the overall architecture, avoiding the need to balance networks, and avoiding bottlenecks; this is demonstrated in Figs. 3.1b and 3.2 which provides a visualisation of the GON process.

3.2.3 Variational GONs

The variational approach can be naturally applied to GONs, allowing sampling in a single step while only requiring the generative network, reducing the parameters necessary. Similar to before, a feedforward neural network F parameterises $p(\mathbf{x}|\mathbf{z})$, while the expectation of $p(\mathbf{z}|\mathbf{x})$ is calculated with empirical Bayes. A normal prior is assumed over $p(\mathbf{z})$ thus Eqn. 3.11 can be written as:

$$\hat{\mathbf{z}}_{\mathbf{x}}(\mathbf{z}_0) = \mathbf{z}_0 + \nabla_{\mathbf{z}_0}\Big(\log p(\mathbf{x}|\mathbf{z}_0) + \log \mathcal{N}(\mathbf{z}_0; \mathbf{0}, 2\boldsymbol{I}_d)\Big), \tag{3.13}$$

where $\mathbf{z}_0 = \mathbf{0}$ as discussed in Section 3.2.1. It would be possible to use this estimate directly within a constant-variance VAE [98], where the approximate posterior uses a fixed pre-defined covariance matrix. However, we opt to incorporate the repa-

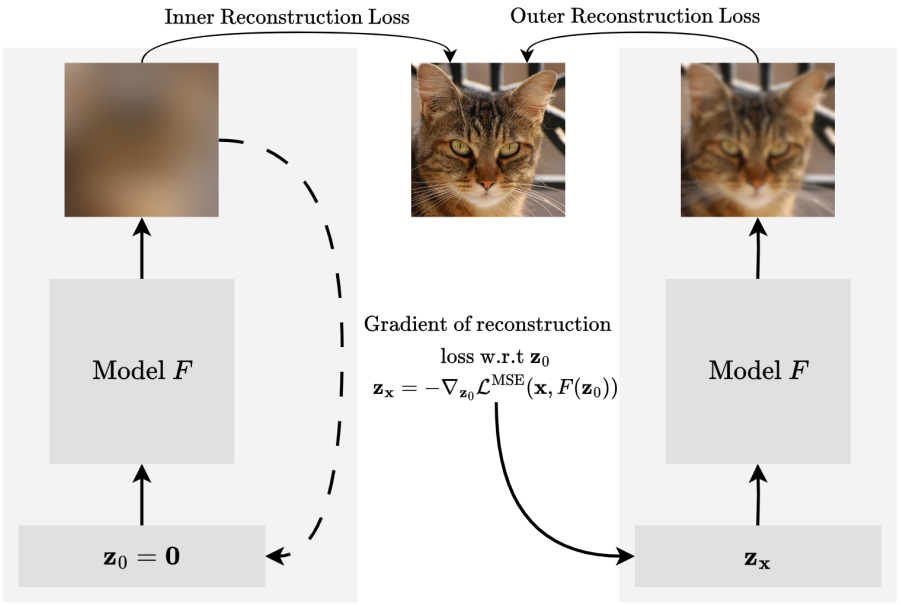


Figure 3.2: Visualisation of how GONs encode data. $\mathbf{z}_0$ is passed through the model F to obtain an initial estimate of the data. Reconstruction loss is computed against the data to be encoded. Following this, the gradient of the reconstruction loss with respect to $\mathbf{z}_0$ is computed. This gives a new vector $\mathbf{z}_\mathbf{x}$ which represents the compressed data. Following this, $\mathbf{z}_\mathbf{x}$ is passed through the model F, providing a reconstruction of the image. Finally, the reconstruction loss is computed again, and is backpropagated through the entire computation process to update the weights of F. As such, GONs learn to reconstruct data without requiring an encoder therefore allowing application to models such as implicit representation networks where encoders make less sense.

rameterisation trick into the generative network as a stochastic layer, to represent the distribution over which $\mathbf{x}$ could be encoded to, using empirical Bayes to estimate $\mathbf{z}$. This enables a better approximation of the true posterior leading to higher sample quality. Similar to the autoencoding approach, we could ignore $\log p(\mathbf{z}_0)$, however we find assuming a normally distributed $\mathbf{z}_0$ implicitly constrains $\mathbf{z}$, aiding the optimisation procedure. Specifically, the forward pass of F is implemented as follows: $\hat{\mathbf{z}}_\mathbf{x}$ (or $\mathbf{z}_0$) is mapped by linear transformations to $\mu(\hat{\mathbf{z}}_\mathbf{x})$ and $\sigma(\hat{\mathbf{z}}_\mathbf{x})$ and the reparameterisation trick is applied, subsequently the further transformations formerly defined as F in the GON formulation are applied, providing parameters for for $p(\mathbf{x}|\hat{\mathbf{z}}_\mathbf{x})$. Training is performed end-to-end, minimising the ELBO:

$$\mathcal{L}^{\text{VAE}} = -D_{\text{KL}}(\mathcal{N}(\mu(\hat{\mathbf{z}}_\mathbf{x}), \sigma(\hat{\mathbf{z}}_\mathbf{x})^2)||\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)) + \log p(\mathbf{x}|\hat{\mathbf{z}}_\mathbf{x}). \tag{3.14}$$

These steps are shown in Figure 3.1d, which in practice has a simple implementation.

3.2.4 Implicit GONs

In the field of implicit representation networks, the aim is to learn a neural approximation of a function Φ that satisfies an implicit equation of the form:

$$R(\mathbf{c}, \Phi, \nabla_\Phi, \nabla^2_\Phi, \dots) = 0, \quad \Phi\colon \mathbf{c} \mapsto \Phi(\mathbf{c}), \tag{3.15}$$

where R's definition is problem dependent but often corresponds to a loss function. Equations with this structure arise in a myriad of fields, namely 3D modelling, image, video, and audio representation [322]. In these cases, data samples $\mathbf{x} = \{(\mathbf{c}, \Phi_\mathbf{x}(\mathbf{c}))\}$ can be represented in this form in terms of coordinates $\mathbf{c} \in \mathbb{R}^n$ and the corresponding data at those points $\Phi\colon \mathbb{R}^n \to \mathbb{R}^m$. Due to the continuous nature of Φ, data with large spatial dimensions can be represented much more efficiently than approaches using convolutions, for instance. Despite these benefits, representing a distribution of data points using implicit methods is more challenging, leading to the use of hypernetworks which estimate the weights of an implicit network for each data point [321]; this increases the number of parameters and adds significant complexity.

By applying the GON procedure to implicit representation networks, it is possible learn a space of implicit functions without the need for additional networks. We assume there exist latent vectors $\mathbf{z} \in \mathbb{R}^k$ corresponding to data samples; the concatentation of these latent variables with the data coordinates can therefore geometrically be seen as points on a manifold that describe the full dataset in keeping with the manifold hypothesis [87]. An implicit Gradient Origin Network can be trained on this space to mimic Φ using a neural network $F\colon \mathbb{R}^{n+k} \to \mathbb{R}^m$, thereby learning a space of implicit functions by modifying Eqn. 3.12:

$$I_\mathbf{x} = \int \mathcal{L}\Big(\Phi_\mathbf{x}(\mathbf{c}), F\Big(\mathbf{c} \oplus -\nabla_{\mathbf{z}_0} \int \mathcal{L}\big(\Phi_\mathbf{x}(\mathbf{c}), F(\mathbf{c} \oplus \mathbf{z}_0)\big) d\mathbf{c}\Big)\Big) d\mathbf{c}, \tag{3.16}$$

where both integrations are performed over the space of coordinates and $\mathbf{c} \oplus \mathbf{z}$ represents a concatenation (Figure 3.1c). Similar to the non-implicit approach, the

computation of latent vectors can be expressed as $\mathbf{z} = -\nabla_{\mathbf{z}_0} \int \mathcal{L}\big(\Phi_{\mathbf{x}}(\mathbf{c}), F(\mathbf{c} \oplus \mathbf{z}_0)\big) d\mathbf{c}$. In particular, we parameterise F with a SIREN [322], finding that it is capable of modelling both high and low frequency components in high dimensional spaces.

3.2.5 GON Generalisations

There are a number of interesting generalisations that make this approach applicable to other tasks. In Eqns. 3.12 and 3.16 we use the same $\mathcal{L}$ in both the inner term and outer term; however, as with variational GONs, it is possible to use different functions; through this, training a GON concurrently as a generative model and classifier is possible, or through some minor modifications to the loss involving the addition of the categorical cross entropy loss function $\mathcal{L}^{\text{CCE}}$ to maximise the likelihood of classification, solely as a classifier:

$$C_{\mathbf{x}} = \mathcal{L}^{\text{CCE}}(f(-\nabla_{\mathbf{z}_0} \mathcal{L}(\mathbf{x}, F(\mathbf{z}_0))), \mathbf{y}), \tag{3.17}$$

where $\mathbf{y}$ are the target labels and f is a single linear layer followed by softmax. Another possibility is modality conversion for translation tasks; in this case the inner reconstruction loss is performed on the source signal and the outer loss on the target signal.

3.2.6 Justification

Beyond empirical Bayes, we provide some additional analysis on why a single gradient step is in general sufficient as an encoder. Firstly, the gradient of the loss inherently offers substantial information about data making it a good encoder. Secondly, a good latent space should satisfy local consistency [173; 405]. GONs satisfy this since similar data points will have similar gradients due to the constant latent initialisation. As such, the network needs only to find an equilibrium where its prior is the gradient operation, allowing for significant freedom. Finally, since GONs are trained by backpropagating through empirical Bayes, there are benefits to using an activation function whose second derivative is non-zero.

3.3 Results

We evaluate Gradient Origin Networks on a variety of image datasets, from small simple datasets: MNIST [206], Fashion-MNIST [378], Small NORB [207], COIL-20 [251], to larger more complex datasets: CIFAR-10 [196] and CelebA [225], and finally the high-resolution LSUN Bedroom dataset [385]. Simple models are used: for small images, implicit GONs consist of approximately 4 hidden layers of 256 units and convolutional GONs consist of 4 convolution layers with Batch Normalization [159] and the ELU non-linearity [54], for larger images the same general architecture is used, scaled up with additional layers; all training is performed with the Adam optimiser [189]; all experiments are performed on a single NVIDIA RTX 2080 Ti with 11GB of VRAM. Error bars in graphs represent standard deviations over three runs.

3.3.1 Quantitative Evaluation

A quantitative evaluation of the representation ability of GONs is performed in Tab. 3.1 against a number of baseline approaches. Because simple models are used to demonstrate the impact of replacing an encoder with the GON approach, we only quantitatively evaluate on low-resolution datasets. We compare against the single step methods: standard autoencoder, an autoencoder with tied weights [95], and a GON with gradients detached from $\mathbf{z}$, as well as the multi-step methods: 10

	MNIST	Fashion-MNIST	Small NORB	COIL20	CIFAR-10
Single Step					
GON (ours)	**0.41±0.01**	**1.00±0.01**	**0.17±0.01**	5.35±0.01	**9.12±0.03**
AE	1.33±0.02	1.75±0.02	0.73±0.01	6.05±0.11	12.24±0.05
Tied AE	2.16±0.03	2.45±0.02	0.93±0.03	6.68±0.13	14.12±0.34
1 Step Detach	8.17±0.15	7.76±0.21	1.84±0.01	15.72±0.70	30.17±0.29
Multiple Steps					
10 Step Detach	8.13±0.50	7.22±0.92	1.78±0.02	15.48±0.60	28.68±1.16
10 Step	0.42±0.01	1.01±0.01	**0.17±0.01**	5.36±0.01	9.19±0.01
GLO	0.70±0.01	1.78±0.01	0.61±0.01	**3.27±0.02**	**9.12±0.03**

Table 3.1: Validation reconstruction loss (summed squared error) over 500 epochs. For GLO, latents are assigned to data points and jointly optimised with the network. GONs significantly outperform other single step methods and achieve the lowest reconstruction error on four of the five datasets.

	MNIST	Fashion-MNIST	Small NORB	COIL20	CIFAR-10	CelebA
VGON (ours)	**1.06**	3.30	**2.34**	**3.44**	**5.85**	**5.41**
VAE	1.15	**3.23**	2.54	3.63	5.94	5.59

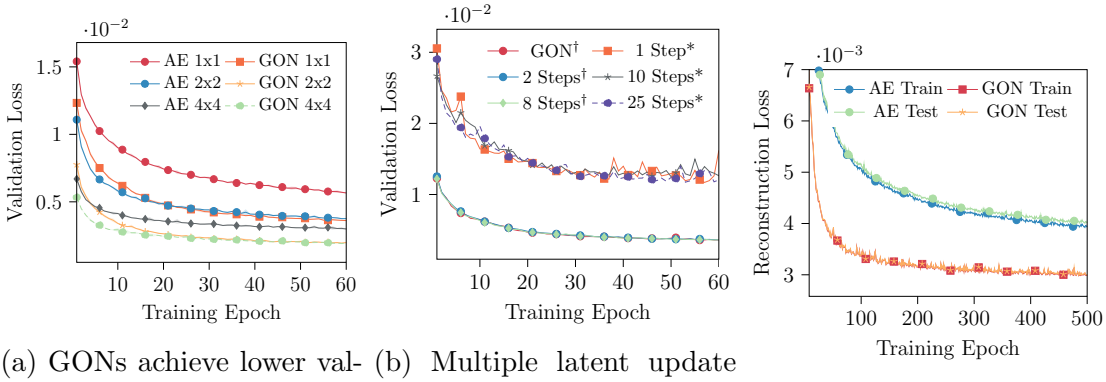
Table 3.2: Validation ELBO in bits/dim over 1000 epochs (CelebA is trained over 150 epochs).

gradient descent steps per data point, with and without detaching gradients, and a Generative Latent Optimisation (GLO) model [26] which assigns a persistent latent vector to each data point and optimises them with gradient descent, therefore taking orders of magnitude longer to reconstruct validation data than other approaches. For the 10-step methods, a learning rate of 0.1 is applied as used in other literature [259]; the GLO is trained with MSE for consistency with the other approaches and we do not project latents onto a hypersphere as proposed by the authors since in this experiment sampling is unimportant and this would handicap the approach. GONs achieve much lower validation loss than other single step methods and are competitive with the multi-step approaches; in fact, GONs achieve the lowest loss on four out of the five datasets.

Our variational GON is compared with a VAE, whose decoder is the same as the GON, quantitatively in terms of ELBO on the test set in Tab. 3.2. We find that the representation ability of GONs is pertinent here also, allowing the variational GON to achieve lower ELBO on five out of the six datasets tested. Both models were trained with the original VAE formulation for fairness; however, we note that variations aimed at improving VAE sample quality such as β-VAEs [130] are also applicable to variational GONs to the same effect.

A number of ablation studies are performed to evaluate GONs; for these we choose the CIFAR-10 dataset since it is more complex than the other low-resolution datasets. First, convolutional GONs are compared with autoencoders whose decoders have exactly the same architecture as the GONs (Fig. 3.3a) and where the autoencoder decoders mirror their respective encoders. The mean reconstruction loss over the test set is measured after each epoch for a variety of latent sizes. Despite having half the number of parameters and linear encodings, GONs achieve significantly lower reconstruction loss over a wide range of architectures.

Our hypothesis that for high dimensional datasets, a single gradient step is suf-



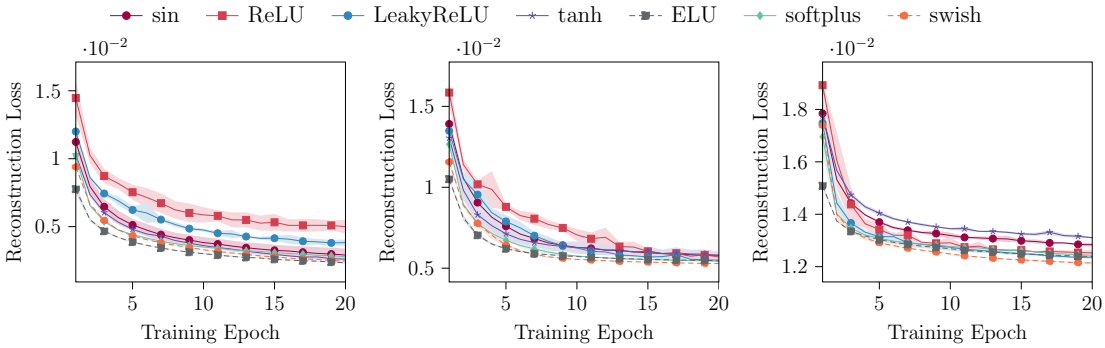
(a) GONs achieve lower validation loss than autoencoders.

(b) Multiple latent update steps. *=grads detached, †=not detached.

(c) GONs overfit less than standard autoencoders.

Figure 3.3: Gradient Origin Networks trained on CIFAR-10 are found to outperform autoencoders using exactly the same architecture without the encoder, requiring half the number of parameters.

ficient when jointly optimised with the forwards pass is tested on the CIFAR-10 dataset in Fig. 3.3b. We observe negligible difference between a single step and multiple jointly optimised steps, in support of our hypothesis. Performing multiple steps greatly increases run-time so there is seemingly no benefit in this case. Additionally, the importance of the joint optimisation procedure is determined by detaching the gradients from $\mathbf{z}$ before reconstructing (Fig. 3.3b); this results in markedly worse performance, even when in conjunction with multiple steps. In Fig. 3.3c we assess whether the greater performance of GONs relative to autoencoders comes at the expense of generalisation; we find that the opposite is true, that the discrepancy be-



(a) 512 latent variables

(b) 128 latent variables

(c) 32 latent variables

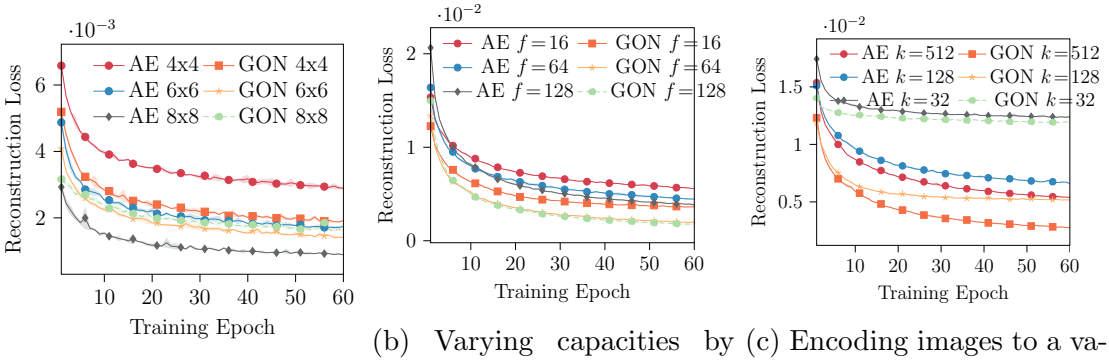
Figure 3.4: The impact of activation function and number of latent variables on model performance for a GON trained on CIFAR-10 measured by comparing reconstruction losses through training.

tween reconstruction loss on the training and test sets is greater with autoencoders.

Fig. 3.4 demonstrates the effect activation functions have on convolutional GON performance for different numbers of latent variables. Since optimising GONs requires computing second order derivatives, the choice of nonlinearity requires different characteristics to standard models. In particular, GONs prefer functions that are not only effective activation functions, but also whose second derivatives are non-zero, unlike ReLUs where $\text{ReLU}''(x) = 0$. The ELU non-linearity is effective with all tested architectures.

In Fig. 3.5a the depth of networks is altered by removing blocks thereby downscaling to various resolutions. We find that GONs outperform autoencoders until latents have a spatial size of 8x8 (where the equivalent GON now only has only 2 convolution layers). Considering the limit where neither model has any parameters, the latent space is the input data i.e. $\mathbf{z} = \mathbf{x}$. Substituting the definition of a GON (Eqn. 3.12) this becomes $-\nabla_{\mathbf{z}_0}\mathcal{L} = \mathbf{x}$ which simplifies to $\mathbf{0} = \mathbf{x}$ which is a contradiction. This is not a concern in normal practice, as evidenced by the results presented here.

Fig. 3.5b explores the relationship between autoencoders and GONs when changing the number of convolution filters; GONs are found to outperform autoencoders

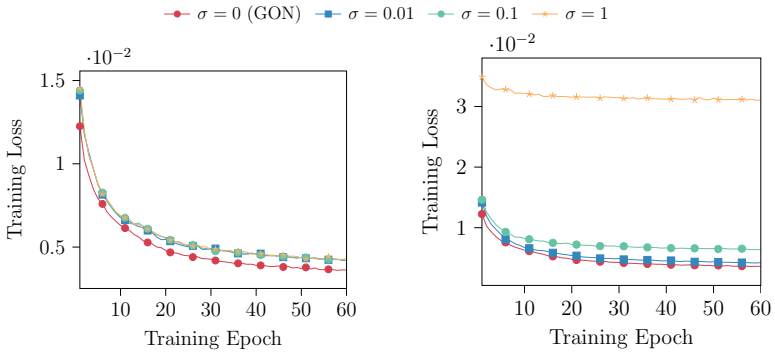


(a) Encoding images to various spatial dimensions.

(b) Varying capacities by changing the number of convolution filters f.

(c) Encoding images to a variety of latent space sizes.

Figure 3.5: Experiments comparing convolutional GONs with autoencoders on CIFAR-10, where the GON uses exactly same architecture as the AE, without the encoder. (a) At the limit autoencoders tend towards the identity function whereas GONs are unable to operate with no parameters. As the number of network parameters increases (b) and the latent size decreases (c), the performance lead of GONs over AEs decreases due to diminishing returns/bottlenecking.



(a) $\mathbf{z} = -\nabla_{\mathbf{z}_0}\mathcal{L}$ (b) $\mathbf{z} = \mathbf{z}_0 - \nabla_{\mathbf{z}_0}\mathcal{L}$

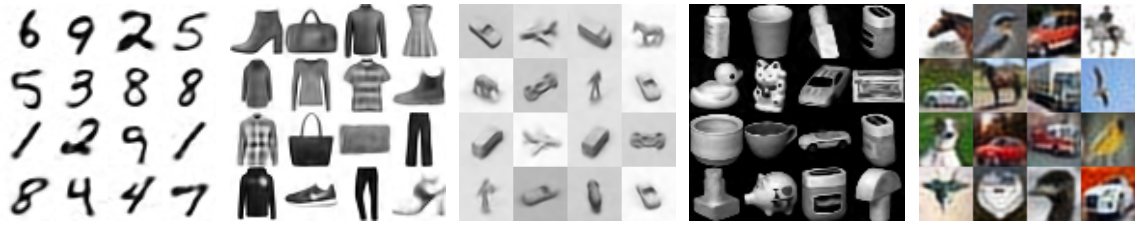
Figure 3.6: Training GONs on CIFAR-10 with $\mathbf{z}_0$ sampled from a variety of normal distributions with different standard deviations σ, $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$. Approach (a) directly uses the negative gradients as encodings while approach (b) performs one gradient descent style step initialised at $\mathbf{z}_0$.

in all cases. The discrepancy between loss curves decreases as the number of filters increases likely due to the diminishing returns of providing more capacity to the GON when its loss is significantly closer to 0. A similar pattern is found when decreasing the latent space (Fig. 3.5c); in this case the latent space likely becomes the limiting factor. With larger latent spaces GONs significantly outperform autoencoders; however, when the latent bottleneck becomes more pronounced this lead lessens.

Finally, we evaluate different initialisations of $\mathbf{z}_0$ in Fig. 3.6 by sampling $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$ for a variety of standard deviations σ. The proposed approach ($\sigma = 0$) achieves the lowest reconstruction loss (Fig. 3.6a); results for $\sigma > 0$ are similar, suggesting that the latent space is adjusted so $\mathbf{z}_0$ simulates the origin. An alternative parameterisation of $\mathbf{z}$ is to use a use a single gradient descent style step $\mathbf{z} = \mathbf{z}_0 - \nabla_{\mathbf{z}_0}\mathcal{L}$ (Fig. 3.6b); however, losses are higher than the proposed GON initialisation.

3.3.2 Qualitative Evaluation

The representation ability of implicit GONs is shown in Fig. 3.7 where we train on large image datasets using a relatively small number of parameters. In particular, Fig. 3.7a shows MNIST can be well-represented with just 4,385 parameters (a SIREN with 3 hidden layers each with 32 hidden units, and 32 latent dimensions). An advantage of modelling data with implicit networks is that coordinates can be at



(a) 4,385 parameters (b) 297k parameters (c) 270k parameters (d) 270k parameters (e) 396k parameters

Figure 3.7: Training implicit GONs with few parameters demonstrates their representation ability.

arbitrarily high resolutions. In Fig. 3.8 we train on 32x32 images then reconstruct at 256x256. This resolution is chosen because it is substantially higher than the original 32x32 resolution; while it is possible to reconstruct at arbitrarily high resolutions, far beyond 256x256 there are diminishing returns when reconstructing at even higher resolutions. A significant amount of high frequency detail is modelled despite only seeing low resolution images. Similar can be seen in Fig. 3.9 for 28x28 MNIST test images upsampled to 256x256. The structure of the implicit GON latent space is shown by sampling latent codes from pairs of images, and then spherically interpolating between them to synthesise new samples (Fig. 3.10). These samples are shown to capture variations in shape (the shoes in Fig. 3.10b), size, and rotation (Fig. 3.10d).

We assess the quality of samples from variational GONs using convolutional models trained to convergence in Fig. 3.12. These are diverse and often contain fine details. A simple alternative to variational GONs is to train with early stopping, where we don't train to conversion, instead stopping early; this acts as a form of regularisation, keeping the latent space simple to sample from. Samples from an implicit GON trained with early stopping can be found in Fig. 3.11 however this

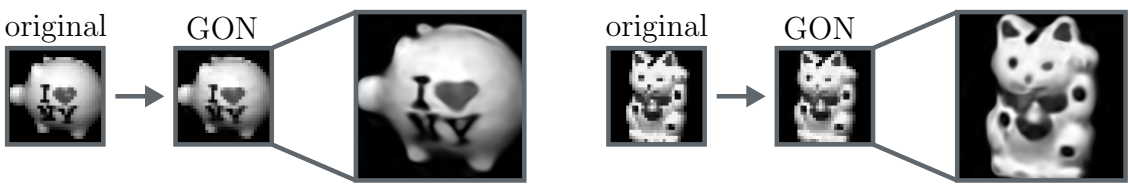
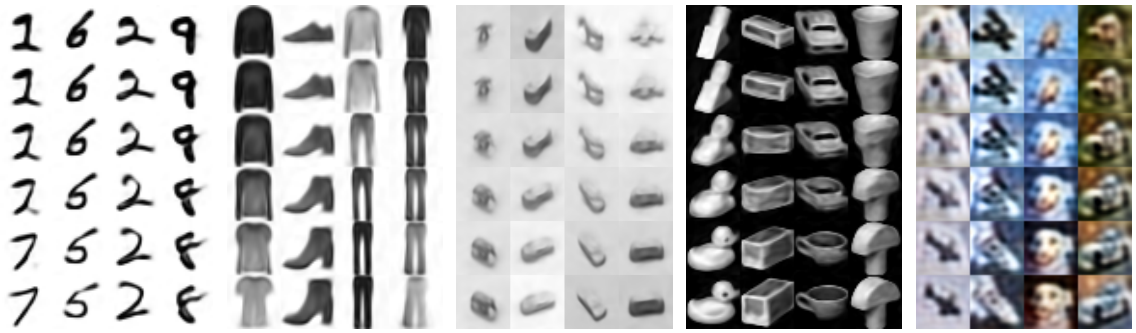


Figure 3.8: By training an implicit GON on 32x32 images, then sampling at 256x256, super-resolution is possible despite never observing high resolution data.

(a) Original data 28x28 (b) Implicit GON at 28x28

Figure 3.9: Super-sampling 28x28 MNIST test data at 256x256 coordinates using an implicit GON.



(a) MNIST (b) Fashion-MNIST (c) Small NORB (d) COIL20 (e) CIFAR-10

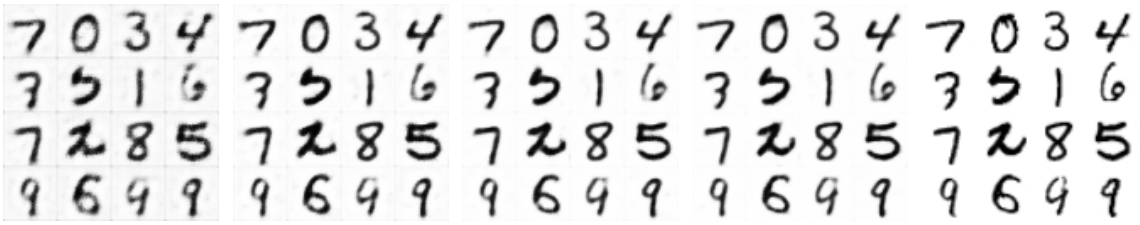
Figure 3.10: Spherical linear interpolations between points in the latent space for trained implicit GONs using different datasets (approximately 2-10 minutes training per dataset on a single GPU).

(a) 4,385 parameters (b) 70k parameters (c) 270k parameters (d) 270k parameters (e) 270k parameters

Figure 3.11: GONs trained with early stopping can be sampled by approximating their latent space with a multivariate normal distribution. These images show samples from an implicit GON trained with early stopping.

(a) 4,331 parameters (b) 270k parameters (c) 270k parameters (d) 270k parameters (e) 270k parameters

Figure 3.12: Random samples from a convolutional variational GON with normally distributed latents.

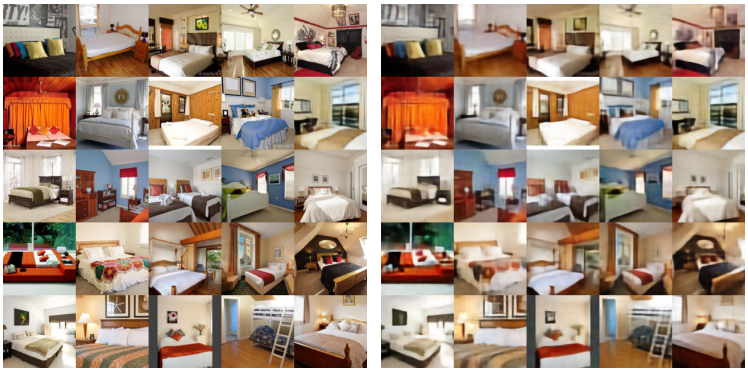


(a) 3s 300 steps (b) 10s 1000 steps (c) 30s 3000 steps (d) 60s 6000 steps (e) Ground Truth

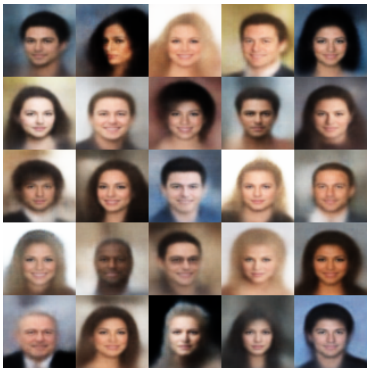
Figure 3.13: Convergence of convolutional GONs with 74k parameters.

approach results in fine details being lost. Training further would result in a more complex latent space which would have to be separately modelled by a powerful density estimator such as a diffusion model. GONs are also found to converge quickly; we plot reconstructions at multiple time points during the first minute of training (Fig. 3.13). After only 3 seconds of training on a single GPU, a large amount of signal information from MNIST is modelled.

In order to evaluate how well GONs can represent high resolution natural images, we train a convolutional GON on the LSUN Bedroom dataset scaled to 128x128 (Fig. 3.14a). As with smaller, more simple data, we find training to always be extremely stable and consistent over a wide range of hyperparameter settings. Reconstructions are of excellent quality given the simple network architecture. A convolutional variational GON is also trained on the CelebA dataset scaled to 64x64



(a) LSUN 128x128 Bedroom validation images (left) reconstructed by a convolutional GON (right).



(b) Samples from a convolutional variational GON trained on CelebA.

Figure 3.14: GONs are able to represent high resolution complex datasets to a high degree of fidelity.

(Fig. 3.14b). Unconditional samples are somewhat blurry as commonly associated with traditional VAE models on complex natural images [400] but otherwise show wide variety.

3.4 Discussion

Despite similarities with autoencoder approaches, the absence of an encoding network offers several advantages. VAEs with overpowered decoders are known to ignore the latent variables [46] whereas GONs only have one network that equally serves both encoding and decoding functionality. Designing inference networks for complicated decoders is not a trivial task [356]; however, inferring latent variables using a GON simplifies this procedure. Similar to GONs, Normalizing Flow methods are also capable of encoding and decoding with a single set of weights; however, they achieve this by restricting the network to be invertible. This requires considerable architectural restrictions that affect performance, make them less parameter efficient, and are unable to reduce the dimension of the data [191]. Similarly, autoencoders with tied weights also encode and decode with a single set of weights by using the transpose of the encoder's weight matrices in the decoder; this however, is only applicable to simple architectures. GONs on the other hand use gradients as encodings which allow arbitrary functions to be used.

A number of previous works have used gradient-based computations to learn latent vectors however as far as we are aware, we are the first to use a single gradient step jointly optimised with the feedforward pass, making it fundamentally different to these approaches. Latent encodings have been estimated for pre-trained generative models without encoders, namely Generative Adversarial Networks, using approaches such as standard gradient descent [219; 407]. A number of approaches have trained generative models directly with gradient-based inference [26; 125; 387]; these assign latent vectors to data points and jointly learn them with the network parameters through standard gradient descent or Langevin dynamics. This is very slow however, and convergence for unseen data samples is not guaranteed. Short run MCMC has also been applied [261] however this still requires approximately

25 update steps. Since GONs train end-to-end, the optimiser can make use of the second order derivatives to allow for inference in a single step. Also of relevance is model-agnostic meta-learning [90], which trains an architecture so that a few gradient descent steps are all that are necessary to new tasks. This is achieved by backpropagating through these gradients, similar to GONs.

In the case of implicit GONs, the integration terms in Eqn. 3.16 result in computation time that scales in proportion with the data dimension. This makes training slower for very high dimensional data, although we have not yet investigated Monte Carlo integration. In general GONs are stable and consistent, capable of generating quality samples with an exceptionally small number of parameters, and converge to diverse results with few iterations. Nevertheless, there are avenues to explore so as to improve the quality of samples and scale to larger datasets. In particular, it would be beneficial to focus on how to better sample these models, perform formal analysis on the gradients, and investigate whether the distance function could be improved to better capture fine details.

3.5 Conclusion

This chapter introduced a method based on empirical Bayes which computes the gradient of the data fitting loss with respect to the origin, and then jointly fits the data while learning this new point of reference in the latent space. The results show that this approach is able to represent datasets using a small number of parameters with a simple overall architecture, which has advantages in applications such as implicit representation networks. GONs are shown to converge faster with lower overall reconstruction loss than autoencoders, using the exact same architecture but without the encoder. Experiments show that the choice of non-linearity is important, as the network derivative jointly acts as the encoding function.

While implicit GONs theoretically enable representing data at arbitrarily high resolutions, in practice, reliance on using a single global latent vector to represent data points results in reconstructions/samples which are blurry. The subsequent chapters address this downside in two different ways: Chapter 4 uses spatially dis-

tributed information rich discrete latent vectors to represent images, while Chapter 5 introduces a diffusion model that operates at arbitrarily high resolutions, thereby allowing more powerful stochastic reconstructions.

CHAPTER 4

Unleashing Transformers: Parallel Token Prediction with Discrete Absorbing Diffusion for Fast High-Resolution Image Generation from Vector-Quantized Codes

While the approach outlined in Chapter 3 offers the ability to sample arbitrarily high-resolution data, these samples are blurry, a problem which is exacerbated as the resolution increases. This chapter instead attempts to find a balance between these extremes with the aim to efficiently generate much higher quality samples that are of large but finite resolution.

Vector-Quantized image models (Sec. 2.3.2) such as VQ-VAEs [362] and VQ-GANs [86] are extremely effective approaches for compressing high-resolution data, offering impressive perceptual reconstruction quality for the very high compression rates achieved. This is primarily due to using a highly information rich codebook, each code from which is relatively high dimensional allowing substantial quantities of information to be stored. The discrete nature of these encodings made autoregressive models (Sec. 2.5) a natural choice to model the distribution over these encodings due to their strong density estimation capability.

Autoregressive models in this context have a number of downsides: sampling is

Figure 4.1: Our approach uses a discrete diffusion to quickly generate high quality images optionally larger than the training data (right).

slow and sequential, growing linearly with dimension making scaling to even higher resolutions more difficult; a fixed ordering of the inputs is required which ignores the 2D structure of images thereby restricting modelling ability.

This chapter addresses these problems by using a discrete diffusion model (Sec. 2.2.4) to represent Vector-Quantized image representations; this is visualised in Fig. 4.2. By removing the autoregressive constraint, allowing bidirectional context when generating samples, not only is it possible to speed up sampling, but an improved feature representation is learned, enabling higher quality image generation. While there are other non-autoregressive discrete generative models, each of them have a number of downsides that make them less suitable in this context, for instance, discrete implicit EBMs [110] have very long mixing times, similarly, generative masked language models have very long sample times and are ineffective at modelling longer sequences [105; 369]. The main contributions of this chapter are:

- A parallel token prediction approach for generating Vector-Quantized images allowing much faster sampling than autoregressive models is proposed.
- This approach is able to generate globally consistent images at resolutions exceeding that of the original training data by aggregating multiple context windows, allowing for much larger context regions (see Fig. 4.1).
- State-of-the art performance on three benchmark datasets is demonstrated in terms of Density (LSUN Bedroom: 1.51; LSUN Churches: 1.12; FFHQ:

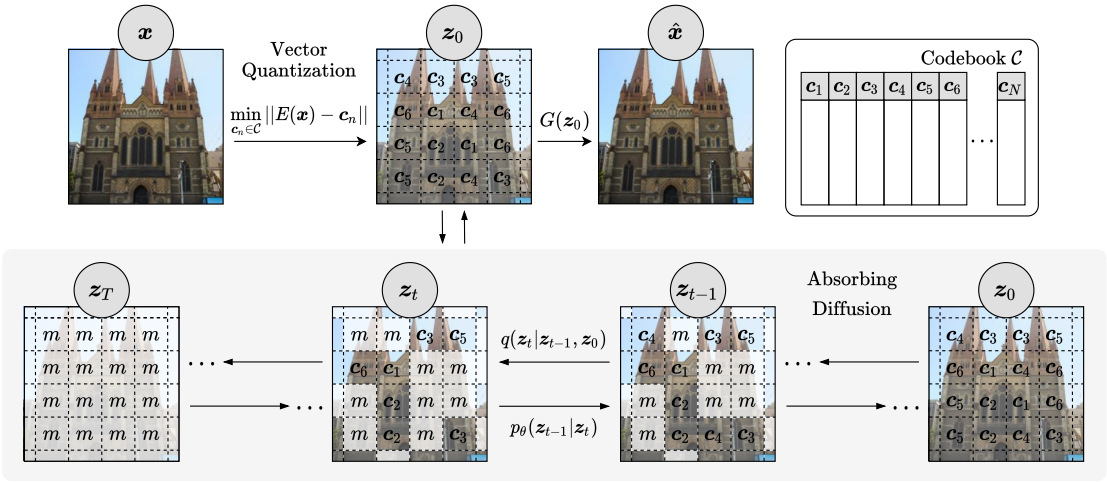


Figure 4.2: Our approach uses a discrete absorbing diffusion model to represent Vector-Quantized images allowing fast high-resolution image generation. Specifically, after compressing images to an information-rich discrete space, elements are randomly masked and an unconstrained Transformer is trained to denoise the data, using global context to ensure samples are consistent and high quality.

1.20) and Coverage (Bedroom: 0.83; Churches: 0.73; FFHQ: 0.80), as well as competitive results on FID (Bedroom: 3.64; Churches: 4.07; FFHQ: 6.11).

4.1 Method

The first stage of the proposed approach is to learn compact discrete representations using a Vector-Quantized autoencoder. Here, a convolutional encoder downsamples images $\boldsymbol{x}$ to a smaller spatial resolution, $E(\boldsymbol{x}) = \{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_L\} \in \mathbb{R}^{L \times D}$. Argmax quantisation [362] is used to discretise the continuous embeddings: for a codebook $\mathcal{C} \in \mathbb{R}^{K \times D}$, where K is the number of discrete codes in the codebook and D is the dimension of each code, each $\boldsymbol{e}_i$ is mapped via a nearest-neighbour lookup onto a discrete codebook value, $\boldsymbol{c}_j \in \mathcal{C}$:

$$\boldsymbol{z}_q = \{\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_L\} \text{ , where } \boldsymbol{q}_i = \min_{\boldsymbol{c}_j \in \mathcal{C}} \|\boldsymbol{e}_i - \boldsymbol{c}_j\|. \tag{4.1}$$

As this operation is non-differentiable, the straight-through gradient estimator [18] is used to approximate gradients resulting in bias. The quantized latents are fed through a decoder $\hat{\boldsymbol{x}} = G(\boldsymbol{z}_q)$ to reconstruct the input based on a perceptual recon-

struction loss [86; 396]; this process is trained by minimising the loss $\mathcal{L}_{\text{VQ}}$,

$$\mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{rec}} + \|\text{sg}[E(\boldsymbol{x})] - \boldsymbol{z}_q\|_2^2 + \beta\|\text{sg}[\boldsymbol{z}_q] - E(\boldsymbol{x})\|_2^2. \tag{4.2}$$

4.1.1 Sampling Globally Coherent Latents

Once the training data is encoded as discrete, integer-valued latents $\boldsymbol{z} \in \mathbb{Z}^D$, a discrete diffusion model can be used to learn the distribution over this highly compressed space. Specifically, we use the absorbing state diffusion [9] where in each forward time step t, each discrete latent at coordinate i, $[\boldsymbol{z}]_i$, is independently either kept the same or masked out entirely with probability $\frac{1}{t}$; the reverse process gradually unveils these masks. In this formulation, the transition matrix is defined as $\boldsymbol{Q}_t = (1-\beta_t)I + \beta_t \mathbb{1} e_m^T$ where e_m is a vector with a one on mask states m and zeros elsewhere, and the beta schedule is $\beta_t = \frac{1}{T-t+1}$. Rather than directly approximating $p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$, training stochasticity is reduced by predicting $p_\theta(\boldsymbol{z}_0|\boldsymbol{z}_t)$ [135]. In this case, the variational bound reduces to

$$\mathbb{E}_{q(\boldsymbol{z}_0)}\left[\sum_{t=1}^{T}\frac{1}{t}\,\mathbb{E}_{q(\boldsymbol{z}_t|\boldsymbol{z}_0)}\Big[\sum_{[\boldsymbol{z}_t]_i=m}\log p_\theta([\boldsymbol{z}_0]_i|\boldsymbol{z}_t)\Big]\right]. \tag{4.3}$$

With p_θ modelled using multinomial distributions, a temperature $\tau < 1$ can be applied to improve sample quality at the expense of diversity. This controls the randomness of predictions by scaling the logits before applying the softmax function, with lower temperatures leading to more deterministic predictions, and higher temperatures resulting in more diverse and random outputs.

Unlike, uniform diffusion, absorbing diffusion is an effective strategy for Vector-Quantized image modelling as noisy elements are removed entirely rather than being changed to a different value which in the discrete case may be unrelated but are much less easy to identify. Gaussian and token distance transitions which change states based on embedding distances are similarly ineffective as Vector-Quantized latents are not ordinal meaning that state changes can significantly change tokens' semantics. This effectiveness is further evidenced by the success of BERT [65] which similarly learns to denoise randomly masked data.

Architecture

Esser et al. [86] demonstrated that in the autoregressive case, Transformers [364] are better suited for modelling Vector-Quantized images than convolutional approaches due to the importance of long-distance relationships in this compressed form. As such, we use transformers to model the prior, but without the architectural restrictions imposed by autoregressive approaches. This procedure (Eqn. 4.4) encodes inputs as key-value pairs, where values $\boldsymbol{V}$ represent embedded inputs and keys $\boldsymbol{K}$ act as an indexing method, subsequently, a set of queries $\boldsymbol{Q}$ are used to select which values to observe:

$$\text{Attn}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V}. \tag{4.4}$$

This process allows interactions with strong gradients between all inputs, irrespective of their spatial relationships.

Fast Sampling

Because the diffusion model is trained to predict $p(\boldsymbol{z}_0|\boldsymbol{z}_t)$, it is possible to sample skipping an arbitrary number of time steps k, $p_\theta(\boldsymbol{z}_{t-k}|\boldsymbol{z}_t)$, allowing sampling in significantly fewer steps than autoregressive approaches.

4.1.2 Addressing Gradient Variance

When inputs are very noisy (at time steps close to T), denoising is difficult and the stochastic training results in gradients with high variance. As such, in practice continuous diffusion models are trained to estimate the noise rather than directly predict the denoised data, significantly reducing the variance. Unfortunately, no relevant reparameterisation currently exists for discrete distributions [145]. Instead, we address this problem by reweighting the ELBO based on the information available at time t, $\frac{T-t+1}{T}$ [9], so that components of the loss at time steps closer to T are weighted less than earlier steps. This effectively alters the learning rate based on gradient variance, improving convergence. I.e. we adapt Eqn. 4.3 to,

$$\mathbb{E}_{q(\boldsymbol{z}_0)}\left[\sum\nolimits_{t=1}^{T} \frac{T-t+1}{T}\, \mathbb{E}_{q(\boldsymbol{z}_t|\boldsymbol{z}_0)}\left[\sum\nolimits_{[\boldsymbol{z}_t]_i=m} \log p_\theta([\boldsymbol{z}_0]_i|\boldsymbol{z}_t)\right]\right]. \tag{4.5}$$

This is equivalent to the loss obtained by assuming the posterior does not have access to $\boldsymbol{z}_t$ when $[\boldsymbol{z}_t]_i = m$. Since we predict $\boldsymbol{z}_0$ this assumption does not harm the training.

This can be derived from the true ELBO defined in [9]. The loss at time step t can be written as

$$\begin{aligned}\mathcal{L}_t &= D_{\mathrm{KL}}(q(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t, \boldsymbol{z}_0) \parallel p(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)) \\ &= \sum_i \sum_j q([\boldsymbol{z}_{t-1}]_{i,j}|\boldsymbol{z}_t \boldsymbol{z}_0) \log \frac{q([\boldsymbol{z}_{t-1}]_{i,j}|\boldsymbol{z}_t, \boldsymbol{z}_0)}{p([\boldsymbol{z}_{t-1}]_{i,j}|\boldsymbol{z}_t)},\end{aligned} \tag{4.6}$$

where the first summation sums over latent coordinates i, and the second summation sums over the probabilities of each code j. For the absorbing diffusion case where tokens in $\boldsymbol{z}_t$ are masked independently and uniformly with probability $\frac{t}{T}$, this posterior is defined as

$$q([\boldsymbol{z}_{t-1}]_i = a|\boldsymbol{z}_t, \boldsymbol{z}_0) = \begin{cases} q([\boldsymbol{z}_{t-1}]_i = a|\boldsymbol{z}_0), & \text{if } [\boldsymbol{z}_t]_i = m. \\ 1, & \text{if } a = [\boldsymbol{z}_0]_i \text{ and } [\boldsymbol{z}_t]_i = [\boldsymbol{z}_0]_i. \\ 0, & \text{otherwise,} \end{cases} \tag{4.7}$$

where $q([\boldsymbol{z}_{t-1}]_i = [\boldsymbol{z}_0]_i|\boldsymbol{z}_0) = (1 - \frac{t-1}{T})$ and $q([\boldsymbol{z}_{t-1}]_i = m|\boldsymbol{z}_0) = \frac{t-1}{T}$. The reverse process remains defined in the same way as the standard reverse process:

$$p([\boldsymbol{z}_{t-1}]_i = a|\boldsymbol{z}_t) = \begin{cases} \frac{1}{t} p_\theta([\boldsymbol{z}_0]_i|\boldsymbol{z}_t), & \text{if } a = [\boldsymbol{z}_0]_i \text{ and } [\boldsymbol{z}_t]_i = m. \\ 1 - \frac{1}{t}, & \text{if } a = m \text{ and } [\boldsymbol{z}_t]_i = m. \\ 1, & \text{if } a = [\boldsymbol{z}_0]_i \text{ and } [\boldsymbol{z}_t]_i = [\boldsymbol{z}_0]_i. \end{cases} \tag{4.8}$$

Substituting these definitions into Eqn. 4.6, the loss can be simplified to Eqn. 4.9; by extracting the constants into a single term out of the sum, C, the loss can be further simplified to obtain Eqn. 4.10, which is equivalent to our proposed reweighted ELBO,

$$\mathcal{L}_t = \sum_i \left[1 \log \frac{1}{1} + \frac{t-1}{T} \log \frac{\frac{t-1}{T}}{1 - \frac{1}{t}} + \left(1 - \frac{t-1}{T} \log \frac{1 - \frac{t-1}{T}}{\frac{1}{t} p_\theta([\boldsymbol{z}_0]_i|\boldsymbol{z}_t)} \right) \right], \tag{4.9}$$

$$= C - \sum_i \left[\frac{T-t+1}{T} \log p_\theta([\boldsymbol{z}_0]_i | \boldsymbol{z}_t) \right]. \tag{4.10}$$

4.1.3 Generating High-Resolution Images

Using convolutions to build Vector-Quantized image models encourages latents to be highly spatially correlated with generated images. It is therefore possible to construct essentially arbitrarily sized images by generating latents with the required shape. We propose an approach that allows globally consistent images substantially larger than those in the training data to be generated.

First, a large a by b array of mask tokens, $\bar{\boldsymbol{z}}_T = m^{a \times b}$, is initialised that corresponds to the size of image we wish to generate. In order to capture the maximum context when approximating $\bar{\boldsymbol{z}}_0$ we apply the denoising network to all subsets of $\bar{\boldsymbol{z}}_t$ with the same spatial size as the usual inputs of the network, aggregating estimates at each location. Specifically, using $c_j(\bar{\boldsymbol{z}}_t)$ to represent local subsets, we approximate the denoising distribution as a mixture,

$$p([\bar{\boldsymbol{z}}_0]_i | \bar{\boldsymbol{z}}_t) \approx \frac{1}{Z} \sum_j p([\bar{\boldsymbol{z}}_0]_i | c_j(\bar{\boldsymbol{z}}_t)), \tag{4.11}$$

where the sum is over subsets c_j that contain the i^{th} latent and Z is the normalising constant. For extremely large images, this can require a very large number of function evaluations; however, the sum can be approximated by striding over latents with a step > 1 or by randomly selecting positions.

4.1.4 Improving Code Representations

There are various options to obtain high-quality image representations including using large numbers of latents and codes [288] or building a hierarchy of latent variables [291]. We use the adversarial framework proposed by Esser et al. [86] to achieve higher compression rates with high-quality codes using only a single GPU, without tying our approach to the characteristics typically associated with generative adversarial models. Additionally, we apply differentiable augmentations T, such as translations and colour jitter, to all discriminator inputs; this has proven to be

effective at improving sample quality across methods [171; 402]. The overall loss $\mathcal{L}$ is a linear combination of $\mathcal{L}_{\text{VQ}}$, the Vector-Quantized loss, and $\mathcal{L}_{\text{G}}$ which uses a discriminator D to assess realism based on an adaptive weight λ. On some datasets, λ can grow to extremely large values hindering training. We find simply clamping λ at a maximum value $\lambda_{\max} = 1$ an effective solution that stabilises training,

$$\mathcal{L} = \min_{E,G,\mathcal{C}} \max_{D} \mathbb{E}_{p_d} \left[\mathcal{L}_{\text{VQ}} + \lambda \mathcal{L}_{\text{G}} \right], \quad (4.12a) \qquad \lambda = \min \left(\frac{\nabla_{G_L}[\mathcal{L}_{\text{rec}}]}{\nabla_{G_L}[\mathcal{L}_{\text{G}}] + \delta}, \lambda_{\max} \right), \quad (4.12b)$$

$$\mathcal{L}_{\text{G}} = \log D(T(\boldsymbol{x})) + \log(1 - D(T(\hat{\boldsymbol{x}}))). \quad (4.12c)$$

The argmax quantisation approach can result in codebook collapse, where some codes are never used; while other quantisation methods can reduce this [67; 163; 234; 288], we found argmax to yield the highest reconstruction quality.

4.2 Evaluation

We evaluate our approach on the three largest high-resolution 256x256 datasets described in Sec. 2.9, LSUN Bedroom, LSUN Churches [385], and FFHQ [179]. Large datasets are required to prevent overfitting due to the high parameter count of the model. Sec. 4.2.1 evaluates the quality of samples from our proposed model. Sec. 4.2.2 demonstrates the representation abilities of absorbing diffusion models applied to the learned discrete latent spaces, including how sampling can be sped up, improvements over equivalent autoregressive models, and the effect of our reweighted ELBO. Finally, Sec. 4.2.3 evaluates our Vector-Quantized image model.

In all experiments, our absorbing diffusion model parameterised with an 80M parameter Transformer Encoder [364] is applied to 16×16 latents discretised to a codebook with 1024 entries and optimised using the Adam optimiser [189]. While, as noted by Esser et al. [86], a GPT2-medium [284] architecture (307M parameters) fits onto a GPU with 12GB of VRAM, in practice this requires small batch sizes and learning rates making training in reasonable times impractical.

We perform all experiments on a single NVIDIA RTX 2080 Ti with 11GB of VRAM using automatic mixed precision when possible. We use the same VQGAN architecture as used by Esser et al. [86] which for 256×256 images downsamples to features of size $16 \times 16 \times 256$, and quantizes using a codebook with 1024 entries. Attention layers are applied within both the encoder and decoder on the lowest resolutions to aggregate context across the entire image. Models are optimised using the Adam optimiser [189] using a batch size of 4 and learning rate of 1.8×10^{-5}. For the differentiable augmentations we randomly change the brightness, saturation, and contrast, as well as randomly translate images. The datasets we use are both publically accessible, with FFHQ available under the Creative Commons BY 4.0 licence. LSUN models are trained for 2.2M steps and the FFHQ model for 1.4M steps.

For the absorbing diffusion model we use a scaled down 80M parameter version of GPT-2 [284] consisting of 24 layers, where each attention layer has 8 heads, each 64D. The same architecture is used for experiments with the autoregressive model. Autoregressive models' training are stopped based on the best validation loss. We also stop training the absorbing diffusion models based on validation ELBO; however, on the LSUN datasets we found that it always improved or remained consistent throughout training so each model was trained for 2M steps. Source code is available at https://github.com/samb-t/unleashing-transformers.

Codebook Collapse

One issue with vector quantized methods is codebook collapse, where some codes fall out of use which limits the potential expressivity of the model. We found this to occur across all datasets with often a fraction of the codes in use. We experimented with different quantization schemes such as gumbel softmax, different initalisation schemes such as k-means, and 'code recycling', where codes out of use are reset to an in use code. In all of these cases, we found the reconstruction quality to be comparable or worse so stuck with the argmax quantisation scheme used by Esser et al. [86].

Figure 4.3: Samples from our models trained on 256x256 datasets: LSUN Churches, FFHQ, and LSUN Bedroom.

Precision, Recall, Density, and Coverage

To compute these measures we use the official code releases and pretrained weights in all cases except Taming Transformers on the LSUN datasets where weights were not available; in this case we reproduced results as close as possible with the hardware available, training the VQGANs and autoregressive models with the same hyperparameters used for the rest of our experiments. Following Nash et al. [253] we use the standard 2048D InceptionV3 features, which are also used to compute FID, $k = 3$ nearest neighbours, and 50k samples, and use the code provided by Naeem et al. [250].

4.2.1 Sample Quality

In this section we evaluate our model quantitatively and qualitatively. In contrast to other multi-step methods, our approach allows sampling in the fewest steps. Samples can be found in Figs. 4.3 and 4.4 which are high quality and diverse.

Due to limited computing resources, we are unable to provide Density and Coverage scores for DCT [253] and PRDC scores for StyleGAN2 on LSUN Bedroom since training on a standard GPU would take more than 30 days, much more than the 10 days to train our models. On LSUN our approach achieves the highest Precision, Density, and Coverage; indicating that the data and sample manifolds have the most overlap (Tab. 4.1). On FFHQ our approach achieves the highest Precision and Recall. When sampling with lower temperatures to improve FID, generative

(a) Non-cherry picked, $\tau = 0.9$, 256×256 LSUN Churches samples.

(b) Non-cherry picked, $\tau = 0.85$, 256×256 FFHQ samples.

(c) Non-cherry picked, $\tau = 0.9$, 256×256 LSUN Bedroom samples.

Figure 4.4: Samples from our approach are diverse and high quality.

Model	LSUN Churches				LSUN Bedroom				FFHQ			
	P ↑	R ↑	D ↑	C ↑	P ↑	R ↑	D ↑	C ↑	P ↑	R ↑	D ↑	C ↑
DCT [253]	0.60	0.48	-	-	0.44	**0.56**	-	-	0.51	0.40	-	-
TT [86]	0.67	0.29	1.08	0.60	0.61	0.33	1.15	0.75	0.64	0.29	0.89	0.5
VDVAE [48]	-	-	-	-	-	-	-	-	0.59	0.20	0.80	0.50
PGGAN [178]	0.61	0.38	0.83	0.63	0.43	0.40	0.70	0.64	-	-	-	-
StyleGAN [179]	-	-	-	-	0.55	0.48	0.96	0.80	-	-	-	-
StyleGAN2 [181]	0.60	0.43	0.83	0.68	-	-	-	-	0.69	0.40	1.12	0.80
ProjGAN [314]	0.56	**0.53**	0.65	0.64	0.55	0.46	0.90	0.79	0.66	0.46	0.98	0.77
UT ($\tau = 1.0$)	0.70	0.42	**1.12**	0.73	0.64	0.38	1.27	0.81	0.69	0.48	1.06	0.77
UT ($\tau = 0.9$)	**0.71**	0.45	1.07	**0.74**	**0.67**	0.38	**1.51**	**0.83**	**0.73**	**0.48**	**1.20**	**0.80**

Table 4.1: Precision (P), Recall (R), Density (D), and Coverage (C) [202; 250; 306] for approaches trained on LSUN Churches, LSUN Bedroom, and FFHQ.

models generally trade precision and recall [181; 291]; since we also calculate FID with $\tau = 0.9$, we evaluate the effect on PRDC. In almost all cases this improves scores, indicating that more samples in data regions, increasing overlap.

FID

In Tab. 4.2 we calculate the Fréchet Inception Distance (FID) of samples from our models using torch-fidelity [265]. Using a fraction of the parameters of other Vector-Quantized image models, our approach achieves much lower FID.

Method	Params	Bed	Church	FFHQ
DDPM [135]	114M	6.36	7.89	-
DCT [253]	448M	6.40	7.56	-
VDVAE [48]	115M	-	-	28.5
TT [85; 86]	600M	6.35	7.81	9.6
I-BART [85]	2.1B	5.51	7.32	9.57
PGGAN [178]	47M	8.34	6.42	-
SGAN2 [181]	60M	2.35	3.86	3.8
ADM [66]	552M	1.90	-	-
ProjGAN [314]	106M	1.52	1.59	3.39
UT ($\tau = 1.0$)	145M	5.07	5.58	7.12
UT ($\tau = 0.9$)	145M	3.27	4.07	6.11

Table 4.2: FID comparison on FFHQ, LSUN Bedroom and Churches (lower is better).

Figure 4.5: Our method allows unconditional images larger than those seen during training to be generated by applying the denoising network to all subsets of the image, aggregating probabilities to encourage global continuity.

Higher Resolution

Fig. 4.5 shows samples generated at higher resolutions (up to 768×256) than the observed training data using the method described in Sec. 4.1.3 with $\tau = 0.8$. Even at larger scales we observe high-quality, diverse, and consistent samples.

4.2.2 Absorbing Diffusion

In this section we analyse the usage of absorbing diffusion for high-resolution image generation, determining how many sampling steps are required to obtain high-quality samples and ablating the components of our approach.

Sampling Speed

Our approach applies a diffusion process to a highly compressed image representation, meaning it is already 54× faster to sample from than DDPM when using Flash Attention [61] (ours: 1.3s, DDPM: 70s per image on a NVIDIA RTX 2080 Ti). However, since the absorbing diffusion model is trained to approximate $p(\boldsymbol{z}_0|\boldsymbol{z}_t)$ it is possible to speed the sampling process up further by skipping arbitrary numbers of time steps, unmasking multiple latents at once. In Tab. 4.3 we explore how sample quality is affected using a simple step skipping scheme: evenly skipping a constant number of steps so that the total number of steps meets some fixed computational budget. As expected, FID increases with fewer sampling steps. However, the increase in FID is minor relative to the improvement in sampling speed: our approach achieves similar FID to the equivalent autoregressive model using half the number

Steps	50	100	150	200	256
Church	6.86	6.09	5.81	5.68	5.58
Church ($\tau = 0.9$)	4.90	4.40	4.22	4.19	4.07
FFHQ	9.60	7.90	7.53	7.52	7.12
FFHQ ($\tau = 0.9$)	6.87	6.24	6.16	6.14	6.11

Table 4.3: Our approach allows sampling in much fewer steps with only minor FID increase.

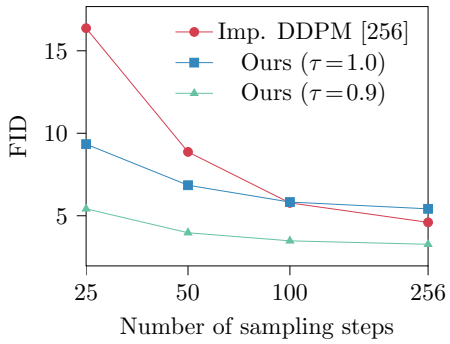


Figure 4.6: FID vs number of sampling steps on LSUN Bedroom.

of steps. With 50 sampling steps, our approach is 88× faster than DDPMs, and achieves lower FID (Fig. 4.6). Using a more sophisticated step selection scheme such as dynamic programming [373], FID could potentially be reduced further.

Autoregressive vs Absorbing DDPM

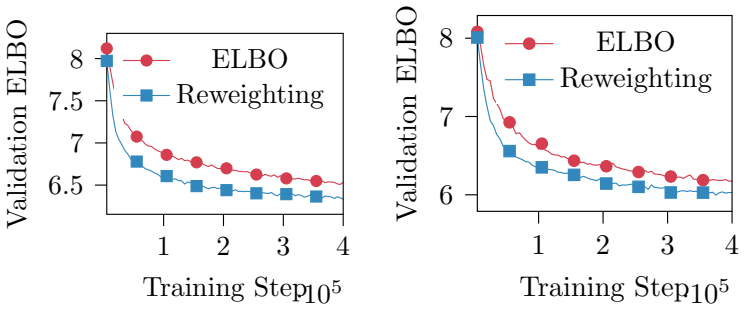
Tab. 4.4 compares the representation ability of our absorbing diffusion model with an autoregressive model, both utilising exactly the same Transformer architecture, but with the Transformer unconstrained in the diffusion case. On both datasets diffusion achieves lower FID, which is calculated in the image space. Validation NLL is evaluated in latent space (i.e. $-\log p(\boldsymbol{z})$) and again the diffusion model outperforms the autoregressive model despite being trained on a harder task with the same number of parameters, indicating that the diffusion models better approximate the prior distribution. Following previous works, early stopping was used to prevent autoregressive models from overfitting [85; 171]; increasing weight decay and dropout in some cases slightly improved validation NLL but caused FID to increase.

Method	Churches		FFHQ	
	FID ↓	NLL ↓	FID ↓	NLL ↓
*AR	13.23	6.67	9.47	6.65
*Absorbing	11.84	6.41	8.52	6.48
AR	5.93	6.24	8.15	6.18
Absorbing	**5.58**	**6.01**	**7.12**	**5.96**

Table 4.4: FID and validation latent NLL (in bpd) using the same Transformer. *=Default VQGAN

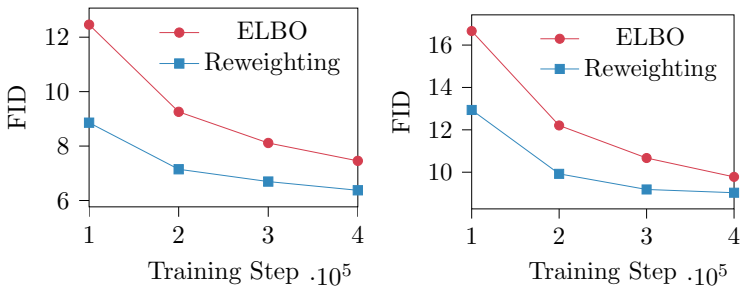
Reweighted ELBO

In Sec. 4.1.2 we proposed using a reweighted ELBO when training the diffusion model to reduce gradient variance. We evaluate this in Fig. 4.7 by comparing validation ELBO (calculated with Eqn. 4.3) during training for models trained directly on ELBO and our reweighting. The models trained on reweighted ELBO converge substantially faster, demonstrating that our reweighting is valid and simplifies optimisation. To further substantiate this and show that improvements extend to sample quality we compare models trained directly on ELBO and our reweighting in terms of FID in Fig. 4.8. The same trend is observed, with the models trained on the reweighting converging faster.



(a) LSUN Churches (b) FFHQ

Figure 4.7: Models trained with reweighting converge faster than models trained on ELBO.



(a) LSUN Churches (b) FFHQ

Figure 4.8: Models trained with our reweighted ELBO converge faster than models trained directly on ELBO.

4.2.3 Reconstruction Quality

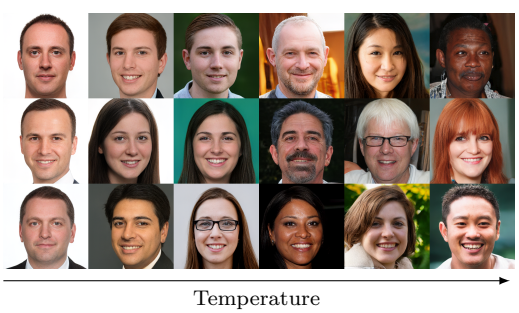
In Tab. 4.5 we evaluate the effect of DiffAug [402] and λ limiting on Vector-Quantized image models. While each technique individually can lead to worse FID due to imbalance between the generator and discriminator, we found combining techniques offered the most stability and improved FID across all datasets.

4.2.4 Sample Diversity

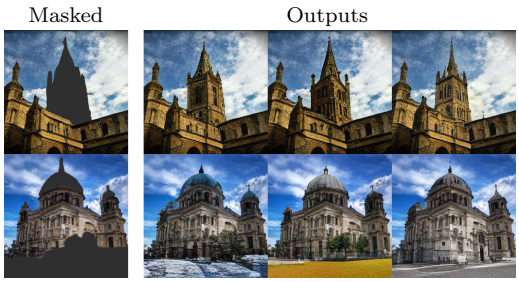
To improve sample quality, many generative models are sampled using a reduced temperature or by truncating distributions. For a multinomial distribution, the temperature τ is applied to the logits z_i to give the probability for x_i as $p(x_i) = \frac{\exp(z_i)/\tau}{\sum_j \exp(z_j/\tau)}$. This is problematic, as these methods amplify any biases in the dataset. We visualise the impact of temperature on sampling from a model trained on FFHQ in Fig. 4.9a. For very low temperatures the bias is obvious: samples are mostly front-facing white men with brown hair on solid white/black backgrounds. This is because when the temperature is small, samples come from the most common regions of the data distribution, which in this case is white men. Exactly how the

Modifications	Churches	FFHQ
Default	5.25	3.37
$\lambda_{\max} = 1$	8.67	4.72
DiffAug	5.16	6.57
Both	**2.70**	**3.12**

Table 4.5: Effect of proposed VQGAN changes on FID.



(a) Impact of sampling temperature on diversity. For small temperature changes it is unclear how bias changes.



(b) Our bidirectional approach allows local image editing by targeting regions to be changed (highlighted in grey).

Figure 4.9: Evaluation of practical use cases of our proposed generative model.

bias changes for more subtle temperature changes is less clear, which is problematic. Practitioners should be aware of this effect and it emphasises the importance of dataset balancing.

4.2.5 Image Editing

An additional advantage of using a bidirectional diffusion model to model the latent space is that image inpainting is possible. Since autoregressive models are conditioned only on the upper left region of the image, they are unable to edit internal masked image regions in a consistent manner. Diffusion models, on the other hand, allow masked regions to be placed at arbitrary locations. After a region has been highlighted, we mask corresponding latents, identify the starting time step by counting the number of masked latents, then continue the denoising process from that point. Examples of this process can be found in Fig. 4.9b.

4.2.6 Nearest Neighbours and Additional Samples

When training generative models, being able to detect overfitting is key to ensure the data distribution is well modelled. Overfitting is not detected by popular metrics such as FID, making overfitting difficult to identify in approaches such as GANs. With our approach we are able to approximate the ELBO on a validation set making it simple to prevent overfitting. In this section we demonstrate that our approach is not overfit by providing nearest neighbour images from the training dataset to

samples from our model, measured using LPIPS [396] (see Figs. 4.10 to 4.12). Additionally, Fig. 4.13 contains unconditional samples with resolutions larger than observed in the training data from a model trained on LSUN Bedroom.

4.2.7 Limitations

In our experiments we only tested our approach on 256×256 datasets; directly scaling to higher resolutions would require more GPU resources. However, future work using more efficient Transformer architectures [161] may alleviate this. Our method outperforms all approaches tested on FID except StyleGAN2 [181]; we find that the primary bottleneck is the Vector-Quantized image model, therefore more research is necessary to improve these discrete representations. Whilst our approach is trained for significantly less time than other approaches such as StyleGAN2, the stochastic training procedure means that more training steps are required compared to autoregressive approaches. Although when generating extra-large images the large context window made possible by the diffusion model encourages consistency, a reduced temperature is required, reducing diversity.

4.2.8 Quantitative Comparison with Previous Chapter

By using a multi-stage training approach, where the first stage uses a multi-scale architecture paired with a more expressive discrete latent space, the approach developed in this chapter allows for substantially higher sample quality than implicit GONs. This can be seen in Tab. 4.6 where on complex high resolution datasets, UT achieves significantly lower FID_{CLIP} than implicit GONs. Because UT is specifically designed for modelling high resolution data, only high resolution datasets are used for the comparison. However, this comes at the expense of the approach no longer being resolution agnostic, requiring a specific set up for a certain training data resolution as well as only being able to sample at a fixed single resolution. While it is possible to improve the quantitative score for implicit GONs by increasing the number of parameters, the lack of a multi-scale architecture makes this challenging due to the high GPU memory required to cache high resolution feature activations

Figure 4.10: Nearest neighbours for a model trained on LSUN Churches based on LPIPS distance. The left column contains samples from our model and the right column contains the nearest neighbours in the training set (increasing in distance from left to right).

Figure 4.11: Nearest neighbours for a model trained on FFHQ based on LPIPS distance. The left column contains samples from our model and the right column contains the nearest neighbours in the training set (increasing in distance from left to right).

Figure 4.12: Nearest neighbours for a model trained on LSUN Bedroom based on LPIPS distance. The left column contains samples from our model and the right column contains the nearest neighbours in the training set (increasing in distance from left to right).

Figure 4.13: Unconditional samples from a model trained on LSUN Bedroom larger than images in the training dataset.

for efficient backpropagation.

Method	FFHQ-256	Church-256	Bed-256
I-GON (Chapter 3)	44.65	72.29	66.40
UT (Chapter 4)	**3.05**	**5.52**	**4.53**

Table 4.6: FID_{CLIP} [203] evaluation against the implicit GON method proposed in the previous chapter.

One downside of this significant improvement in sample quality is the increase in compute time required for training and sampling, as can be seen in Tab. 4.7. The increase in training time is due to the increase in parameter count and increased stochasticity of training baused by the masked training objective; the increase in sampling time is due to the sampling process being inherently stochastic, whereas for implicit GONs it always only requires a single model call.

	NVIDIA RTX 2080Ti		NVIDIA A100	
Method	Training	Sampling (s)	Training	Sampling (s)
I-GON (Chapter 3)	<1 day	0.25s	<1 day	0.17s
UT (Chapter 4)	≈2 weeks	1.29s	< 1 week	0.69s

Table 4.7: Approximate time to train a model on a 256×256 dataset and the time to sample a single 256×256 image.

4.3 Discussion

While other classes of discrete generative model exist, they are less suitable for Vector-Quantized image modelling than discrete diffusion models: VAEs introduce prior assumptions about the latent space that can be limiting, in particular, continuous spaces may not be appropriate when modelling discrete data [28]; GAN training requires sampling from the generator meaning that gradients must be backpropagated through a discretistion procedure [257]; discrete normalising flows require functions to be invertible, significantly restricting function space [22; 143].

Another approach for modelling latent spaces using diffusion models is LSGMs [357], which model continuous latents with SDEs. However, our approach trains more than 15× faster thanks to the efficiency discrete approaches allow. There also exists a variety of different discrete diffusion methods [9; 146; 315]: ImageBART [85], developed concurrently with this work, models discrete latents using multinomial diffusion with separate autoregressive Transformers per diffusion step leading to slower training, inference, and substantially more parameters than our method.

Concurrent with our work, a number of similar approaches independently proposed using diffusion-like models to model VQGAN latents, these approaches are complementary to ours and distinct in a number of ways. VQ-Diffusion [117] use a combination of multinomial and absorbing diffusion to encourage the model to focus less on mask tokens. This, however, requires the use of an additional auxilliary objective function to improve stability, and in practice our approach achieves lower FID on the only shared dataset, FFHQ. MaskGIT [39] models discrete latents by learning to unmask tokens using a similar training scheme to ours; during sampling, tokens are unmasked based on the model's confidence. This approach allows sampling in very few steps, but the lack of theoretical justification makes it unclear how representative samples are. Latent Diffusion [298] relaxes the discrete assumption, using continuous diffusion parameterised by a convolutional U-Net to model latents of greater spatial size, but with lower dimensional codes. Both compressing spatially/depth-wise and discrete/continuous diffusion come with different trade-offs such as sampling time. Also of interest are non-autoregressive discrete methods for translation [97; 116; 302] and alignment [38; 304].

There are a number of avenues that would make for interesting future work based upon the models proposed in this paper: methods that scale diffusion models such as momentum [72], noise schedules [256], cascaded models [136; 305] and classifier guidance [66] may yield improved performance. Or, to improve discrete image representations, networks invariant to translation and rotation [182] or other more powerful generative models could be used. Finally, by conditioning on both text and discrete image representations, absorbing diffusion models could allow text-to-image generation and image captioning to be accomplished using a single model with faster

run-time than independent approaches [285; 288].

4.4 Conclusion

This chapter introduced a discrete diffusion probabilistic model prior capable of predicting Vector-Quantized image representations in parallel, overcoming the high sampling times, unidirectional nature and overfitting challenges associated with autoregressive priors. This approach makes no assumptions about the inherent ordering of latents by utilising an unconstrained Transformer architecture. Experimental results demonstrate the ability of the approach to generate diverse, high-quality images, optionally at resolutions exceeding the training samples.

In contrast to the approach proposed in Chapter 3, this approach is able to generate significantly higher quality samples, particularly so at high resolutions. However, these advantages come at the expense of longer sampling times (although faster than comparable approaches) and the ability to train on arbitrarily high resolution data is lost.

While these results offer notable improvements over similar approaches, there are a number of promising research directions in this area that could improve results: training times are long for both the autoencoder and diffusion stages, while improvements in GPU hardware have already significantly reduced these, improvements in architectures and types of discrete diffusion could further improve this; similarly, hardware improvements have already helped scaling to higher resolutions.

Nonetheless, dropping the ability to train/sample on arbitrary resolution data is a significant downside of such an approach. As such, Chapter 5 explores how complex high resolution data can be modelled with resolution agnostic models. Specifically, how the hierarchical unconstrained architectures paired with diffusion, a powerful generation approach, that proved successful in this chapter can be applied to the functional data domain. For this, a different approach to modelling functional data is required; as such, Chapter 5 explores how diffusion models can be defined and efficiently scaled for modelling functional representations.

CHAPTER 5

∞-Diff: Infinite Resolution Diffusion with Subsampled Mollified States

While the previous chapters introduce new approaches for generating high resolution data, both approaches have disadvantages. The approach proposed in Chapter 3 uses neural fields, which naturally allows scaling to high resolutions since coordinates are directly mapped to values, global context comes from conditioning on a single latent vector which is quite restrictive, and therefore limits sample quality. In contrast, the approach proposed in Chapter 4 obtains global context by using powerful spatial transformations thereby obtaining substantially higher sample quality, particularly on high resolution datasets. However, this comes at the expense of losing of the ability to sample at arbitrarily high resolutions. This chapter addresses this dichotomy, introducing an approach that allows very high quality data to be sampled at arbitrarily high resolutions.

In particular, this chapter builds on Gaussian diffusion models [135; 326] because they have become a dominant choice for data generation, offering stable training and the ability to high quality samples with excellent mode coverage. Scaling diffusion models to higher resolutions has been the topic of various recent research, with approaches including iteratively upsampling lower resolution images [136] and op-

erating in a compressed latent space [298]. In prior work, diffusion models have achieved these properties by assuming that data can be represented with a fixed uniform grid, allowing powerful spatial transformations to be used, similar to that used in Chapter 4.

Together with the approach developed in Chapter 3, a number of subsequent papers also developed methods to represent distributions of data points as continuous functions. These approaches similarly extend neural fields by conditioning the network on compressed latent vectors to describe single data points. Dupont et al. [79] first uses meta-learning to compress the dataset into latent conditional neural fields, then approximates the distribution of latents with a DDPM [135] or Normalizing Flow [293]. Zhuang et al. [409] design a diffusion model, with a small subset of coordinates used to provide context at each step. Finally, some approaches use hypernetworks to outputs the weight of neural fields including Dupont et al. [80] who define the hypernetwork as a generator in an adversarial framework, and Du et al. [76] who use manifold learning to represent the latent space of the hypernetwork. This chapter differs from these works in a number of aspects: 1. these works only train on low-resolution datasets, whereas this chapter focuses on developing a framework which can scale to substantially higher resolution, more complex datasets, particularly by enabling training on subsets of coordinates; 2. these works rely on compression to a global vector, this means that sample quality quality is substantially lower than achieved with the approach proposed in this chapter.

In this chapter, an approach is introduced that substantially improves upon the quality and scaling of infinite dimensional generative models (Fig. 5.1), reduc-

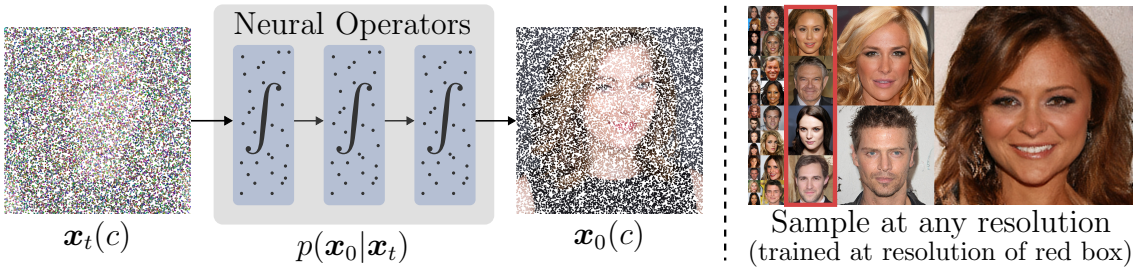


Figure 5.1: This chapter defines a diffusion process in an infinite dimensional image space by randomly sampling coordinates and training a model parameterised by neural operators to denoise at those coordinates.

Figure 5.2: Modelling data as functions allows sampling at arbitrary resolutions using the same model with different sized noise. Left to right: 64×64, 128×128, 256×256 (original), 512×512, 1024×1024.

ing the gap to finite-dimensional methods, while retaining the benefits of infinite dimensional models: subsampling coordinates to decouple memory/run-time from resolution, making scaling more computationally feasible, while also allowing training and sampling at arbitrary resolutions. This is achieved by designing a Gaussian diffusion model in an infinite dimensional state space. This chapter argues that latent-based neural fields cannot effectively be used to parameterise such diffusion models due to the reliance on compression, going against standard diffusion architecture design, with it also being impractical to compress states to latents at every step. Instead, this chapter proposes using non-local integral operators to model the denoising function, aggregating both global and local information in order to effectively denoise the data.

Specifically, this chapter proposes ∞-Diff, addressing these issues:

- A new Gaussian diffusion model defined in an infinite-dimensional state space is introduced that allows infinite resolution data to be generated (see Fig. 5.2).
- A powerful and scalable, function-space architecture is developed that operates directly on raw sparsely subsampled coordinates, enabling improvements in run-time and memory usage.
- ∞-Diff achieves state-of-the-art FID scores on multiple high-res image datasets, trained with up to $8\times$ subsampling, substantially outperforming prior infinite resolution generative models.

5.1 Finite Dimensional Diffusion Models

Before proceeding to introduce diffusion models in infinite dimensional spaces, for clarity we first recap finite dimensional diffusion models which assume that data lies on a uniform grid [135; 323]. The discrete time interpretation is formed by defining a forward process $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)$ that gradually adds noise to the data, $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$, over T steps, resulting in a sequence of latent variables $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ such that $q(\boldsymbol{x}_T) \approx \mathcal{N}(\boldsymbol{x}_T; \boldsymbol{0}, \boldsymbol{I})$. The reverse of this process can also be expressed as a Markov chain $p(\boldsymbol{x}_{0:T})$. Choosing Gaussian transition densities chosen to ensure these properties hold, the densities may be expressed as

$$q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}), \qquad q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}), \tag{5.1}$$

$$p(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t), \quad p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)), \tag{5.2}$$

where $0 < \beta_1, \ldots, \beta_T < 1$ is a pre-defined variance schedule and the covariance is typically of the form $\boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t) = \sigma_t^2 \boldsymbol{I}$. Aiding training efficiency, $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ can be expressed in closed form as $q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I})$ where $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ for $\alpha_s = 1 - \beta_s$. Training is possible by optimising the evidence lower bound on the negative log-likelihood which can be expressed as the KL-divergence between the forward process posteriors and backward transitions at each step

$$\mathcal{L} = \sum_{t \geq 1} \mathbb{E}_q \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \,\|\, p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)) \right] \tag{5.3}$$

$$= \sum_{t \geq 1} \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)\|_2^2 \right]. \tag{5.4}$$

for $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I})$, where $\tilde{\boldsymbol{\mu}}_t$ and $\tilde{\beta}_t$ can be derived in closed form.

5.2 Infinite Dimensional Diffusion Models

In this section we extend diffusion models to infinite dimensions in order to allow higher-resolution data to be trained on by subsampling coordinates during training and permit training/sampling at arbitrary resolutions. We argue that application of conditional neural fields to diffusion models is problematic due to the need to compress to a latent vector, adding complexity and opportunity for error, instead, the denoising function should be a non-local integral operator with no compression. A number of parallel works also developed diffusion models in infinite dimensions, including Kerrigan et al. [184]; Lim et al. [216]; and Franzese et al. [91]; we recommend also reading these works, which go further in the theoretical treatment, while ours focuses more on design and practical scaling.

To achieve this, we restrict the diffusion state space to a Hilbert space $\mathcal{H}$, elements of which, $x \in \mathcal{H}$, are functions, e.g. $x \colon \mathbb{R}^n \to \mathbb{R}^d$. Hilbert spaces are equipped with an inner product $\langle \cdot, \cdot \rangle$ and corresponding norm $\| \cdot \|_{\mathcal{H}}$. For simplicity we consider the case where $\mathcal{H}$ is the space of L^2 functions from $[0,1]^n$ to $\mathbb{R}^d$ although the following sections can be applied to other spaces. As such, a point in $\mathcal{H}$ could represent an image, audio signal, video, 3D model, etc. A Gaussian measure μ can be defined in $\mathcal{H}$ in terms of its characteristic function $\hat{\mu}$ [59],

$$\hat{\mu}(x) = \exp\left(i\langle x, m\rangle + \tfrac{1}{2}\langle Cx, x\rangle\right), \tag{5.5}$$

where the mean m lies in $\mathcal{H}$, $m \in \mathcal{H}$ and the covariance operator ($C : \mathcal{H} \to \mathcal{H}$) is self-adjoint (denoted $C = C^*$), non-negative (i.e. $C \geq 0$), and trace-class ($\int_{\mathcal{H}} \|x\|_{\mathcal{H}} d\mu(x) = \operatorname{tr}(C) < \infty$) [198]. For a Gaussian random element x with distribution μ, $x \sim \mathcal{N}(m, C)$. The Radon-Nikodym theorem states the existence of a density for a measure v absolutely continuous with respect to a base measure μ: for example, the density between two Gaussians is given by Minh [244]; see Kerrigan et al. [184]; Lim et al. [216] for more detail in the context of functional diffusion models.

Figure 5.3: Example Diffusion Processes. Mollified diffusion smooths diffusion states allowing the space to be more effectively modelled with continuous operators.

5.2.1 Mollification

When defining diffusion in infinite dimensions, it may seem natural to use white noise in the forwards process, where each coordinate is an independent and identically distributed Gaussian random variable; that is, $\mathcal{N}(0, C_I)$ where $C_I(z(s), z(s')) = \delta(s - s')$, using the Dirac delta function δ. However, this noise does not lie in $\mathcal{H}$ [59] with it not satisfying the trace-class requirement. Instead, obtain Gaussian noise in $\mathcal{H}$ by convolving white noise with a mollifier kernel $k(s) > 0$ corresponding to a linear operator T, giving $\mathcal{N}(0, TT^*)$, smoothing the white noise to lie in $\mathcal{H}$ [129]. To ensure one-to-one correspondence between kernel and noise, k must satisfy $\int_{\mathbb{R}^d} k(s)ds < \infty$ and $\int_{\mathbb{R}^d} k^2(s)ds < \infty$, making TT^* self-adjoint and non-negative. Considering k to be a Gaussian kernel with smoothing parameter $l > 0$, $h = Tx$ is given by

$$h(c) = \int_{\mathbb{R}^n} K(c - y, l)x(y)\,\mathrm{d}y, \text{ where } K(y, l) = \frac{1}{(4\pi l)^{\frac{n}{2}}} e^{-\frac{|y|^2}{4l}}. \tag{5.6}$$

5.2.2 Infinite Dimensional Mollified Diffusion

To formulate a diffusion model in $\mathcal{H}$, we must specify the transition distributions. However, irregularity in data points x can impact stability, leading to the model being unable to generalise across different subsampling rates/resolutions. This can be mitigated by careful hyperparameter tuning or, in our case, by also mollifying x (as with the previous noise mollification); see Fig. 5.3. While the necessity of this depends on the nature of x, we have included it for completeness. First, we define

the marginals

$$q(x_{t-1}|x_0) = \mathcal{N}(x_{t-1}; \sqrt{\bar{\alpha}_{t-1}}Tx_0, (1-\bar{\alpha}_{t-1})TT^*), \tag{5.7}$$

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}Tx_0, (1-\bar{\alpha}_t)TT^*), \tag{5.8}$$

where coefficients are the same as in Sec. 2.2. From this we are able to derive a closed form representation of the posterior,

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &= \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t TT^*), \quad \text{where} \\ \tilde{\mu}_t(x_t, x_0) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}Tx_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \quad \text{and} \quad \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t. \end{aligned} \tag{5.9}$$

This solution can be seen by considering Bishop and Nasrabadi [24, Eqns. 2.113 to 2.117],

$$q(x_t|x_0) = \mathcal{N}(x_t; \mu, \Lambda^{-1}), \tag{5.10}$$

$$q(x_{t-1}|x_0) = \mathcal{N}(x_{t-1}; A\mu + b, L^{-1} + A\Lambda^{-1}A^*), \tag{5.11}$$

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; Ax_t + b, L^{-1}). \tag{5.12}$$

From this we can immediately see that $\mu = \sqrt{\bar{\alpha}_t}Tx_0$ and $\Lambda^{-1} = (1-\bar{\alpha}_t)TT^*$. Additionally, $A\mu + b = A\sqrt{\bar{\alpha}_t}Tx_0 + b = \sqrt{\bar{\alpha}_{t-1}}Tx_0$, therefore we can modify the approach by Ho et al. [135], including T where relevant and set A, b and L^{-1} as

$$A = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}I, \qquad b = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}Tx_0, \qquad L^{-1} = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t TT^* \tag{5.13}$$

This can be shown to be correct by passing A, b, and L^{-1} into equations Eqns. 5.10 to 5.12, first:

$$A\mu + b = \frac{\sqrt{\bar{\alpha}_t}\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}Tx_0 + b \tag{5.14}$$

$$= \frac{\sqrt{\bar{\alpha}_t}\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1}) + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}Tx_0 \tag{5.15}$$

$$= \frac{\sqrt{\bar{\alpha}_{t-1}}(\alpha_t - \bar{\alpha}_t + 1 - \alpha_t)}{1-\bar{\alpha}_t}Tx_0 = \sqrt{\bar{\alpha}_{t-1}}Tx_0, \tag{5.16}$$

and secondly,

$$L^{-1} + A\Lambda^{-1}A^* = \frac{(1-\bar{\alpha}_{t-1})(1-\alpha_t)}{1-\bar{\alpha}_t}TT^* + \frac{\alpha_t(1-\bar{\alpha}_{t-1})^2(1-\bar{\alpha}_t)}{(1-\bar{\alpha}_t)^2}TT^* \tag{5.17}$$

$$= \frac{1-\bar{\alpha}_{t-1}-\alpha_t+\bar{\alpha}_t+\alpha_t-2\bar{\alpha}_t+\alpha_t\bar{\alpha}_{t-1}^2}{1-\bar{\alpha}_t} = (1-\bar{\alpha}_{t-1})TT^* \tag{5.18}$$

$$= \frac{(1-\bar{\alpha}_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}TT^* = (1-\bar{\alpha}_{t-1})TT^*, \tag{5.19}$$

which together form $q(x_{t-1}|x_0)$.

Defining the reverse transitions similarly as mollified Gaussian densities, $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t,t), \sigma_t^2 TT^*)$, then we can parameterise $\mu_\theta : \mathcal{H}, \mathbb{R} \to \mathcal{H}$ to directly predict x_0. The loss defined in Eqn. 5.4 can be extended to infinite dimensions [277]. This can be derived by calculating the Kullback-Leibler divergence in infinite dimensions [277], for conciseness we ignore additive constants throughout since they do not affect optimisation,

$$\mathcal{L}_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| T^{-1}(\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)) \right\|_{\mathcal{H}}^2 \right]. \tag{5.20}$$

To find a good representation for μ_θ we expand out $\tilde{\mu}_t$ as defined in Eqn. 5.9 in the above loss giving

$$\mathcal{L}_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| T^{-1} \left(\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} Tx_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t - \mu_\theta(x_t, t) \right) \right\|_{\mathcal{H}}^2 \right]. \tag{5.21}$$

From this we can see that one possible parameterisation is to directly predict x_0, that is,

$$\mu_\theta(x_t, t) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} T f_\theta(x_t, t) + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t. \tag{5.22}$$

This parameterisation is interesting because when sampling, we can use the output of f_θ to directly obtain an estimate of the unmollified data. Additionally, when calculating the loss $\mathcal{L}_{t-1}$, all T and T^{-1} terms cancel out meaning there are no concerns with reversing the mollification during training, which can be numerically unstable. To see this, we can further expand out Eqn. 5.21 using the parameterisation of μ_θ

defined in Eqn. 5.22,

$$\begin{aligned}\mathcal{L}_{t-1} = \mathbb{E}_q \Bigg[\frac{1}{2\sigma_t^2} \Bigg\| T^{-1} \Bigg(& \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} T x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t \\ & - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} T f_\theta(x_t, t) - \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t \Bigg) \Bigg\|_{\mathcal{H}}^2 \Bigg] \end{aligned} \tag{5.23}$$

$$= \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} f_\theta(x_t, t) \right\|_{\mathcal{H}}^2 \right] \tag{5.24}$$

$$= \mathbb{E}_q \left[\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{2\sigma_t^2(1-\bar{\alpha}_t)} \left\| x_0 - f_\theta(x_t, t) \right\|_{\mathcal{H}}^2 \right]. \tag{5.25}$$

Using this parameterisation, we can sample from $p_\theta(x_{t-1}|x_t)$ as

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t + T \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} f_\theta(x_t, t) + \sigma_t \xi \quad \text{where} \quad \xi \sim \mathcal{N}(0, TT^*). \tag{5.26}$$

Alternatively, we can parameterise $\tilde{\mu}$ to predict the noise ξ rather than x_0, which was found by Ho et al. [135] to yield higher sample quality. To see this, we can write Eqn. 5.8 as $x_t(x_0, \xi) = \sqrt{\bar{\alpha}_t} T x_0 + \sqrt{1-\bar{\alpha}_t}\xi$. Expanding out Eqn. 5.20 with this gives the following loss,

$$\mathcal{L}_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| T^{-1} \Big(\tilde{\mu}(x_t, x_0) - \mu_\theta(x_t, t) \Big) \right\|_{\mathcal{H}}^2 \right] \tag{5.27}$$

$$= \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| T^{-1} \Big(\tilde{\mu}(x_t(x_0, \xi), \frac{1}{\sqrt{\bar{\alpha}_t}} T^{-1}(x_t(x_0, \xi) - \sqrt{1-\bar{\alpha}_t}\xi)) - \mu_\theta(x_t, t) \Big) \right\|_{\mathcal{H}}^2 \right] \tag{5.28}$$

$$= \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| T^{-1} \left(\frac{1}{\sqrt{\alpha_t}} \left(x_t(x_0, \xi) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \xi \right) - \mu_\theta(x_t, t) \right) \right\|_{\mathcal{H}}^2 \right]. \tag{5.29}$$

As such, this prompts the following parameterisation of μ_θ, and therefore the loss

can be further simplified to the following,

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left[x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} f_\theta(x_t, t)\right], \tag{5.30}$$

$$\mathcal{L}_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}} T^{-1}\left(\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} f_\theta(x_t, t) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\xi\right)\right\|_{\mathcal{H}}^2\right]. \tag{5.31}$$

In this case T^{-1} is a linear transformation that does not affect the minima. In addition to this, we can remove the weights as suggested by Ho et al. [135], giving the following proxy loss,

$$\mathcal{L}_{t-1}^{\text{simple}} = \mathbb{E}_q\left[\left\|f_\theta(x_t, t) - \xi\right\|_{\mathcal{H}}^2\right]. \tag{5.32}$$

An alternative parameterisation which can train more reliably is v-prediction [148; 307]; we experimented with this parameterisation but found ξ-prediction to yield higher quality samples. The concurrent work by Kerrigan et al. [184] showed that in the infinite-dimensional limit, the loss will be finite only for specific choices of $\tilde{\beta}_t$, while Lim et al. [216] found similar only for specific parameterisations of μ_θ; however, since the loss is Monte-Carlo approximated, this is not problematic in practice.

Data Mollification By mollifying the training data x_0 to ensure regularity, resulting samples are similarly regular; directly predicting x_0 would give an estimate of the original data, but by predicting ξ we are only able to sample Tx_0. However, in the case of the Gaussian mollifier kernel with adequate boundary conditions, the existence of the inverse T^{-1} is clear if we consider the Fourier transform of $x(c)$, denoted $\hat{x}(\omega)$, then the Gaussian convolution can be defined by $\hat{h}(\omega) = e^{-\omega^2 t}\hat{x}(\omega)$. And so Tx is one-to-one on any class of Fourier transformable functions, with Tx being bounded ensuring uniqueness and therefore invertibility [167]. Explicitly, the inverse is given by $\hat{x}(\omega) = e^{\omega^2 t}\hat{h}(\omega)$ [155]. However, inverting is ill-conditioned, with arbitrarily small changes (for instance by floating point error) destroying smoothness [155]. In this case, the Wiener filter can for instance be used as an approximate inverse, defined as $\tilde{x}(\omega) = \frac{e^{-\omega^2 t}}{e^{-2(\omega^2 t)} + \epsilon^2}\hat{h}(\omega)$, where ϵ is an estimate of the inverse SNR [23].

5.3 Parameterising the Diffusion Process

In order to model the denoising function in Hilbert space, there are certain properties that is essential for the class of learnable functions to satisfy so as to allow training on infinite resolution data:

1. Can take as input points positioned at arbitrary coordinates.
2. Generalises to different numbers of input points than trained on, sampled on a regular grid.
3. Able to capture both global and local information.
4. Scales to very large numbers of input points, i.e. efficient in terms of runtime and memory.

Recent diffusion models often use a U-Net [300] consisting of a convolutional encoder and decoder with skip-connections between resolutions allowing both global and local information to be efficiently captured. Unfortunately, U-Nets function on a fixed grid making them unsuitable. However, we can take inspiration to build an architecture satisfying the desired properties.

5.3.1 Neural Operators

Neural Operators [195; 213] are a framework designed for efficiently solving partial differential equations by learning to directly map the PDE parameters to the solution in a single step. However, more generally they are able to learn a map between two infinite dimensional function spaces making them suitable for parameterising an infinite dimensional diffusion model.

Let $\mathcal{X}$ and $\mathcal{S}$ be separable Banach spaces representing the spaces of noisy and denoised data respectfully; a neural operator is a map $\mathcal{F}_\theta \colon \mathcal{X} \to \mathcal{S}$. Since $x \in \mathcal{X}$ and $s \in \mathcal{S}$ are both functions, we only have access to pointwise evaluations. Let $\boldsymbol{c} \in \binom{D}{m}$ be an m-point discretisation of the domain and assume we have observations $x(\boldsymbol{c}) \in \mathbb{R}^{m \times d}$. To be discretisation invariant, the neural operator may be evaluated at any $c \in D$, potentially $c \notin \boldsymbol{c}$, thereby allowing a transfer of solutions between different discretisations i.e. satisfying properties 1 and 2. Each operator layer is built using a non-local integral kernel operator, $\mathcal{K}(x; \phi)$, parameterised by neural

network κ_ϕ, aggregating information spatially,

$$(\mathcal{K}(x;\phi)v_l)(c) = \int_D \kappa_\phi(c, b, x(c), x(b))v_l(b)\,\mathrm{d}b, \qquad \forall c \in D. \tag{5.33}$$

Deep networks can be built in a similar manner to conventional methods, by stacking layers of linear operators with non-linear activation functions, $v_0 \mapsto v_1 \mapsto \cdots \mapsto v_L$ where $v_l \mapsto v_{l+1}$ is defined as

$$v_{l+1}(c) = \sigma(Wv_l(c) + (\mathcal{K}(x;\phi)v_l)(c)), \qquad \forall c \in D, \tag{5.34}$$

for pointwise linear transformation $W\colon \mathbb{R}^d \to \mathbb{R}^d$ and non-linear activation function $\sigma\colon \mathbb{R} \to \mathbb{R}$. One approach of interest is the Fourier Neural Operator (FNO) [214], defined as

$$(\mathcal{K}(x;\phi)v_l)(c) = \mathcal{F}^{-1}\left(R_\phi \cdot (\mathcal{F}v_t)\right)(c), \qquad \forall c \in D, \tag{5.35}$$

where $\mathcal{F}$ is the Fourier transform, and R_ϕ is learned transformation in Fourier space. When coordinates lie on a regular grid, the fast fourier transform can be used, making FNOs fast and scalable.

5.3.2 Multi-Scale Architecture

While neural operators which satisfy all the required properties (1-4) exist, such as Galerkin attention [35] (a softmax-free linear attention operator) and MLP-Mixers [343], scaling beyond small numbers of coordinates is still challenging due to the high memory costs associated with caching activations for backpropagation. Instead we design a U-Net inspired multi-scale architecture (Fig. 5.4) that aggregates information locally and globally at different points.

In a continuous setting, there are two main approaches to downsampling: (1) selecting a subset of coordinates [370] and (2) interpolating points to a regularly spaced grid [287]. We found that with repeated application of (1), approximating integral operators on non-uniformly spaced grids with very few points did not perform nor generalise well, likely due to the high variance. On the other hand, while working with a regular grid removes some sparsity properties, issues with variance

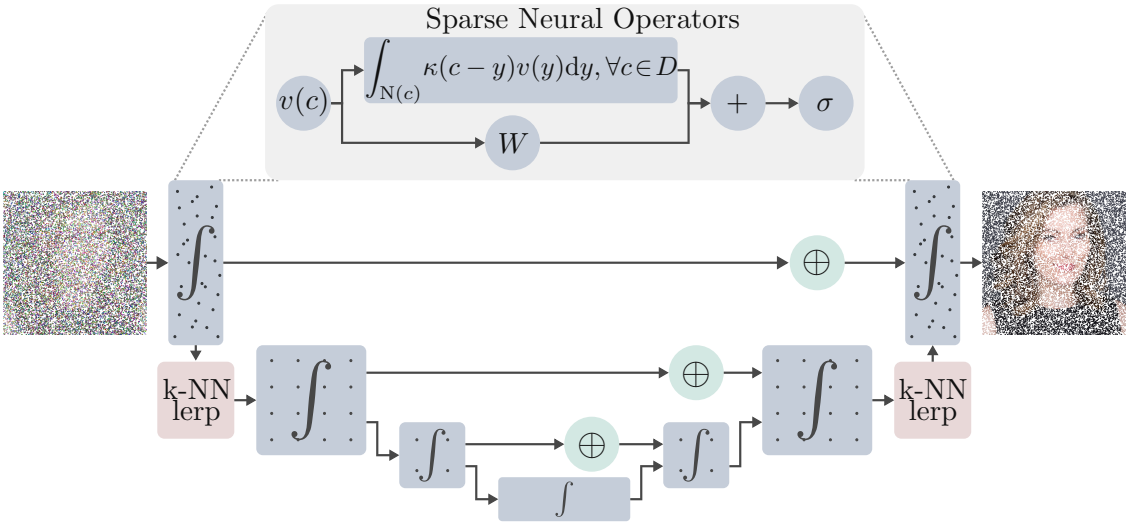


Figure 5.4: ∞-Diff uses a hierarchical architecture that operates on irregularly sampled functions at the top level to efficiently capture fine details, and on fixed grids at the other levels to capture global structure. This approach allows scaling to complex high resolution data.

are much lower. As such, we use a hybrid approach with sparse operators applied on the raw irregularly sampled data to local regions; after this points are interpolated to a regular grid and a grid-based architecture is applied in order to aggregate global information; if the regular grid is of sufficiently high dimension, this combination should be sufficient. While an FNO [214; 287] architecture could be used, we achieved better results with dense convolutions [256], with sparse operators used for resolution changes.

5.3.3 Efficient Sparse Operators

At the sparse level we use convolution operators [195], finding them to be more performant than Galerkin attention, with global context no longer essential due to the multiscale architecture. In this case, the operator is defined using a translation invariant kernel restricted to the local neighbourhood of each coordinate, $N(c)$,

$$x(c) = \int_{\mathrm{N}(c)} \kappa(c-y)v(y)\,\mathrm{d}y, \qquad \forall c \in D. \tag{5.36}$$

We restrict κ to be a depthwise kernel due to the greater parameter efficiency for large kernels (particularly for continuously parameterised kernels) and finding that

they are more able to generalise when trained with fewer sampled coordinates; although the sparsity ratio is the same for regular and depthwise convolutions, because there are substantially more values in a regular kernel, there is more spatial correlation between values. When a very large number of sampled coordinates are used, fully continuous convolutions are extremely impractical in terms of memory usage and run-time. In practice, however, images are obtained and stored on a discrete grid. As such, by treating images as high dimensional, but discrete entities, we can take advantage of efficient sparse convolution libraries [52; 55], making memory usage and run-times much more reasonable. Specifically, we use TorchSparse [336], modified to allow depthwise convolutions. Wang and Golland [370] proposed using low discrepancy coordinate sequences to approximate the integrals due to their better convergence rates. However, we found uniformly sampled points to be more effective, likely because the reduced structure results in points sampled close together which allows high frequency details to be captured more easily.

5.4 Experiments

In this section we demonstrate that the proposed mollified diffusion process modelled with neural operator based networks and trained on coordinate subsets are able to generate high quality, high resolution samples. We explore the properties of this approach including discretisation invariance, the impact of the number of coordinates during training, and compare the sample quality of our approach with other infinite dimensional generative models. We train models on 256×256 datasets, FFHQ [179] and LSUN Church [385], as well as CelebA-HQ [178]; unless otherwise specified models are trained on $1/4$ of pixels (to fit in memory), randomly selected.

When training diffusion models, very large batch sizes are necessary due the high variance [148], making training on high resolution data on a single GPU impractical. To address this, we use the diffusion autoencoder framework [279] which reduces stochasticity by dividing the generation process into two stages. To encode data we use the first half of our proposed architecture (Fig. 5.4), which still operates on sparsely sampled coordinates. When sampling, we use the deterministic DDIM

interpretation with 100 steps.

Implementation Details All 256×256 models are trained on a single NVIDIA A100 80GB GPU using automatic mixed precision. Optimisation is performed using the Adam optimiser [189] with a batch size of 32 and learning rate of 5×10^{-5}; each model being trained to optimise validation loss. Each model is trained as a diffusion autoencoder to reduce training variance, allowing much smaller batch sizes thereby permitting training on a single GPU. A latent size of 1024 is used and the latent model architecture and diffusion hyperparameters are the same as used by Preechakul et al. [279]. In image space, the diffusion model uses a cosine noise schedule [256] with 1000 steps. Mollifying is performed with Gaussian blur with a variance of 1.0.

For the image-space architecture, 3 sparse residual convolution operator blocks are used on the sparse data. Each of these consist of a single depthwise sparse convolution layer with kernel size 7 and 64 channels with the output normalised by the total number of coordinates in each local region, followed by a three layer MLP. Modulated layer normalisation [10; 256], which adds an extra affine transformation with the offset and scale determined by a linear projection of the diffusion time step embedding, is used to normalise and condition on the diffusion time step. These blocks use larger convolution kernels than typically used in diffusion model architectures to increase the number of coordinates present in the kernel when a small number of coordinates are sampled. Using large kernel sizes paired with MLPs has found success in recent classification models such as ConvNeXt [224].

As mentioned in Sec. 5.3.2, for the grid-based component of our architecture we experimented with a variety of U-Net shaped fourier neural operator [214; 287] architectures; although these more naturally allow resolution changes at those levels, we found this came with a substantial drop in performance. This was even the case when operating at different resolutions. As such, we use the architecture used by Nichol and Dhariwal [256] fixed at a resolution of 128×128; the highest resolution uses 128 channels which is doubled at each successive resolution up to a maximum factor of 8; attention is applied at resolutions 16 and 8, as is dropout, as suggested by

Figure 5.5: Samples from ∞-Diff models trained on sets of randomly subsampled coordinates.

Hoogeboom et al. [148]. Although this places more emphasis on the sparse operators for changes in sampling resolution, we found this approach to yield better sample quality across the board. As a result, the full model has 500M parameters.

Sample Quality Samples from our approach can be found in Fig. 5.5 which are high quality, diverse, and capture fine details. In Tab. 5.1 we quantitatively compare with other approaches that treat inputs as infinite dimensional data, as well as more traditional approaches that assume data lies on a fixed grid. As proposed by Kynkäänniemi et al. [203], we calculate FID [128] using CLIP features [285] which is better correlated with human perception of image quality. Our approach scales

Method	CelebAHQ-64	CelebAHQ-128	FFHQ-256	Church-256
Finite-Dimensional				
CIPS [5]	-	-	5.29	10.80
StyleSwin [389]	-	3.39	3.25	8.28
StyleGAN2 [181]	-	**2.20**	**2.35**	**6.21**
Infinite-Dimensional				
D2F [79]	40.4*	-	-	-
DPF [409]	13.21*	-	-	-
GEM [76]	14.65	23.73	35.62	87.57
GASP [80]	9.29	27.31	24.37	37.46
∞-Diff	**4.57**	**3.02**	**3.87**	**10.36**

Table 5.1: FID_{CLIP} [203] evaluation against finite-dimensional methods as well as other infinite-dimensional approaches which are trained on coordinate subsets. *=Inception FID.

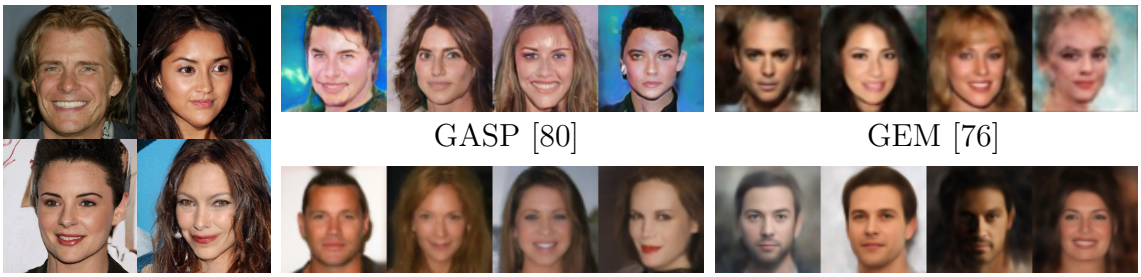


Figure 5.6: Qualitative comparison with other infinite dimensional approaches.

to high resolutions much more effectively than the other function-based approaches as evidenced by the substantially lower scores. Visual comparison between samples from our approach and other function-based approaches can be found in Fig. 5.6 where samples from our approach can be seen to be higher quality and display more details without blurring or adversarial artefacts. All of these approaches are based on neural fields [382] where coordinates are treated independently; in contrast, our approach uses neural operators to transform functions using spatial context thereby allowing more details to be captured. Both GASP [80] and GEM [76] rely on compressed latent-conditional hypernetworks which makes efficiently scaling difficult. D2F [79] relies on a deterministic compression stage which loses detail due to the finite vector size. DPF [409] uses small fixed sized coordinate subsets as global context with other coordinates modelled implicitly, thereby causing blur.

Discretisation Invariance In Fig. 5.2 we demonstrate the discretisation invariance properties of our approach. After training on random coordinate subsets from 256×256 images, we can sample from this model at arbitrary resolutions which we show at resolutions from 64×64 to 1024×1024 by initialising the diffusion with different sized noise. We experimented with (alias-free) continuously parameterised kernels [299] but found bi-linearly interpolating kernels to be more effective. At each resolution, even exceeding the training data, samples are consistent and diverse. In Fig. 5.7 we analyse how the number of sampling steps affects quality at different sampling resolutions.

Architecture Analysis In Tab. 5.2 we ablate the impact of various architecture choices against the architecture described in Sec. 5.3.2, matching the architecture as

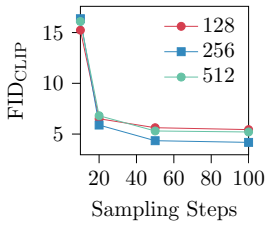


Figure 5.7: FID_{CLIP} at various steps & resolutions.

Architecture	FID
Sparse Downsample	85.99
Nonlinear Kernel	24.49
Quasi Monte Carlo	7.63
Regular Convs	5.63
∞-Diff	**4.75**

Table 5.2: Architectural component ablations in terms of FID_{CLIP}.

Ratio	FID_{CLIP}	Speedup
1	3.15	1.0×
1/2	4.12	1.0×
1/4	4.75	1.3×
1/8	6.48	1.6×

Table 5.3: Impact of coordinate sparsity on quality for FFHQ 128. FID_{CLIP} calculated with 10k samples.

closely as possible. In particular, sparse downsampling (performed by randomly subsampling coordinates; we observed similar with equidistant subsampling, Qi et al., 2017) fails to capture the distribution. Similarly using a spatially nonlinear kernel (Eqn. 5.33), implemented as conv, activation, conv, does not generalise well unlike linear kernels (we observed similar for softmax transformers, Kovachki et al., 2023).

Coordinate Sparsity One factor influencing the quality of samples is the number of coordinates sampled during training; fewer coordinates means fewer points from which to approximate each integral. We analyse the impact of this in Tab. 5.3, finding that as expected, performance decreases with fewer coordinates; however, this effect is fairly minimal. With fewer coordinates also comes substantial speedup and memory savings; at 256×256 with $4\times$ subsampling the speedup is $1.4\times$.

Super-resolution The discretisation invariance of the proposed approach makes superresolution a natural application. We evaluate this in a simple way, passing a low resolution image through the encoder, then sampling at a higher resolution; see Fig. 5.8 where it is clear that more details have been added. A downside of this specific approach is that information is lost in the encoding process; however, this could potentially by improved by incorporating DDIM encodings [325].

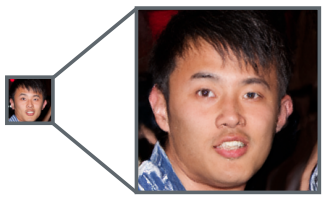


Figure 5.8: Super-resolution

Inpainting Inpainting is possible with mollified diffusion (Fig. 5.9), using reconstruction guidance [137], $x_{t-1} \leftarrow x_{t-1} - \lambda \nabla_{x_t} \|m \odot (\tilde{\mu}_0(x_t, t) - T\bar{x})\|_2^2$ for inpainting mask m, learned estimate of Tx_0, $\tilde{\mu}_0$, and image to be inpainted $\bar{x}$. The diffusion autoencoder framework gives an additional level of control when inpainting since the reverse diffusion process can be applied to encodings from a chosen time step t_s, allowing control over how different the inpainted region is from the original image.

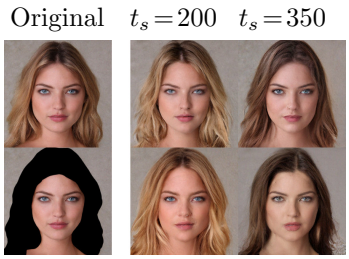


Figure 5.9: Inpainting.

Additional Samples and Nearest Neighbours Here, we provide additional samples from our models to visually assess quality (Figs. 5.10 to 5.12). Detecting overfitting is crucial when training generative models. Scores such as FID are unable to detect overfitting, making identifying overfitting difficult in approaches such as GANs. Because diffusion models are trained to optimise a bound on the likelihood, training can be stopped to minimise validation loss. As further evidence we provide nearest neighbour images from the training data to samples from our model, measured using LPIPS [396] (Figs. 5.13 to 5.15).

Quantitative Comparison with Previous Chapters ∞-Diff addresses the challenges associated with scaling implicit GONs, the method introduced in Chapter 3 by using stochastic rather than deterministic reconstruction from latents, as well as using a more scalable multi-scale architecture that operates over images rather than being coordinate-wise independent. As such, ∞-Diff achieves substantially higher image quality than implicit GONs, as can be seen in Tab. 5.4. Here, only datasets with resolution greater than 32×32 are used. Furthermore, ∞-Diff achieves more comparable quantitative metrics on 256×256 datasets to UT, proposed in Chapter 4, despite being being resolution agnostic like implicit GONs. Additionally, it is worth noting that since ∞-Diff is a diffusion model, it does not suffer from mode collapse; in contrast, UT which uses a VQGAN results in partial mode collapse. However, this is not captured by the FID metric.

Figure 5.10: Non-cherry picked, CelebA-HQ 256×256 samples.

Figure 5.11: Non-cherry picked, LSUN Church 256×256 samples.

Figure 5.12: Non-cherry picked, FFHQ 256×256 samples.

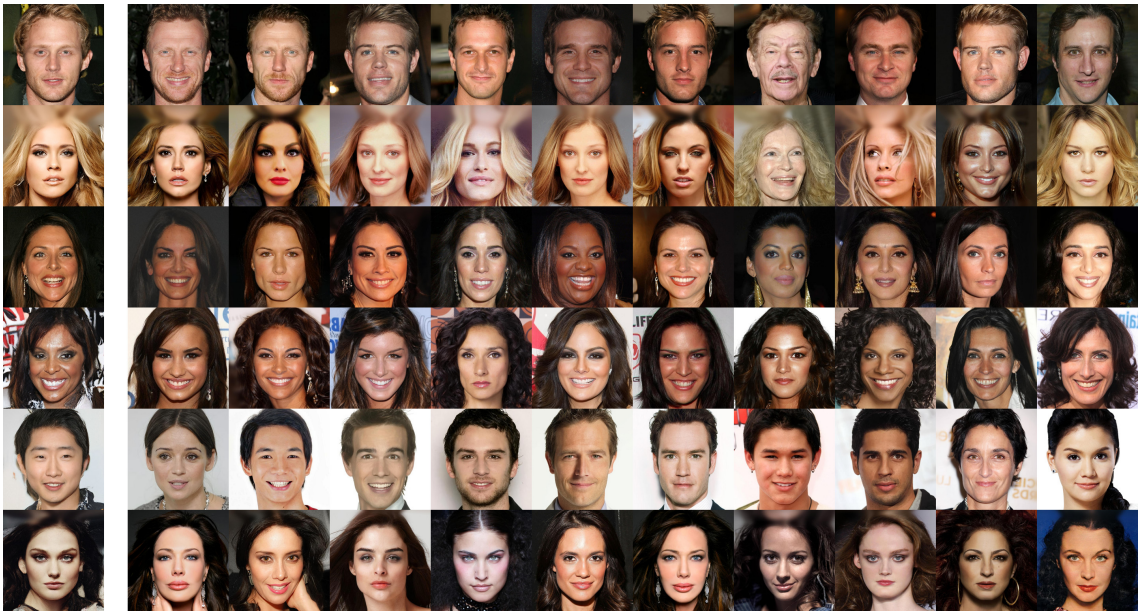


Figure 5.13: Nearest neighbours for a model trained on CelebA-HQ based on LPIPS distance. The left column contains samples from our model and the right column contains the nearest neighbours in the training set (increasing in distance from left to right)

Figure 5.14: Nearest neighbours for a model trained on LSUN Church based on LPIPS distance. The left column contains samples from our model and the right column contains the nearest neighbours in the training set (increasing in distance from left to right)

Figure 5.15: Nearest neighbours for a model trained on FFHQ based on LPIPS distance. The left column contains samples from our model and the right column contains the nearest neighbours in the training set (increasing in distance from left to right)

Method	CelebAHQ-64	CelebAHQ-128	FFHQ-256	Church-256	Bed-256
I-GON (Chapter 3)	21.41	36.72	44.65	72.29	66.40
UT (Chapter 4)	-	-	**3.05**	**5.52**	**4.53**
∞-Diff (Chapter 5)	**4.57**	**3.02**	3.87	10.36	12.41

Table 5.4: FID_{CLIP} [203] evaluation against the methods proposed in the previous chapters. UT is designed for resolutions 256×256 and higher so is not evaluated on CelebAHQ-64 and CelebAHQ-128.

A downside of the reduced mode collapse of diffusion over VQGAN meaning a larger batch size is necessary, as well as the overhead involved with operating on a continuous signal is the increased memory requirements. As such, it is not possible to train ∞-Diff on the NVIDIA 2080Ti GPU used in previous chapters. Additionally, more time is required to train ∞-Diff than UT (Tab. 5.5). It is worth noting, however, that both approaches require substantially less compute than comparable methods, with StyleSwin [389] for instance being trained for 2 weeks using 8× more GPUs.

	NVIDIA RTX 2080Ti		NVIDIA A100	
Method	Training	Sampling (s)	Training	Sampling (s)
I-GON (Chapter 3)	<1 day	0.25s	<1 day	0.17s
UT (Chapter 4)	≈2 weeks	1.29s	< 1 week	0.69s
∞-Diff (Chapter 5)	OOM	3.74s	1-2 weeks	2.29s

Table 5.5: Approximate time to train a model on a 256×256 dataset and the time to sample a single 256×256 image.

5.5 Discussion

There are a number of interesting directions to improve our approach including more powerful/efficient neural operators, more efficient sparse methods, better integral approximations, and improved UNet design [376]. Having demonstrated that diffusion models can be trained with 8× subsampling rates, we believe there is substantial room for further performance gains. Also of interest are recent works which speed up diffusion sampling by iteratively upsampling throughout the backwards process, requiring a separate model for each resolution [166; 393]; the resolution invariance of our approach permits this with a single model.

Recent diffusion advances are also complementary to our approach, these include consistency models [330], stochastic interpolants [2], Schrödinger bridges [62], critically-damped diffusion [71], architecture improvements [148], and faster solvers [228]. Similar to our mollified diffusion, blurring has been used to improve diffusion [141; 295]. Similar to GASP [80], other neural field GAN approaches exist such as CIPS [5] and Poly-INR [319]; however, these approaches use convolutional discriminators requiring all coordinates on a fixed grid, preventing scaling to infinite resolutions. Also of relevance are Neural Processes [84; 94] which learn distributions over functions similar to Gaussian Processes; however, these approaches address conditional inference, whereas we construct an unconditional generative model for substantially more complex data.

Concurrent with this work, other papers independently proposed diffusion mod-

els in infinite dimensions [12; 91; 124; 184; 216; 276; 409], these approaches are complementary to ours and distinct in a number of ways. While our work focuses on the practical development necessary to efficiently model complex high-dimensional data, these papers instead focus more on theoretical foundations, typically being only applied to simple data (e.g. Gaussian mixtures and MNIST). Of particular interest, Kerrigan et al. [184] also develop diffusion models in Hilbert space, going further than our work in foundational theory, including more on the requirements to obtain well-posed models, as well as considering different function spaces; [216] develop infinite-dimensional diffusion defined as SDEs; and Franzese et al. [91] prove the existence of the backwards SDE. Unlike our work, these approaches make use of conditional neural fields or operate on uniform grids of coordinates, whereas our approach operates on raw sparse data, enabling better scaling. The closest to this work in terms of scaling is Diffusion Probabilistic Fields [409] which denoises coordinates independently using small coordinate subsets for context; this is much more restrictive than our approach and resolutions are much smaller than ours (up to 64×64).

5.6 Conclusion

This chapter introduced a new type of denoising diffusion model defined in an infinite-dimensional Hilbert space with transition densities represented by non-local integral operators, able to generate high-quality samples at arbitrary resolutions. Despite only observing subsets of pixels during training, sample quality is competitive with state-of-the-art models trained on all pixels at once, with the benefit of faster training and lower memory requirements due to the sparsity of the training samples.

Prior infinite dimensional approaches, as well as the approach introduced in Chapter 3 use latent conditional neural fields which is a severely limiting constraint. In contrast, the approach introduced in this chapter operates directly on the raw sparse data, offering significant performance advantages by not treating all coordinates independently, showing that neural operators are a capable alternative to

neural fields.

Evaluating the approach on a variety of complex high resolution datasets, namely CelebA-HQ [178], FFHQ [179], and LSUN Church [385] demonstrates superior sample quality over other infinite dimensional approaches, as evidenced visually in Fig. 5.6 and quantitatively by the substantially lower FID scores.

There are a number of key directions to improve this approach. One component is the usage of sparse convolution operators; these are crucial for the success of the approach, without which sample quality would be much lower. Increasing the sparsity of the training data requires using larger convolution kernels to ensure there are sufficiently many points from which to approximate the convolution; however, this also affects runtime particularly at test time. As such, the development of new efficient and scalable sparse neural operators which can be approximated with few coordinates will surely aid future scaling of this approach.

CHAPTER 6

Conclusion

Recent developments in deep generative modelling has led to an influx in applications in a variety of areas from pure generative models such as text, image, video, and text to other modality generation, to more general applications including expanding training datasets, active learning, and representation learning. A major challenge in this field is efficiently scaling generative models to higher dimensional data, allowing for instance higher resolution images, longer videos, and high sample-rate audio to be synthesised. The majority of prior work focuses on scaling through training and architectural improvements; in many cases using a brute force approach that requires a large number of GPUs making it impractical for smaller institutions/individuals to effectively replicate.

This thesis began by introducing a comprehensive overview of the wide assortment of generative modelling approaches, old and new, in order to provide different perspectives into generative modelling. The assortment of properties associated with each approach are explored, including methods to scale to high resolutions, and how hybrids between approaches can trade off advantages/disadvantages. To address the problems associated with scaling to very high dimensional data, this thesis addressed the problem from a direct perspective, with each following chapter

Table 6.1: Comparison between methods introduced in this thesis.

Method	Train Speed	Sample Speed	Num. Params.	Resolution Scaling
GONs (Chapter 3)	★★★★★	★★★★★	★★★★★	★★☆☆☆
UT (Chapter 4)	★★☆☆☆	★★☆☆☆	★☆☆☆☆	★★★★☆
∞-Diff (Chapter 5)	★☆☆☆☆	★★☆☆☆	★☆☆☆☆	★★★★★

introducing a new approach that is able to more efficiently represent very high dimensional data, taking a particular focus on the infinite dimensional regime. The following section outlines the main contributions of the thesis.

6.1 Contributions

Chapter 2 reviews the extremely broad field of generative modelling, extensively covering the various different classes of approaches. This chapter aims to bring the reader up to date with current research, understanding the history leading up to current innovations, and what prompts them. Specifically, energy-based models, diffusion and score-based models, variational autoencoders, generative adversarial networks, autoregressive models, and normalizing flows are covered. This chapter draws connections between these classes, with each addressing problems associated with the others, characterising disadvantages in the process. Within Chapter 2, Tab. 2.1 compared a broad range of deep generative models. This is expanded in Tab. 6.1, adding the approaches developed within this thesis. From this table, it is clear to see that compared to GONs (Chapter 3), UT (Chapter 4) and ∞-Diff (Chapter 5) trade off training speed, sampling speed, and use more parameters, in order to scale to higher resolutions and obtain higher image quality. While GONs practically can generate arbitrarily high resolutions, meaning it is arguable that they should have a 5 star rating, because at these resolutions the quality is lacking, a 2 star rating has been assigned.

Chapter 3 proposes using empirical Bayes to approximate the latent posterior of implicit generative models, rather than using an explicit encoding network as is typical. This is achieved by initialising a latent vector with zeros, then using the gradient of the log-likelihood of the data with respect to that vector as new latent points. As such, this approach can be applied to neural fields, allowing a space of functions to be learned without the need for hypernetworks, encoders, nor meta-

learning. The approach is evaluated by comparing against comparable autoencoders and variational autoencoders on a variety of simple datasets including MNIST [206], CIFAR-10 [196], and CelebA [225] where quantitative results are shown to be comparable or better despite the lack of an explicit encoder. Visually samples are diverse and can represent fine details; however, similar to comparable approaches they are blurry as a result of compressing to small latents. The impact of various components are analysed including latent dimension, activation function, and number of gradient steps. Since neural fields can be evaluated at arbitrary coordinates, this approach can be used for super-resolution; despite never observing high resolution data, when evaluated at higher resolutions, samples show considerable detail demonstrating that the underlying data is well understood.

Addressing the blurry nature of samples produced in Chapter 3 as a result of using coordinate-wise functions conditioned on compressed latents, Chapter 4 introduces an approach for high quality and high resolution image generation through the use of a powerful diffusion-based prior that utilises global context. Rather than compressing to a single latent vector, images are compressed to sets of discrete codes, where each code is a vector that contains substantial amounts of information. A discrete diffusion model parameterised by an unconstrained Transformer is used to represent the distribution of these discrete representations thereby achieving better sample quality while also having a much faster runtime than prior approaches. This approach is evaluated on much more complex and higher resolution datasets, FFHQ [179], LSUN Church and LSUN Bedroom [385], with state-of-the-art results demonstrated in terms of precision and recall metrics [250] as well as competitive FID [128]. While this approach can also be used to generate samples higher than the resolution of the training data, by generating compressed representations with larger spatial sizes, these can be thought of as extensions of images, or out-paintings, rather than true higher resolution samples. Additionally, in Fig. 4.9a, this chapter explored how the temperature parameter impacts the diversity of samples. In future work it would be interesting to explore how what biases are present and how they can be identified by modifying such a parameter.

Chapter 5 draws together the findings of the previous chapters, proposing an ap-

proach that is able to represent infinite dimensional data, while using spatial transformations to provide global context rather than relying on compression, thereby producing very high quality samples. This is achieved by using a mollified-state diffusion model which smooths states to be continuous, thereby allowing transition densities to be represented by neural operators. In order to efficiently represent infinte dimensional transition densities, this chapter introduces a hybrid operator architecture that operates on raw sparse data in the infinite dimensional regime, while discretising and operating on uniform grids to efficiently capture global context. In contrast to the approach in Chapter 4, the approach in this chapter is trained on sparsely sampled coordinates allowing better scaling; and in contrast to the approach in Chapter 3 and prior infinite dimensional approaches, operating directly on the raw data rather than relying on compression allows much more expressive models to be built. These advantages are corroborated by the substantially lower FID scores achieved compared to prior approaches.

6.2 Limitations and Future Work

While each approach introduced in this thesis makes strides to allow higher dimensional data to be efficiently modelled, solving challenges and allowing new problems to be solved, no single approach offers a complete solution. This section discusses the limitations of the approaches introduced and associated challenges, how combination with concurrent work enables some of these to be at least partially addressed, as well as suggesting some promising avenues of future research.

6.2.1 Further Scaling

The primary topic of this thesis is efficient scaling of generative models to high-dimensional data while providing favourable properties such as good mode coverage and high quality sample, with Chapters 3 and 5 addressing this by enabling coordinate subsampling to allow Monte-Carlo approximating the loss function thereby enabling higher resolution data to be more efficiently modelled. Nonetheless, there is substantial room to further improve in this regard to allow ever higher dimensional

data to be modelled.

∞-Diff proposes that the main limitation with neural field methods such as GONs is the over reliance on latent compression, and as such, integral operators are a better choice. In contrast, concurrent work by Bauer et al. [16] propose increasing the representation capacity of latents by arranging them spatially, demonstrating that this approach enables more complex data to be represented. This approach allows more easily sampling at arbitrary coordinates due to the independence properties of neural fields, with ∞-Diff relying on efficient sparse convolution libraries; however, this comes at the expense of having to approximate latents during training in a slow iterative optimisation procedure, necessitates a 2-stage model due to the need to also represent the latents, and in practice for the best performing cases, the latents only minimally compress the raw data raising questions of how this approach would scale. Moreover, this approach cannot easily be used to parameterise diffusion models meaning that integral operators are much more applicable in this highly important scenario.

As such, an important area of future work is developing neural integral operators than are able to efficiently scale to very large numbers of arbitrarily positioned sparse coordinates. However, recent work on more efficient neural operators primarily focuses on uniformly spaced training data [194]. For more effective sparse methods, approaches such as RIN [160] are a promising alternative, decoupling computation from data dimensionality by moving computation to a set of expressive latents which can then more efficiently "read" and "write" into the raw data using cross-attention. Such an approach would be able to operate on arbitrarily positioned coordinates, while scaling considerably more favourably than full self-attention based approaches.

Finally, in future work it would be interesting to apply ∞-Diff to other modalities such as video and 3D. The additional dimensions provide extra regularity which should allow for even more aggressive subsampling during training and therefore provide even greater levels of speedup and memory reductions.

6.2.2 Training/Sampling Times

The three approaches introduced in this thesis offer three different degrees of sampling tames, making different trade-offs in the process in terms of mode coverage and sample quality. Both Unleashing Transformers and ∞-Diff make use of diffusion models therefore use slow iterative sampling. There has been considerable recent research improving this area. For example, Lu et al. [228] introduce an ODE sampler for diffusion models requiring around 10 steps. However, this approach is not applicable to discrete state space like in Unleashing Transformers and is an important area for future research. There has also been considerable research into distilling models to allow single or few step sampling [330]. Also of interest are recent methods which regularise the mappings between distributions so that paths are straighter and therefore easier to integrate over [345]. None of these approaches are perfect, with the number of steps required still often scaling with data dimension. Developing a model which has the excellent quality and mode coverage of diffusion models, while can be sampled with a single or few steps is an important topic of future work since it will drastically reduce energy usage for training/sampling.

6.2.3 Mode Coverage

As discussed in Chapter 1, a key property to strive for when developing generative models is good mode coverage. Likelihood-based models such as the VAE and diffusion methods in Chapters 3 and 5 excel in this space since directly maximising (a bound on) the likelihood ensures that all training data points are considered in the final trained model. In contrast, to enable efficient scaling, the approach in Chapter 4 uses a VAE-GAN based compression scheme to aid scaling. While the GAN component is crucial for high sample quality [86], it results in a minor level of "local" mode collapse around data points. In many applications this is a fairly inconsequential downside, but in some cases such as medical imaging the corresponding minor hallucinations have the potential to be catastrophic. While this can be reduced to a negligible level, by reducing the compression level, future work could instead make use of stochastic compression techniques [279], but more

work is needed to make this approach as scalable.

6.2.4 Applications

The developments in this thesis to allow efficient scaling of deep generative models opens up opportunities for new applications. For example, the ability to extract useful feature representations without an encoding network, proposed in Chapter 3, has been applied in a recent work by Karnewar et al. [176] to enable 3D generative models to be trained on only 2D image views. Here, a GON-style encoder module was used to extract local 3D feature representations from multiple 2D image views which can then be used in a 3D diffusion model for generation. Furthermore, the scalability, unconstrained nature, and additional in-built regularisation of the approach proposed in Chapter 4 has been taken advantage of in works by Corona-Figueroa et al. [57]; Jiang et al. [165]; and Lin et al. [217]. More generally, recent improvements in the quality of generative models has made applications such as generating private datasets [88], solving inverse problems [211], learning 3D models from 2D images [177], digital art [153], semantic segmentation [13], semantic correspondence [394], and image translation [351]. The ability of generative models to better understand the underlying data than supervised models paired with the exceptional flexibility that has allowed many of these applications opens up the potential for solving many new ill-posed problems that have until now been considered unsolvable. In particular, improvements in scalablity opens up potential for application to difficult modalities such as high-resolution 3D and video, as well as situations where megapixel, or even gigapixel images are necessary to capture sufficient detail.

6.3 Ethical Considerations

First and foremost, it is important to consider and keep in mind the wide assortment of positive use cases of generative models that prompts this area of research, from digital art to medical image computing that have been discussed throughout this thesis, and in particular in the preceding section. Advances in theoretical generative modelling consequently have the potential to have wide reaching impact; improving

properties such as sample quality, diversity, and of particular importance to this thesis, greater scaling, opens up opportunities for new applications. Beyond this, the improvements in scaling introduced in this thesis also result in greater levels of efficiency that benefit smaller research groups, opening up the ability to train high resolution generative models to a wider audience. Greater efficiency also has the effect of reducing energy consumption during training and sampling, which is of particular importance at the current time when climate change is having such a major impact on the planet.

While the work in this thesis does not directly contribute to this, it would be remiss to not discuss the negative aspects of generative model development. In particular, there are a wide variety of malicious applications which should be considered, including fake news, phishing attacks, and social bots for propaganda/disinformation. Additionally, while not intentionally malicious, generative models can be used, for instance, for artistic tasks, directly affecting livelihoods through commissions, particularly since these models can be used to mimic writing/music/artwork styles, or even explicitly plagiarise existing works. While these abilities have the potential to improve productivity, or even directly replace components of many jobs, it is a political decision how to accordingly adjust society, with existing precedent in this regard generally not a positive one [106].

At the start of this thesis it is mentioned how, unlike supervised methods, generative models can be trained on unlabelled data, thus opening up the vast amounts of information available on the internet as training data. While strictly true, this does not consider the substantial bias in such data which is then reflected in the trained models. This affects capabilities in different languages, results in misinformation, explicit racism, sexism, and all biases present on the internet. Recent models have addressed this alignment problem through the use of reinforcement learning from human feedback (RLHF) [410] or direct preference optimisation (DPO) [286], but this brings back a labour intensive task and all the associated problems, particularly for reinforcement learning, which is notably inefficient due to lack of gradients.

Major corporations' pursuit of profit has resulted in each company separately developing these massive models, each with private datasets with unknown biases

and copyright violations, requiring substantial labour, and consuming extraordinary amounts of energy and therefore resulting in substantial CO_2 emissions.

It is also important to remember that these models are only possible because of the extremely vast amount of data put online in good faith by the general public, together with decades of research from not-for-profit researchers; yet these models, training which are beyond the capabilities of many states, are kept locked behind closed doors, with access typically only provided through paid APIs and are used as one more tool to lock users into a single ecosystem [73]. This is all to say that for safety, the environment, workers, understanding and diagnosing problems, assessing ability, to allow fine-tuning (rather than re-training), the availability of high quality open source models and well thought out interoperability are crucial to democratising the future of "AI".

Bibliography

[1] Guillaume Alain, Yoshua Bengio, Li Yao, Jason Yosinski, éric Thibodeau-Laufer, Saizheng Zhang, and Pascal Vincent. GSNs: generative stochastic networks. *Information and Inference*, 5(2):210–249, 2016. ISSN 2049-8764. doi: 10.1093/imaiai/iaw003.

[2] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic Interpolants: A Unifying Framework for Flows and Diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[3] Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy. Fixing a Broken ELBO. In *ICML*, 2018.

[4] Brian DO Anderson. Reverse-Time Diffusion Equation Models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[5] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image Generators with Conditionally-Independent Pixel Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14278–14287, 2021.

[6] Michael Arbel, Liang Zhou, and Arthur Gretton. Generalized Energy Based Models. In *ICLR*, 2021.

[7] Martin Arjovsky and Leon Bottou. Towards Principled Methods for Training Generative Adversarial Networks. In *ICLR*, 2017.

[8] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *arXiv:1701.07875*, 2017.

[9] Jacob Austin, Daniel Johnson, Jonathan Ho, Danny Tarlow, and Rianne van den Berg. Structured Denoising Diffusion Models in Discrete State-Spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

[10] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer Normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[11] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic Variational Video Prediction. In *ICLR*, 2018.

[12] Lorenzo Baldassari, Ali Siahkoohi, Josselin Garnier, Knut Solna, and Maarten V de Hoop. Conditional Score-Based Diffusion Models for Bayesian Inference in Infinite Dimensions. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-Efficient Semantic Segmentation with Diffusion Models. *arXiv preprint arXiv:2112.03126*, 2021.

[14] Serguei Barannikov, Ilya Trofimov, Grigorii Sotnikov, Ekaterina Trimbach, Alexander Korotin, Alexander Filippov, and Evgeny Burnaev. Manifold Topology Divergence: a Framework for Comparing Data Manifolds. *arXiv preprint arXiv:2106.04024*, 2021.

[15] Matthias Bauer and Andriy Mnih. Resampled Priors for Variational Autoencoders. In *AISTATS*, pages 66–75, 2019.

[16] Matthias Bauer, Emilien Dupont, Andy Brock, Dan Rosenbaum, Jonathan Schwarz, and Hyunjik Kim. Spatial Functa: Scaling Functa to ImageNet Classification and Generation. *arXiv preprint arXiv:2302.03130*, 2023.

[17] Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jrn-Henrik Jacobsen. Invertible Residual Networks. In *ICML*, 2019.

[18] Yoshua Bengio. Estimating or propagating gradients through stochastic neurons, 2013.

[19] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A Neural Probabilistic Language Model. *JMLR*, 3:1137–1155, 2003. ISSN ISSN 1533-7928.

[20] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized Denoising Auto-Encoders as Generative Models. *Advances in Neural Information Processing Systems*, 26, 2013.

[21] Rianne van den Berg, Leonard Hasenclever, Jakub M. Tomczak, and Max Welling. Sylvester Normalizing Flows for Variational Inference. In *UAI*, 2018.

[22] Rianne van den Berg, Alexey A Gritsenko, Mostafa Dehghani, Casper Kaae Sønderby, and Tim Salimans. IDF++: Analyzing and Improving Integer Discrete Flows for Lossless Compression. In *International Conference on Learning Representations*, 2021.

[23] Jan Biemond, Reginald L Lagendijk, and Russell M Mersereau. Iterative Methods for Image Deblurring. *Proceedings of the IEEE*, 78(5):856–883, 1990.

[24] Christopher M Bishop and Nasser M Nasrabadi. *Pattern Recognition and Machine Learning*. Springer, 2006.

[25] Mikoaj Bikowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *ICLR*, 2018.

[26] Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the Latent Space of Generative Networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, volume 80, pages 600–609, 2018.

[27] Ali Borji. Pros and Cons of GAN Evaluation Measures: New Developments. *arXiv preprint arXiv:2103.09396*, 2021.

[28] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. *arXiv:1511.06349*, 2016.

[29] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations*, 2019.

[30] Tom B. Brown et al. Language Models are Few-Shot Learners. *arXiv:2005.14165*, 2020.

[31] Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders. In *ICLR*, 2016.

[32] Ryan Burgert, Kanchana Ranasinghe, Xiang Li, and Michael S Ryoo. Peekaboo: Text to Image Diffusion Models are Zero-Shot Segmentors. *arXiv preprint arXiv:2211.13224*, 2022.

[33] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding Disentangling in β-VAE. *Advances in Neural Information Processing Systems*, 30, 2017.

[34] Luis A. Caffarelli and Mario Milman. *Monge Ampre Equation: Applications to Geometry, Optimization.* American Mathematical Soc., 1999.

[35] Shuhao Cao. Choose a Transformer: Fourier or Galerkin. *Advances in Neural Information Processing Systems*, 34:24924–24940, 2021.

[36] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting Training Data from Diffusion Models. In *Proceedings of the 32nd USENIX Conference on Security Symposium*, pages 5253–5270, 2023.

[37] Miguel A Carreira-Perpinan and Geoffrey E Hinton. On Contrastive Divergence Learning. In *AISTATS*, volume 10, pages 33–40, 2005.

[38] William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly. Imputer: Sequence Modelling via Imputation and Dynamic Programming. In *International Conference on Machine Learning*, pages 1403–1413. PMLR, 2020.

[39] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked Generative Image Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.

[40] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. Your GAN is Secretly an Energy-based Model and You Should Use Discriminator Driven Latent Sampling. *Advances in Neural Information Processing Systems*, 33, 2020.

[41] Jianfei Chen, Cheng Lu, Biqi Chenli, Jun Zhu, and Tian Tian. VFlow: More Expressive Generative Flows with Variational Data Augmentation. In *ICML*, 2020.

[42] Ricky T. Q. Chen, Jens Behrmann, David K. Duvenaud, and Joern-Henrik Jacobsen. Residual Flows for Invertible Generative Modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

[43] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural Ordinary Differential Equations. *Advances in neural information processing systems*, 31, 2018.

[44] Tianrong Chen, Guan-Horng Liu, and Evangelos Theodorou. Likelihood training of schrödinger bridge using forward-backward sdes theory. In *International Conference on Learning Representations*, 2022.

[45] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-Supervised GANs via Auxiliary Rotation Loss. In *IEEE CVPR*, 2019.

[46] Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational Lossy Autoencoder. In *5th International Conference on Learning Representations, ICLR*, 2017.

[47] Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. PixelSNAIL: An Improved Autoregressive Generative Model. *ICML*, 2017.

[48] Rewon Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. In *International Conference on Learning Representations*, 2021.

[49] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating Long Sequences with Sparse Transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[50] Francois Chollet. Xception: Deep Learning With Depthwise Separable Convolutions. In *IEEE CVPR*, 2017.

[51] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking Attention with Performers. In *ICLR*, 2021.

[52] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[53] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv:1412.3555*, 2014.

[54] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *4th International Conference on Learning Representations, ICLR*, 2016.

[55] Spconv Contributors. Spconv: Spatially Sparse Convolution Library. `https://github.com/traveller59/spconv`, 2022.

[56] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing Bijectivity Constraints with Continuously Indexed Normalising Flows. In *ICML*, 2020.

[57] Abril Corona-Figueroa, Sam Bond-Taylor, Neelanjan Bhowmik, Yona Falinie A Gaus, Toby P Breckon, Hubert PH Shum, and Chris G Willcocks. Unaligned 2D to 3D Translation with Conditional Vector-Quantized Code Diffusion using Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14585–14594, 2023.

[58] Chris Cremer, Xuechen Li, and David Duvenaud. Inference Suboptimality in Variational Autoencoders. In *ICML*, 2018.

[59] Giuseppe Da Prato and Jerzy Zabczyk. *Stochastic Equations in Infinite Dimensions*. Cambridge university press, 2014.

[60] Bin Dai and David Wipf. Diagnosing and Enhancing VAE Models. In *International Conference on Learning Representations*, 2019.

[61] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[62] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion Schrödinger Bridge with Applications to Score-Based Generative Modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021.

[63] Nicola De Cao, Ivan Titov, and Wilker Aziz. Block Neural Autoregressive Flow. In *UAI*, 2019.

[64] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. *Advances in Neural Information Processing Systems*, 28, 2015.

[65] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, 2019.

[66] Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat GANs on Image Synthesis. *Advances in Neural Information Processing Systems*, 34, 2021.

[67] Sander Dieleman, Aäron van den Oord, and Karen Simonyan. The Challenge of Realistic Music Generation: Modelling Raw Audio at Scale. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[68] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-Linear Independent Components Estimation. In *ICLR Workshop*, 2015.

[69] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *ICLR*, 2017.

[70] Laurent Dinh, Jascha Sohl-Dickstein, Razvan Pascanu, and Hugo Larochelle. A RAD approach to deep mixture models. In *ICLR Workshop*, 2019.

[71] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-Based Generative Modeling with Critically-Damped Langevin Diffusion. In *International Conference on Learning Representations*, 2022.

[72] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-Based Generative Modeling with Critically-Damped Langevin Diffusion. In *International Conference on Learning Representations*, 2022.

[73] Cory Doctorow. *The Internet Con: How to Seize the Means of Computation.* Verso Books, 2023.

[74] Alexey Dosovitskiy and Thomas Brox. Generating Images with Perceptual Similarity Metrics based on Deep Networks. *Advances in Neural Information Processing Systems*, 2016.

[75] Yilun Du and Igor Mordatch. Implicit Generation and Generalization in Energy-Based Models. *Advances in Neural Information Processing Systems*, 33, 2019.

[76] Yilun Du, Katie Collins, Josh Tenenbaum, and Vincent Sitzmann. Learning Signal-Agnostic Manifolds of Neural Fields. *Advances in Neural Information Processing Systems*, 34:8320–8331, 2021.

[77] Leo L. Duan. Transport Monte Carlo. *arXiv:1907.10448*, 2020.

[78] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented Neural ODEs. *Advances in Neural Information Processing Systems*, 32, 2019.

[79] Emilien Dupont, Hyunjik Kim, SM Ali Eslami, Danilo Jimenez Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. In *International Conference on Machine Learning*, pages 5694–5725. PMLR, 2022.

[80] Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative Models as Distributions of Functions. In *International Conference on Artificial Intelligence and Statistics*, pages 2989–3015. PMLR, 2022.

[81] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural Spline Flows. *Advances in Neural Information Processing Systems*, 32, 2019.

[82] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Cubic-Spline Flows. In *ICML Workshop*, 2019.

[83] Conor Durkan, Iain Murray, and George Papamakarios. On Contrastive Learning for Likelihood-free Inference. In *ICML*, 2020.

[84] Vincent Dutordoir, Alan Saul, Zoubin Ghahramani, and Fergus Simpson. Neural Diffusion Processes. In *International Conference on Machine Learning*, pages 8990–9012, 2023.

[85] Patrick Esser, Robin Rombach, Andreas Blattmann, and Björn Ommer. Imagebart: Bidirectional Context with Multinomial Diffusion for Autoregressive Image Synthesis. *arXiv preprint arXiv:2108.08827*, 2021.

[86] Patrick Esser, Robin Rombach, and Bjrn Ommer. Taming Transformers for High-Resolution Image Synthesis. *arXiv:2012.09841*, 2021. URL `http://arxiv.org/abs/2012.09841`.

[87] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the Manifold Hypothesis. *Journal of the American Mathematical Society*, 29(4): 983–1049, 2016.

[88] Virginia Fernandez, Pedro Sanchez, Walter Hugo Lopez Pinaya, Grzegorz Jacenków, Sotirios A Tsaftaris, and Jorge Cardoso. Privacy Distillation: Reducing Re-Identification Risk of Multimodal Diffusion Models. *arXiv preprint arXiv:2306.01322*, 2023.

[89] Chris Finlay, Joern-Henrik Jacobsen, Levon Nurbekyan, and Adam Oberman. How to Train Your Neural ODE: the World of Jacobian and Kinetic Regularization. In *ICML*, 2020.

[90] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[91] Giulio Franzese, Simone Rossi, Dario Rossi, Markus Heinonen, Maurizio Filippone, and Pietro Michiardi. Continuous-Time Functional Diffusion Processes. *Advances in Neural Information Processing Systems*, 36, 2024.

[92] Ruiqi Gao, Erik Nijkamp, Diederik P. Kingma, Zhen Xu, Andrew M. Dai, and Ying Nian Wu. Flow Contrastive Estimation of Energy-Based Models. In *IEEE CVPR*, 2020.

[93] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning Energy-Based Models by Diffusion Recovery Likelihood. In *ICLR*, 2021.

[94] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *ICML Workshop*, 2018.

[95] TD Gedeon. Stochastic Bidirectional Training. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, volume 2, pages 1968–1971, 1998.

[96] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. MADE: Masked Autoencoder for Distribution Estimation. In *ICML*, 2015.

[97] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-Predict: Parallel Decoding of Conditional Masked Language Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, 2019.

[98] Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From Variational to Deterministic Autoencoders. In *International Conference on Learning Representations*, 2020.

[99] Ilka H Gleibs. Are all "research fields" equal? rethinking practice for the use of data from crowdsourcing market places. *Behavior Research Methods*, 49(4): 1333–1342, 2017.

[100] Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. Its Raw! Audio Generation with State-Space Models. In *International Conference on Machine Learning*, pages 7616–7633. PMLR, 2022.

[101] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, 27, 2014.

[102] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT press, 2016.

[103] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J. Cree. Regularisation of neural networks by enforcing Lipschitz continuity. *Machine Learning*, 110(2):393–416, 2021. ISSN 1573-0565. doi: 10.1007/s10994-020-05929-w.

[104] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving Robustness using Generated

Data. *Advances in Neural Information Processing Systems*, 34:4218–4233, 2021.

[105] Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Exposing the Implicit Energy Networks behind Masked Language Models via Metropolis–Hastings. In *International Conference on Learning Representations*, 2022.

[106] David Graeber. Bullshit Jobs. *E mploi*, page 131, 2018.

[107] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models. In *International Conference on Learning Representations*, 2019.

[108] Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, and Richard Zemel. Learning the Stein Discrepancy for Training and Evaluating Energy-Based Models without Sampling. In *ICML*, 2020.

[109] Will Grathwohl, Kuan-Chieh Wang, Jrn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One. In *ICLR*, 2020.

[110] Will Grathwohl, Kevin Swersky, Milad Hashemi, David Duvenaud, and Chris J Maddison. Oops I Took A Gradient: Scalable Sampling for Discrete Distributions. In *International Conference on Machine Learning*, 2021.

[111] Will Sussman Grathwohl, Jacob Jin Kelly, Milad Hashemi, Mohammad Norouzi, Kevin Swersky, and David Duvenaud. No MCMC for me: Amortized sampling for fast and stable training of energy-based models. In *ICLR*, 2021.

[112] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. DRAW: A Recurrent Neural Network For Image Generation. In *ICML*, 2015.

[113] Paulina Grnarova, Kfir Y Levy, Aurelien Lucchi, Nathanael Perraudin, Ian Goodfellow, Thomas Hofmann, and Andreas Krause. A Domain Agnostic Measure for Monitoring and Evaluating GANs. *Advances in Neural Information Processing Systems*, 32, 2019.

[114] Paulina Grnarova, Yannic Kilcher, Kfir Y Levy, Aurelien Lucchi, and Thomas Hofmann. Generative Minimization Networks: Training GANs Without Competition. *arXiv:2103.12685*, 2021.

[115] Aditya Grover, Manik Dhar, and Stefano Ermon. Flow-GAN: Combining Maximum Likelihood and Adversarial Learning in Generative Models. In *AAAI*, 2018.

[116] Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein Transformer. *Advances in Neural Information Processing Systems*, 32, 2019.

[117] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector Quantized Diffusion Model for Text-to-Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022.

[118] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved Training of Wasserstein GANs. *Advances in Neural Information Processing Systems*, 30, 2017.

[119] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. PixelVAE: A Latent Variable Model for Natural Images. In *ICLR*, 2017.

[120] Ishaan Gulrajani, Colin Raffel, and Luke Metz. Towards GAN Benchmarks Which Require Generalization. In *ICLR*, 2019.

[121] Michael Gutmann and Aapo Hyvärinen. Noise-Contrastive Estimation: A New Estimation Principle for Unnormalized Statistical Models. In *AISTATS*, pages 297–304, 2010.

[122] David Ha and Jrgen Schmidhuber. World Models. *arXiv:1803.10122*, 2018.

[123] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *ICLR*, 2020.

[124] Paul Hagemann, Lars Ruthotto, Gabriele Steidl, and Nicole Tianjiao Yang. Multilevel Diffusion: Infinite Dimensional Score-Based Diffusion Models for Image Generation. *arXiv preprint arXiv:2303.04772*, 2023.

[125] Tian Han, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. Alternating Back-Propagation for Generator Network. *AAAI*, 31(1), 2017. ISSN 2374-3468.

[126] Leonard Hasenclever, Jakub M Tomczak, and Max Welling. Variational Inference with Orthogonal Normalizing Flows. In *Workshop on Bayesian Deep Learning, NIPS*, 2017.

[127] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE CVPR*, 2016.

[128] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *Advances in neural information processing systems*, 30, 2017.

[129] Dave Higdon. Space and Space-Time Modeling using Process Convolutions. In *Quantitative methods for current environmental issues*, pages 37–56. Springer, 2002.

[130] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-Vae: Learning Basic Visual Concepts with a Constrained Variational Framework. In *ICLR*, 2017.

[131] G E Hinton and T J Sejnowski. Optimal Perceptual Inference. In *IEEE CVPR*, 1983.

[132] Geoffrey E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 2002. ISSN 0899-7667, 1530-888X. doi: 10.1162/089976602760128018.

[133] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Comput.*, 18(7):1527–1554, 2006. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.2006.18.7.1527.

[134] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving Flow-Based Generative Models with Variational Dequantization and Architecture Design. In *ICML*, 2019.

[135] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[136] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded Diffusion Models for High Fidelity Image Generation. *J. Mach. Learn. Res.*, 23(47):1–33, 2022.

[137] Jonathan Ho, Tim Salimans, Alexey A Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video Diffusion Models. *Advances in Neural Information Processing Systems*, 2022.

[138] Sepp Hochreiter and Jrgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

[139] Matthew Hoffman, Pavel Sountsov, Joshua V. Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. NeuTra-lizing Bad Geometry in Hamiltonian Monte Carlo Using Neural Transport. *arXiv:1903.03704*, 2019.

[140] Matthew D Hoffman and Matthew J Johnson. ELBO surgery: yet another way to carve up the variational evidence lower bound. In *Advances in Neural Information Processing Systems Workshop*, 2016.

[141] Emiel Hoogeboom and Tim Salimans. Blurring Diffusion Models. In *International Conference on Learning Representations*, 2023.

[142] Emiel Hoogeboom, Rianne van den Berg, and Max Welling. Emerging Convolutions for Generative Normalizing Flows. In *ICML*, 2019.

[143] Emiel Hoogeboom, Jorn Peters, Rianne van den Berg, and Max Welling. Integer Discrete Flows and Lossless Compression. *Advances in Neural Information Processing Systems*, 32, 2019.

[144] Emiel Hoogeboom, Victor Garcia Satorras, Jakub Tomczak, and Max Welling. The Convolution Exponential and Generalized Sylvester Flows. *Advances in Neural Information Processing Systems*, 33, 2020.

[145] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax Flows and Multinomial Diffusion: Towards Non-Autoregressive Language Models. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021.

[146] Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive Diffusion Models. In *International Conference on Learning Representations*, 2022.

[147] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant Diffusion for Molecule Generation in 3D. In *International Conference on Machine Learning*, pages 8867–8887. PMLR, 2022.

[148] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. Simple Diffusion: End-to-End Diffusion for High Resolution Images. In *International Conference on Machine Learning*, pages 13213–13232. PMLR, 2023.

[149] Chin-Wei Huang, Ahmed Touati, Laurent Dinh, Michal Drozdzal, Mohammad Havaei, Laurent Charlin, and Aaron Courville. Learnable Explicit Density for Continuous Latent Space and Variational Inference. *arXiv:1710.02248*, 2017.

[150] Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. Neural Autoregressive Flows. In *ICML*, 2018.

[151] Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented Normalizing Flows: Bridging the Gap Between Generative Flows and Latent Variable Models. *arXiv:2002.07101*, 2020.

[152] Huaibo Huang, zhihang li, Ran He, Zhenan Sun, and Tieniu Tan. Introvae: Introspective Variational Autoencoders for Photographic Image Synthesis. *Advances in Neural Information Processing Systems*, 31, 2018.

[153] Nisha Huang, Fan Tang, Weiming Dong, and Changsheng Xu. Draw Your Art Dream: Diverse Digital Art Synthesis with Multimodal Guided Diffusion. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1085–1094, 2022.

[154] Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. Stacked Generative Adversarial Networks. In *IEEE CVPR*, 2017.

[155] Robert A Hummel, B Kimia, and Steven W Zucker. Deblurring Gaussian Blur. *Computer Vision, Graphics, and Image Processing*, 38(1):66–80, 1987.

[156] Ferenc Huszr. How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary? *arXiv:1511.05101*, 2015.

[157] Stephanie L Hyland, Shruthi Bannur, Kenza Bouzid, Daniel C Castro, Mercy Ranjit, Anton Schwaighofer, Fernando Pérez-García, Valentina Salvatelli, Shaury Srivastav, Anja Thieme, et al. Maira-1: A specialised large multimodal model for radiology report generation. *arXiv preprint arXiv:2311.13668*, 2023.

[158] Aapo Hyvärinen. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6(4), 2005.

[159] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, page 448456, 2015.

[160] Allan Jabri, David Fleet, and Ting Chen. Scalable Adaptive Computation for Iterative Generation. In *International Conference on Machine Learning*, pages 14569–14589, 2023.

[161] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver IO: A General Architecture for Structured Inputs & Outputs. In *International Conference on Learning Representations*, 2022.

[162] Priyank Jaini, Kira A. Selby, and Yaoliang Yu. Sum-of-Squares Polynomial Flow. In *ICML*, 2019.

[163] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017.

[164] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. TransGAN: Two Transformers Can Make One Strong GAN. *arXiv:2102.07074*, 2021.

[165] Yuming Jiang, Shuai Yang, Haonan Qiu, Wayne Wu, Chen Change Loy, and Ziwei Liu. Text2Human: Text-Driven Controllable Human Image Generation. *ACM Transactions on Graphics (TOG)*, 41(4):1–11, 2022.

[166] Bowen Jing, Gabriele Corso, Renato Berlinghieri, and Tommi Jaakkola. Subspace Diffusion Generative Models. In *European Conference on Computer Vision*, pages 274–289. Springer, 2022.

[167] Fritz John. Numerical Solution of the Equation of Heat Conduction for Preceding Times. *Annali di Matematica pura ed Applicata*, 40:129–142, 1955.

[168] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. *ICLR*, 2018.

[169] Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta Go Fast When Generating Data with Score-Based Models. *arXiv preprint arXiv:2105.14080*, 2021.

[170] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999. ISSN 08856125. doi: 10.1023/A:1007665907178.

[171] Heewoo Jun, Rewon Child, Mark Chen, John Schulman, Aditya Ramesh, Alec Radford, and Ilya Sutskever. Distribution Augmentation for Generative Modeling. In *ICML*, 2020.

[172] Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. Fast Decoding in Sequence Models using Discrete Latent Variables. In *ICML*, 2018.

[173] Konstantinos Kamnitsas, Daniel C Castro, Loic Le Folgoc, Ian Walker, Ryutaro Tanno, Daniel Rueckert, Ben Glocker, Antonio Criminisi, and Aditya Nori. Semi-Supervised Learning via Compact Latent Space Clustering. *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[174] Mahdi Karami, Dale Schuurmans, Jascha Sohl-Dickstein, Laurent Dinh, and Daniel Duckworth. Invertible Convolutional Flow. *Advances in Neural Information Processing Systems*, 2019.

[175] Animesh Karnewar and Oliver Wang. MSG-GAN: Multi-Scale Gradients for Generative Adversarial Networks. In *IEEE CVPR*, 2020.

[176] Animesh Karnewar, Andrea Vedaldi, Niloy J Mitra, and David Novotny. Goenfusion: Gradient origin encodings for 3d forward diffusion models. *arXiv preprint arXiv:2312.08744*, 2023.

[177] Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy J Mitra. HoloDiffusion: Training a 3D Diffusion Model using 2D Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18423–18433, 2023.

[178] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*, 2018.

[179] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[180] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training Generative Adversarial Networks with Limited Data. *Advances in Neural Information Processing Systems*, 33, 2020.

[181] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and Improving the Image Quality of StyleGAN. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[182] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-Free Generative Adversarial Networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.

[183] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

[184] Gavin Kerrigan, Justin Ley, and Padhraic Smyth. Diffusion Generative Models in Infinite Dimensions. In *International Conference on Artificial Intelligence and Statistics*, pages 9538–9563. PMLR, 2023.

[185] Firas Khader, Gustav Mueller-Franzes, Soroosh Tayebi Arasteh, Tianyu Han, Christoph Haarburger, Maximilian Schulze-Hagen, Philipp Schad, Sandy Engelhardt, Bettina Baessler, Sebastian Foersch, et al. Medical Diffusion–Denoising Diffusion Probabilistic Models for 3D Medical Image Generation. *Nature Scientific Reports*, 13(1):7303, 2023.

[186] Hyunjik Kim, George Papamakarios, and Andriy Mnih. The Lipschitz Constant of Self-Attention. *arXiv:2006.04710*, 2020.

[187] Taesup Kim and Yoshua Bengio. Deep Directed Generative Models with Energy-Based Probability Estimation. *arXiv:1606.03439*, 2016.

[188] Yoon Kim, Sam Wiseman, Andrew Miller, David Sontag, and Alexander Rush. Semi-Amortized Variational Autoencoders. In *ICML*, 2018.

[189] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015.

[190] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

[191] Durk P Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[192] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved Variational Inference with Inverse Autoregressive Flow. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[193] I. Kobyzev, S. Prince, and M. Brubaker. Normalizing Flows: An Introduction and Review of Current Methods. *IEEE TPAMI*, pages 2008–2026, 2020.

[194] Jean Kossaifi, Nikola Kovachki, Kamyar Azizzadenesheli, and Anima Anandkumar. Multi-Grid Tensorized Fourier Neural Operator for High-Resolution PDEs. *arXiv preprint arXiv:2310.00120*, 2023.

[195] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural Operator: Learning Maps Between Function Spaces with Applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[196] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009.

[197] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[198] Alexander Kukush. *Gaussian Measures in Hilbert Space: Construction and Properties*. John Wiley & Sons, 2020.

[199] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron C. Courville. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis. *Advances in Neural Information Processing Systems*, 32, 2019.

[200] Manoj Kumar, Mohammad Babaeizadeh, Dumitru Erhan, Chelsea Finn, Sergey Levine, Laurent Dinh, and Durk Kingma. VideoFlow: A Flow-Based Generative Model for Video. In *ICML Workshop*, 2019.

[201] Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum Entropy Generators for Energy-Based Models. *arXiv:1901.08508*, 2019.

[202] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved Precision and Recall Metric for Assessing Generative Models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, 2019.

[203] Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The Role of ImageNet Classes in Fréchet Inception Distance. In *International Conference on Learning Representations*, 2023.

[204] Hugo Larochelle and Iain Murray. The Neural Autoregressive Distribution Estimator. In *AISTATS*, 2011.

[205] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

[206] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[207] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, volume 2, pages II–104, 2004.

[208] Yann LeCun, Sumit Chopra, Raia Hadsell, MarcAurelio Ranzato, and Fu Jie Huang. A Tutorial on Energy-Based Learning. *A Tutorial on Energy-Based Learning*, 1(0), 2006.

[209] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. *Nature*, 521(7553):436–444, 2015.

[210] Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic Adversarial Video Prediction. *arXiv:1804.01523*, 2018.

[211] Brett Levac, Ajil Jalal, Kannan Ramchandran, and Jonathan I Tamir. Mri reconstruction with side information using diffusion models. In *2023 57th Asilomar Conference on Signals, Systems, and Computers*, pages 1436–1442, 2023.

[212] Gen Li, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. Breaking the Sample Size Barrier in Model-Based Reinforcement Learning with a Generative Model. *Advances in neural information processing systems*, 33:12861–12872, 2020.

[213] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural Operator: Graph Kernel Network for Partial Differential Equations. *arXiv preprint arXiv:2003.03485*, 2020.

[214] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar, et al. Fourier Neural Operator for Parametric Partial Differential Equations. In *International Conference on Learning Representations*, 2021.

[215] Jae Hyun Lim and Jong Chul Ye. Geometric GAN. *arXiv:1705.02894*, 2017.

[216] Jae Hyun Lim, Nikola B Kovachki, Ricardo Baptista, Christopher Beckham, Kamyar Azizzadenesheli, Jean Kossaifi, Vikram Voleti, Jiaming Song, Karsten Kreis, Jan Kautz, et al. Score-Based Diffusion Models in Function Space. *arXiv preprint arXiv:2302.07400*, 2023.

[217] Haitao Lin, Yufei Huang, Meng Liu, Xuanjing Li, Shuiwang Ji, and Stan Z Li. DiffBP: Generative Diffusion of 3D Molecules for Target Protein Binding. *arXiv preprint arXiv:2211.11214*, 2022.

[218] Ji Lin, Richard Zhang, Frieder Ganz, Song Han, and Jun-Yan Zhu. Anycost GANs for Interactive Image Synthesis and Editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14986–14996, 2021.

[219] Zachary C Lipton and Subarna Tripathi. Precise Recovery of Latent Vectors from Generative Adversarial Networks. In *5th International Conference on Learning Representations, ICLR*, 2017.

[220] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards Faster and Stabilized GAN Training for High-fidelity Few-shot Image Synthesis. In *International Conference on Learning Representations*, 2021.

[221] Kanglin Liu, Guoping Qiu, Wenming Tang, and Fei Zhou. Spectral Regularization for Combating Mode Collapse in GANs. In *IEEE ICCV*, 2019. ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00648.

[222] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised Image-to-Image Translation Networks. In *Advances in Neural Information Processing Systems*, 2017.

[223] Xuanqing Liu and Cho-Jui Hsieh. Rob-GAN: Generator, Discriminator, and Adversarial Attacker. In *IEEE CVPR*, 2019.

[224] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

[225] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[226] Gabriel Loaiza-Ganem and John P. Cunningham. The continuous Bernoulli: fixing a pervasive error in variational autoencoders. *Advances in Neural Information Processing Systems*, 32, 2019.

[227] Cheng Lu, Jianfei Chen, Chongxuan Li, Qiuhao Wang, and Jun Zhu. Implicit Normalizing Flows. In *ICLR*, 2021.

[228] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. In *Advances in Neural Information Processing Systems*, 2022.

[229] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are GANs Created Equal? A Large-Scale Study. *Advances in Neural Information Processing Systems*, 31, 2018.

[230] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using Denoising Diffusion Probabilistic Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

[231] Xuezhe Ma, Xiang Kong, Shanghang Zhang, and Eduard Hovy. MaCow: Masked Convolutional Generative Flow. *Advances in Neural Information Processing Systems*, 2019.

[232] Xuezhe Ma, Chunting Zhou, and Eduard Hovy. MAE: Mutual Posterior-Divergence Regularization for Variational AutoEncoders. In *ICLR*, 2019.

[233] Lars Maale, Marco Fraccaro, Valentin Liévin, and Ole Winther. BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. *Advances in Neural Information Processing Systems*, 33, 2019.

[234] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations*, 2017.

[235] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv:1706.06083*, 2019.

[236] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial Autoencoders. *arXiv:1511.05644*, 2016.

[237] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least Squares Generative Adversarial Networks. In *IEEE CVPR*, 2017.

[238] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. SampleRNN: An Unconditional End-to-End Neural Audio Generation Model. In *ICLR*, 2017.

[239] Chenlin Meng, Lantao Yu, Yang Song, Jiaming Song, and Stefano Ermon. Autoregressive Score Matching. *Advances in Neural Information Processing Systems*, 34, 2020.

[240] Chenlin Meng, Jiaming Song, Yang Song, Shengjia Zhao, and Stefano Ermon. Improved Autoregressive Modeling with Distribution Smoothing. In *ICLR*, 2021.

[241] Jacob Menick and Nal Kalchbrenner. Generating High Fidelity Images with Subscale Pixel Networks and Multidimensional Upscaling. In *International Conference on Learning Representations*, 2019.

[242] Chenfeng Miao, Shuang Liang, Minchuan Chen, Jun Ma, Shaojun Wang, and Jing Xiao. Flow-TTS: A Non-Autoregressive Network for Text to Speech Based on Flow. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7209–7213. IEEE, 2020.

[243] Beren Millidge, Tommaso Salvatori, Yuhang Song, Rafal Bogacz, and Thomas Lukasiewicz. Predictive Coding: Towards a Future of Deep Learning beyond Backpropagation? *arXiv preprint arXiv:2202.09467*, 2022.

[244] Hà Quang Minh. Regularized Divergences Between Covariance Operators and Gaussian Measures on Hilbert Spaces. *Journal of Theoretical Probability*, 34: 580–643, 2021.

[245] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv:1411.1784*, 2014.

[246] Yasuhide Miura, Yuhao Zhang, Emily Bao Tsai, Curtis P Langlotz, and Dan Jurafsky. Improving Factual Completeness and Consistency of Image-to-Text Radiology Report Generation. *arXiv preprint arXiv:2010.10042*, 2020.

[247] Koichi Miyasawa. An Empirical Bayes Estimator of the Mean of a Normal Population. *Bulletin of the International Statistical Institute*, 38(4):181–188, 1961.

[248] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *International Conference on Learning Representations*, 2018.

[249] Thomas Mller, Brian Mcwilliams, Fabrice Rousselle, Markus Gross, and Jan Novk. Neural Importance Sampling. *ACM TOG*, 38(5):145:1–145:19, 2019. ISSN 0730-0301. doi: 10.1145/3341156.

[250] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable Fidelity and Diversity Metrics for Generative Models. In *International Conference on Machine Learning*, pages 7176–7185, 2020.

[251] SA Nane, SK Nayar, and H Murase. Columbia Object Image Library: COIL-20. *Technical Report CUCS-005-96, Columbia University*, 1996.

[252] Charlie Nash and Conor Durkan. Autoregressive Energy Machines. In *ICML*, 2019.

[253] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W Battaglia. Generating Images with Sparse Representations. In *International Conference on Machine Learning*, pages 7958–7968. PMLR, 2021.

[254] Kirill Neklyudov, Evgenii Egorov, and Dmitry P. Vetrov. The Implicit Metropolis-Hastings Algorithm. *Advances in Neural Information Processing Systems*, 32, 2019.

[255] M. Ngxande, J. Tapamo, and M. Burke. DepthwiseGANs: Fast Training Generative Adversarial Networks for Realistic Image Synthesis. In *SAUPEC/RobMech/PRASA*, pages 111–116, 2019. doi: 10.1109/RoboMech.2019.8704766.

[256] Alexander Quinn Nichol and Prafulla Dhariwal. Improved Denoising Diffusion Probabilistic Models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[257] Weili Nie, Nina Narodytska, and Ankit Patel. RelGAN: Relational Generative Adversarial Networks for Text Generation. In *International Conference on Learning Representations*, 2019.

[258] Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. SurVAE Flows: Surjections to Bridge the Gap between VAEs and Flows. *Advances in Neural Information Processing Systems*, 33, 2020.

[259] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning Non-Convergent Non-Persistent Short-Run MCMC Toward Energy-Based Model. In *Neural Information Processing Systems (Advances in Neural Information Processing Systems)*, pages 5232–5242, 2019.

[260] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the Anatomy of MCMC-Based Maximum Likelihood Learning of Energy-Based Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5272–5280, 2020.

[261] Erik Nijkamp, Bo Pang, Tian Han, Linqi Zhou, Song-Chun Zhu, and Ying Nian Wu. Learning Multi-layer Latent Variable Model via Variational Optimization of Short Run MCMC for Approximate Inference. In *European Conference on Computer Vision*, pages 361–378. Springer, 2020.

[262] Erik Nijkamp, Ruiqi Gao, Pavel Sountsov, Srinivas Vasudevan, Bo Pang, Song-Chun Zhu, and Ying Nian Wu. Mcmc should mix: Learning energy-based model with neural transport latent space mcmc. In *International Conference on Learning Representations*, 2022.

[263] Sajad Norouzi, David J. Fleet, and Mohammad Norouzi. Exemplar VAE: Linking Generative Models, Nearest Neighbor Retrieval, and Data Augmentation. *Advances in Neural Information Processing Systems*, 33, 2020.

[264] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. *Advances in Neural Information Processing Systems*, 29, 2016.

[265] Anton Obukhov, Maximilian Seitzer, Po-Wei Wu, Semen Zhydenko, Jonathan Kyl, and Elvis Yu-Jing Lin. High-fidelity performance metrics for generative models in pytorch, 2020. URL `https://github.com/toshas/torch-fidelity`. Version: 0.3.0, DOI: 10.5281/zenodo.4957738.

[266] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional Image Synthesis with Auxiliary Classifier GANs. In *ICML*, 2017.

[267] Derek Onken, Samy Wu Fung, Xingjian Li, and Lars Ruthotto. OT-Flow: Fast and Accurate Continuous Normalizing Flows via Optimal Transport. In *AAAI*, 2021.

[268] Georg Ostrovski, Will Dabney, and Rémi Munos. Autoregressive Quantile Networks for Generative Modeling. In *International Conference on Machine Learning*, 2018.

[269] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning Latent Space Energy-Based Prior Model. In *Advances in Neural Information Processing Systems*, volume 34, 2020.

[270] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation. *Advances in Neural Information Processing Systems*, 30, 2017.

[271] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing Flows for Probabilistic Modeling and Inference. *JMLR*, 22(57):1–64, 2021.

[272] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *IEEE CVPR*, 2019.

[273] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image Transformer. In *ICML*, 2018.

[274] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035, 2019.

[275] Fernando Pérez-García, Sam Bond-Taylor, Pedro P Sanchez, Boris van Breugel, Daniel C Castro, Harshita Sharma, Valentina Salvatelli, Maria TA Wetscherek, Hannah Richardson, Matthew P Lungren, et al. Radedit: stress-testing biomedical vision models via diffusion image editing. In *European Conference of Computer Vision*, 2024.

[276] Jakiw Pidstrigach, Youssef Marzouk, Sebastian Reich, and Sven Wang. Infinite-Dimensional Diffusion Models for Function Spaces. *arXiv preprint arXiv:2302.10130*, 2023.

[277] Francis J Pinski, Gideon Simpson, Andrew M Stuart, and Hendrik Weber. Kullback–Leibler Approximation for Probability Measures on Infinite Dimensional Spaces. *SIAM Journal on Mathematical Analysis*, 47(6):4091–4122, 2015.

[278] L. S. Pontryagin. *Mathematical Theory of Optimal Processes*. Routledge, 2018. ISBN 978-1-351-43306-8.

[279] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion Autoencoders: Toward a Meaningful and Decodable Representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10619–10629, 2022.

[280] R. Prenger, R. Valle, and B. Catanzaro. WaveGlow: A Flow-based Generative Network for Speech Synthesis. In *IEEE ICASSP*, pages 3617–3621, 2019.

[281] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *Advances in neural information processing systems*, 30, 2017.

[282] Shilin Qiu, Qihe Liu, Shijie Zhou, and Chunjiang Wu. Review of Artificial Intelligence Adversarial Attack and Defense Technologies. *Applied Sciences*, 9 (5):909, 2019.

[283] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In *ICLR*, 2016.

[284] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. 2019.

[285] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[286] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[287] Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-NO: U-Shaped Neural Operators. *Transactions on Machine Learning Research*, 2023.

[288] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-Shot Text-to-Image Generation. *arXiv preprint arXiv:2102.12092*, 2021.

[289] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:2204.06125*, 2022.

[290] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. Skilful Precipitation Nowcasting using Deep Generative Models of Radar. *Nature*, 597(7878):672–677, 2021.

[291] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. *Advances in Neural Information Processing Systems*, 32, 2019.

[292] Scott Reed, Aäron Oord, Nal Kalchbrenner, Sergio Gmez Colmenarejo, Ziyu Wang, Yutian Chen, Dan Belov, and Nando Freitas. Parallel Multiscale Autoregressive Density Estimation. In *International Conference on Machine Learning*, 2017.

[293] Danilo Rezende and Shakir Mohamed. Variational Inference with Normalizing Flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015.

[294] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conference on Machine Learning*, 2014.

[295] Severi Rissanen, Markus Heinonen, and Arno Solin. Generative Modelling with Inverse Heat Dissipation. In *International Conference on Learning Representations*, 2023.

[296] Herbert Robbins. An Empirical Bayes Approach to Statistics. In *Proc. Third Berkely Symp.*, volume 1, pages 157–163, 1956.

[297] Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4): 341–363, 1996. ISSN 1350-7265.

[298] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[299] David W Romero, Robert-Jan Bruintjes, Jakub Mikolaj Tomczak, Erik J Bekkers, Mark Hoogendoorn, and Jan van Gemert. Flexconv: Continuous kernel convolutions with differentiable kernel sizes. In *International Conference on Learning Representations*, 2022.

[300] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[301] Lucas SP Rudden, Mahdi Hijazi, and Patrick Barth. Deep Learning Approaches for Conformational Flexibility and Switching Properties in Protein Design. *Frontiers in Molecular Biosciences*, page 840, 2022.

[302] Laura Ruis, Mitchell Stern, Julia Proskurnia, and William Chan. Insertion-Deletion Transformer. *arXiv preprint arXiv:2001.05540*, 2020.

[303] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3): 211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[304] Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. Non-Autoregressive Machine Translation with Latent Alignments. *arXiv preprint arXiv:2004.07437*, 2020.

[305] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image Super-Resolution via Iterative Refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[306] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing Generative Models via Precision and Recall. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[307] Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations*, 2022.

[308] Tim Salimans, Diederik Kingma, and Max Welling. Markov Chain Monte Carlo and Variational Inference: Bridging the Gap. In *ICML*, 2015.

[309] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. *Advances in Neural Information Processing Systems*, 2016.

[310] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications. *ICLR*, 2017.

[311] Saeed Saremi and Aapo Hyvarinen. Neural Empirical Bayes. *Journal of Machine Learning Research*, 20:1–23, 2019.

[312] Saeed Saremi, Arash Mehrjou, Bernhard Schlkopf, and Aapo Hyvärinen. Deep Energy Estimator Networks. *arXiv:1805.08306*, 2018.

[313] Hiroshi Sasaki, Chris G Willcocks, and Toby P Breckon. UNIT-DDPM: UNpaired Image Translation with Denoising Diffusion Probabilistic Models. *arXiv preprint arXiv:2104.05358*, 2021.

[314] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected GANs Converge Faster. *Advances in Neural Information Processing Systems*, 34:17480–17492, 2021.

[315] Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled Denoising Autoencoders for Text Generation. In *International Conference on Learning Representations*, 2022.

[316] Jürgen Schmidhuber. Making the World Differentiable: On Using Self-Supervised Fully Recurrent Neural Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments. 1990.

[317] Jürgen Schmidhuber. Generative Adversarial Networks are special cases of Artificial Curiosity (1990) and also closely related to Predictability Minimization (1991). *Neural Netw.*, 127:58–66, 2020.

[318] Jiaxin Shi, Shengyang Sun, and Jun Zhu. A Spectral Approach to Gradient Estimation for Implicit Distributions. In *ICML*, 2018.

[319] Rajhans Singh, Ankita Shukla, and Pavan Turaga. Polynomial Implicit Neural Representations For Large Diverse Datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2041–2051, 2023.

[320] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-GAN: Speeding up GAN Training using Core-Sets. In *ICML*, 2020.

[321] Vincent Sitzmann, Eric R. Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. MetaSDF: Meta-learning Signed Distance Functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020.

[322] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit Neural Representations with Periodic Activation Functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

[323] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[324] Jiaming Song, Shengjia Zhao, and Stefano Ermon. A-NICE-MC: Adversarial Training for MCMC. *Advances in Neural Information Processing Systems*, 30, 2017.

[325] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*, 2021.

[326] Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[327] Yang Song, Chenlin Meng, and Stefano Ermon. MintNet: Building Invertible Neural Networks with Masked Convolutions. *Advances in Neural Information Processing Systems*, 32, 2019.

[328] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced Score Matching: A Scalable Approach to Density and Score Estimation. In *UAI*, pages 574–584, 2020.

[329] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021.

[330] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR, 2023.

[331] Yuxuan Song, Qiwei Ye, Minkai Xu, and Tie-Yan Liu. Discriminator Contrastive Divergence: Semi-Amortized Generative Modeling by Exploring Energy of the Discriminator. *arXiv:2004.01704*, 2020.

[332] Ilya Sutskever and Tijmen Tieleman. On the Convergence Properties of Contrastive Divergence. In *AISTATS*, pages 789–795, 2010.

[333] Casper Kaae Snderby, Tapani Raiko, Lars Maale, Sren Kaae Snderby, and Ole Winther. Ladder Variational Autoencoders. *Advances in Neural Information Processing Systems*, 29, 2016.

[334] Casper Kaae Snderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszr. Amortised MAP Inference for Image Super-resolution. In *ICLR*, 2017.

[335] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

[336] Haotian Tang, Zhijian Liu, Xiuyu Li, Yujun Lin, and Song Han. TorchSparse: Efficient Point Cloud Inference Engine. In *Conference on Machine Learning and Systems (MLSys)*, 2022.

[337] H. Thanh-Tung and T. Tran. Catastrophic forgetting and mode collapse in GANs. In *IJCNN*, pages 1–10, 2020. doi: 10.1109/IJCNN48605.2020.9207181.

[338] Lucas Theis and Matthias Bethge. Generative Image Modeling Using Spatial LSTMs. In *Advances in Neural Information Processing Systems*, 2015.

[339] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv:1511.01844*, 2016.

[340] Tijmen Tieleman. Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient. In *ICML*, 2008.

[341] Michalis Titsias and Petros Dellaportas. Gradient-based Adaptive Markov Chain Monte Carlo. *Advances in Neural Information Processing Systems*, 32, 2019.

[342] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein Auto-Encoders. *arXiv:1711.01558*, 2019.

[343] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. MLP-Mixer: An All-MLP Architecture for Vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021.

[344] Jakub Tomczak and Max Welling. VAE with a VampPrior. In *AISTATS*, pages 1214–1223, 2018.

[345] Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian Fatras, Guy Wolf, and Yoshua Bengio. Conditional Flow Matching: Simulation-Free Dynamic Optimal Transport. *arXiv preprint arXiv:2302.00482*, 2023.

[346] Dustin Tran, Rajesh Ranganath, and David M. Blei. Variational Gaussian Process. In *ICLR*, 2016.

[347] Dustin Tran, Keyon Vafa, Kumar Agrawal, Laurent Dinh, and Ben Poole. Discrete Flows: Invertible Generative Models of Discrete Data. *Advances in Neural Information Processing Systems*, 32, 2019.

[348] N.-T. Tran, V.-H. Tran, N.-B. Nguyen, T.-K. Nguyen, and N.-M. Cheung. On Data Augmentation for GAN Training. *IEEE TIP*, 30:1882–1897, 2021. ISSN 1941-0042. doi: 10.1109/TIP.2021.3049346.

[349] Ngoc-Trung Tran, Viet-Hung Tran, Bao-Ngoc Nguyen, Linxiao Yang, and Ngai-Man (Man) Cheung. Self-supervised GAN: Analysis and Improvement with Multi-class Minimax Game. *Advances in Neural Information Processing Systems*, 32, 2019.

[350] Anton Tsitsulin, Marina Munkhoeva, Davide Mottin, Panagiotis Karras, Alex Bronstein, Ivan Oseledets, and Emmanuel Müller. The Shape of Data: Intrinsic Distance for Data Distributions. In *International Conference on Learning Representations*, 2020.

[351] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2023.

[352] Richard Eric Turner and Maneesh Sahani. Two problems with variational expectation maximisation for time series models. In *Bayesian Time Series Models*, pages 104–124, Cambridge, 2011. ISBN 978-0-511-98467-9. doi: 10.1017/CBO9780511984679.006.

[353] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. Metropolis-Hastings Generative Adversarial Networks. In *ICML*, 2019.

[354] Belinda Tzen and Maxim Raginsky. Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *COLT*, pages 3084–3114, 2019.

[355] Benigno Uria, Iain Murray, and Hugo Larochelle. RNADE: The Real-Valued Neural Autoregressive Density-Estimator. In *Advances in Neural Information Processing Systems*, 2013.

[356] Arash Vahdat and Jan Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[357] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-Based Generative Modeling in Latent Space. *Advances in Neural Information Processing Systems*, 34, 2021.

[358] Gabriele Valvano, Andrea Leo, and Sotirios A Tsaftaris. Learning to Segment from Scribbles using Multi-scale Adversarial Attention Gates. *IEEE T-MI*, 2021.

[359] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. *arXiv:1609.03499*, 2016.

[360] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, koray kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional Image Generation with PixelCNN Decoders. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[361] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *ICML*, 2016. ISBN 978-1-5108-2900-8.

[362] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural Discrete Representation Learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[363] Aaron van den Oord et al. Parallel WaveNet: Fast High-Fidelity Speech Synthesis. In *ICML*, 2018.

[364] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. *Advances in neural information processing systems*, 30, 2017.

[365] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete Denoising Diffusion for Graph Generation. In *The Eleventh International Conference on Learning Representations*, 2023.

[366] Cédric Villani. *Topics in Optimal Transportation*. Number 58. American Mathematical Soc., 2003. ISBN 978-0-8218-3312-4.

[367] Cédric Villani. *Optimal Transport: Old and New*, volume 338. Springer Science & Business Media, 2008.

[368] Pascal Vincent. A Connection Between Score Matching and Denoising Autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[369] Alex Wang and Kyunghyun Cho. BERT has a Mouth, and it Must Speak: BERT as a Markov Random Field Language Model. In *NeuralGen*, 2019.

[370] Clinton Wang and Polina Golland. Discretization Invariant Learning on Neural Fields. In *arXiv preprint arXiv:2206.01178*, 2022.

[371] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-Attention with Linear Complexity. *arXiv:2006.04768*, 2020.

[372] Yuanhao Wang, Guodong Zhang, and Jimmy Ba. On Solving Minimax Optimization Locally: A Follow-the-Ridge Approach. In *ICLR*, 2020.

[373] Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to Efficiently Sample from Diffusion Probabilistic Models. *arXiv preprint arXiv:2106.03802*, 2021.

[374] Antoine Wehenkel and Gilles Louppe. Unconstrained Monotonic Neural Networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[375] Max Welling and Yee Whye Teh. Bayesian Learning via Stochastic Gradient Langevin Dynamics. In *ICML*, 2011.

[376] Christopher Williams, Fabian Falck, George Deligiannidis, Chris Holmes, Arnaud Doucet, and Saifuddin Syed. A Unified Framework for U-Net Design and Analysis. *Advances in Neural Information Processing Systems*, 36:27745–27782, 2023.

[377] Hao Wu, Jonas Khler, and Frank Noé. Stochastic Normalizing Flows. *Advances in Neural Information Processing Systems*, 2020.

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

[379] Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. VAEBM: A Symbiosis between Variational Autoencoders and Energy-based Models. In *International Conference on Learning Representations*, 2021.

[380] J. Xie, Y. Lu, R. Gao, S. Zhu, and Y. N. Wu. Cooperative Training of Descriptor and Generator Networks. *IEEE TPAMI*, 42(1):27–45, 2020. ISSN 1939-3539. doi: 10.1109/TPAMI.2018.2879081.

[381] Jianwen Xie, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. A Theory of Generative ConvNet. In *International Conference on Machine Learning*, 2016.

[382] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural Fields in Visual Computing and Beyond. In *Computer Graphics Forum*, volume 41, pages 641–676. Wiley Online Library, 2022.

[383] Yilun Xu, Ziming Liu, Max Tegmark, and Tommi S Jaakkola. Poisson flow generative models. In *Advances in Neural Information Processing Systems*, volume 36, 2022.

[384] L. Yang and G. E. Karniadakis. Potential Flow Generator With L2 Optimal Transport Regularity for Generative Models. *IEEE TNNLS*, pages 1–11, 2020. ISSN 2162-2388. doi: 10.1109/TNNLS.2020.3028042.

[385] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365*, 2015.

[386] Lantao Yu, Yang Song, Jiaming Song, and Stefano Ermon. Training Deep Energy-Based Models with f-Divergence Minimization. In *ICML*, 2020.

[387] Amir Zadeh, Yao-Chong Lim, Paul Pu Liang, and Louis-Philippe Morency. Variational Auto-Decoder. *arXiv preprint arXiv:1903.00840*, 2019.

[388] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. *arXiv:1605.07146*, 2017.

[389] Bowen Zhang, Shuyang Gu, Bo Zhang, Jianmin Bao, Dong Chen, Fang Wen, Yong Wang, and Baining Guo. StyleSwin: Transformer-Based GAN for High-Resolution Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11304–11314, 2022.

[390] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. StackGAN: Text to Photo-Realistic Image Synthesis With Stacked Generative Adversarial Networks. In *IEEE CVPR*, 2017.

[391] Han Zhang, Xi Gao, Jacob Unterman, and Tom Arodz. Approximation Capabilities of Neural Ordinary Dierential Equations. *arXiv:1907.12998*, 2019.

[392] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-Attention Generative Adversarial Networks. *arXiv:1805.08318*, 2019.

[393] Han Zhang, Ruili Feng, Zhantao Yang, Lianghua Huang, Yu Liu, Yifei Zhang, Yujun Shen, Deli Zhao, Jingren Zhou, and Fan Cheng. Dimensionality-Varying Diffusion Process. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14307–14316, 2023.

[394] Junyi Zhang, Charles Herrmann, Junhwa Hur, Luisa Polania Cabrera, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. A Tale of Two Features: Stable Diffusion Complements DINO for Zero-Shot Semantic Correspondence. *Advances in Neural Information Processing Systems*, 36, 2024.

[395] Linfeng Zhang, Weinan E, and Lei Wang. Monge-Ampère Flow for Generative Modeling. *arXiv:1809.10188*, 2018.

[396] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[397] Zijian Zhang, Zhou Zhao, and Zhijie Lin. Unsupervised Representation Learning from Pre-trained Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, 2022.

[398] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based Generative Adversarial Network. In *ICLR*, 2017.

[399] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Learning Hierarchical Features from Deep Generative Models. In *ICML*, 2017.

[400] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Towards Deeper Understanding of Variational Autoencoding Models. *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[401] Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Balancing Learning and Inference in Variational Autoencoders. *AAAI*, 33(01):5885–5892, 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33015885.

[402] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable Augmentation for Data-Efficient GAN Training. 33, 2020.

[403] Zhengli Zhao, Zizhao Zhang, Ting Chen, Sameer Singh, and Han Zhang. Image Augmentations for GAN Training. *arXiv:2006.02595*, 2020.

[404] Jiachen Zhong, Xuanqing Liu, and Cho-Jui Hsieh. Improving the Speed and Quality of GAN by Adversarial Training. *arXiv:2008.03364*, 2020.

[405] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with Local and Global Consistency. *Advances in neural information processing systems*, 16:321–328, 2003.

[406] Jia-Jie Zhu and José Bento. Generative Adversarial Active Learning. *arXiv preprint arXiv:1702.07956*, 2017.

[407] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative Visual Manipulation on the Natural Image Manifold. In *European Conference on Computer Vision*, pages 597–613, 2016.

[408] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[409] Peiye Zhuang, Samira Abnar, Jiatao Gu, Alex Schwing, Joshua M Susskind, and Miguel Ángel Bautista. Diffusion Probabilistic Fields. In *International Conference on Learning Representations*, 2023.

[410] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

[411] Zachary Ziegler and Alexander Rush. Latent Normalizing Flows for Discrete Sequences. In *ICML*, 2019.