Reining in the Deep Generative Models

Partha Ghosh

MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS

Reining in the Deep Generative Models

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Partha Ghosh
aus Joygoria, Indien

Tübingen
2022

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	24.11.2022
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Michael J. Black
2. Berichterstatter:	Prof. Dr. Andrew Blake

To my parents and friends who have helped shape my personality

Abstract

This thesis studies controllability of generative models (specifically VAEs and GANs) applied primarily to images. We improve 1. generation quality, by removing the arbitrary prior assumptions, 2. classification by suitably choosing the latent space distribution, and 3. inference performance by optimizing the generative and inference objective simultaneously.

Variational autoencoders (VAEs) are an incredibly useful tool as they can be used as a backbone for a variety of machine learning tasks *e.g.*, semi-supervised learning, representation learning, unsupervised learning, *etc*. However, the generated samples are overly smooth and this limits their practical usage tremendously. There are two leading hypotheses to explain this: 1. bad likelihood model and 2. overly simplistic prior. We investigate these by designing a deterministic yet samplable autoencoder named Regularized Autoencoders (RAE). This redesign helps us enforce arbitrary priors over the latent distribution of a VAE addressing hypothesis (1) above. This leads us to conclude that a poor likelihood model is the predominant factor that makes VAEs blurry. Furthermore, we show that combining generative (*e.g.*, VAE objective) and discriminative objectives (*e.g.*, classification objective) improve performance of both. Specifically, We use a special case of an RAE to build a classifier that offers robustness against adversarial attack.

Conditional generative models have the potential to revolutionize the animation industry, among others. However, to do so, the two key requirements are, 1. they must be of high quality (*i.e.*, generate high-resolution images) and 2. must follow their conditioning (*i.e.*, generate images that have the properties specified by the condition). We exploit pixel-localized correlation between the conditioning variable and generated image to ensure strong association between the two and thereby gain precise control over the generated content. We further show that closing the generation-inference loop (training them together) in latent variable models benefits both the generation and the inference component. This opens up the possibility to train an inference and a generative model simultaneously in one unified framework, in the fully or semi supervised setting.

With the proposed approach, one can build a robust classifier by introducing the marginal likelihood of a data point, removing arbitrary assumptions about the prior distribution, mitigating posterior-prior distribution mismatch and completing the generation inference loop. In this thesis, we study real-life implications of each of the themes using various image classification and generation frameworks.

Zusammenfassung

Diese Doktorarbeit untersucht die Kontrollierbarkeit generativer Modelle (insbesondere VAEs und GANs), angewandt hauptsächlich auf Bilder. Wir verbessern 1. die Qualität der generierten Bilder durch das Entfernen der willkürlichen Annahme über den Prior, 2. die Performanz der Klassifikation durch das wählen einer passenden Verteilung im latenten Raum und 3., die Inferenzperformanz durch die gleichzeitige Optimierung einer Kostenfunktion für die Generierung und Inferenz.

Variationale Autoencoder (VAEs) sind ein sehr nützliches Werkzeug, da sie als Basis für eine Vielzahl von Aufgaben im Bereich „Maschinelles Lernen“ verwendet werden können, wie beispielsweise für teilüberwachtes Lernen, lernen von Repräsentationen, und unüberwachtem Lernen, usw. Die von VAEs generierten Bilder sind meist stark geglättet, was die praktische Anwendung deutlich limitiert. Als Erklärung hierfür dienen zwei Hypothesen: erstens, ein schlechtes Modell der Likelihood and zweitens, einen zu einfachen Prior. Wir untersuchen diese Hypothesen durch das Erstellen eines deterministischen Autoencoders, den wir regularisierten Autoencoder (RAE) nennen, von dem Stichproben gezogen werden können. Diese Ergänzung erlaubt es uns beliebige Prior-Verteilungen im latenten Raum vorzugeben, wodurch wir Hypothese Eins untersuchen. Diese Untersuchung führt zur Schlussfolgerung, dass der Hauptgrund für die verschwommenen Bilder eines VAEs ein schlecht gewähltes Prior Modell ist. Des Weiteren zeigen wir, dass die Kombination generativer (z.B. VAE-Objektiv) und diskriminativer (z.B. Klassifikatoren) Kostenfunktionen die Performanz für beide steigert. Dafür verwenden wir eine spezielle Variante eines RAE zum Erstellen eines Klassifikators, der robust gegen „Adversarial Attacks“ ist.

Konditionierte generative Modelle haben das Potential die Animationsindustrie, neben anderer Industrien, zu revolutionieren. Um dies zu erreichen müssen zwei Schlüsselvoraussetzungen erfüllt werden: erstens eine hohe Qualität der generierten Daten (d.h. die Erzeugung von hoch auflösenden Bildern) und zweitens die generierten Daten müssen ihrer Konditionierung folgen (d.h. erzeugte Bilder müssen die durch die Konditionierung festgelegten Eigenschaften erfüllen). Wir verwenden die Pixel-lokalisierte Korrelation zwischen der Konditionierungsvariable und dem generierten Bild, um einen starken Zusammenhang zwischen beiden sicherzustellen. Dadurch erhalten wir präzise Kontrolle über die generierten Daten. Darüber hinaus zeigen wir, dass das Schließen des Generations-Inferenz Kreises (beide gemeinsam trainieren) von latenten Variablenmodellen zur Verbesserung von sowohl der Generierungskomponente als auch der Inferenzkomponente führt. Dies ermöglicht das gemeinsame Trainieren eines generativen Modells und eines Modells für Inferenz in einem einheitlichen Rahmen. Dies ist sowohl im überwachten, als auch im teilüberwachten Lernen, möglich.

Mit diesem vorgeschlagenen Ansatz ist es möglich einen robusten Klassifikator zu trainieren, durch die Verwendung der Marginalen Likelihood eines Datenpunktes, der Entfernung der willkürlichen Annahme über den Prior, der Abmilderung der Diskrepanz zwischen Prior- und Posterior-Verteilung, und des Schließens des Generations-Inferenz Kreises. In dieser Arbeit untersuchen wir die Implikationen von jedem dieser Themen in vielfältigen Aufgaben der Bildklassifizierung und Bildgenerierung.

Acknowledgements

I owe a great deal to Michael Black, who created a wonderful place to work named the Perceiving Systems Department, hosted a sharply focused group of people whose company has enriched me over the years. I am also grateful to all the members of Perceiving Systems department for the fruitful discussions and their capability of grounding abstract concepts to the real-life problems.

My collaboration and frequent discussions with the Empirical Inference department headed by Bernhard Schölkopf has greatly widened the perspective and improved the quality of this thesis. Specifically, I would like to thank my coauthors Mehdi Sajjadi and Antonio Vergari for helping me with various aspects of life.

I thank Mr. Arnav Acharyya for his invaluable contribution to the discussions that helped shape many of my projects and for inspiring me to keep a theoretical perspective alive in this age of empiricism.

I especially thank Soubhik Sanyal and Marilyn Keller for being a close friend, being a critical analyst for all my new ideas and constantly reminding and remembering to keep a healthy balance of work and regular life.

I also thank Alexander Neitz, Anant Raj, Alex Vorobiov, Javier Romero, Betty Mohler-Tesch and Xiaochen Hu for their insightful comments and intriguing discussions. The research on invertible GANs was fully funded and supported by Amazon, for that, I express my gratitude.

Finally, I thank Timo Bolkart and David Hoffmann for helping me with various aspects of my scientific career, ranging from discussing future research ideas to translating abstracts.

Contents

1 Introduction **1**
1.1 Motivation . . . 2
1.2 Outlook . . . 3

2 Generative Models **5**
2.1 Functions and Relations . . . 5
2.2 Compression-based models . . . 5
2.2.1 Variational Autoencoders . . . 6
2.2.2 Generative Adversarial networks (GAN) . . . 7
2.3 Denoising models . . . 8
2.3.1 Diffusion models . . . 8
2.4 Log-likelihood-based methods . . . 9
2.4.1 Autoregressive models . . . 9
2.4.2 Normalizing flow models . . . 9
2.4.3 Energy-based models . . . 10

3 From Variational to Deterministic Auto-encoder **13**
3.1 Introduction . . . 13
3.2 Practice and shortcomings of VAEs . . . 15
3.3 Reconstruction and regularization trade-off . . . 17
3.4 Constant-Variance Encoders . . . 17
3.5 Deterministic Regularized Autoencoders . . . 18
3.5.1 Regularization Schemes for RAEs . . . 19
3.6 Ex-Post Density Estimation . . . 20
3.7 Related works . . . 20
3.8 Experiments . . . 22
3.8.1 RAEs for image modeling . . . 22
3.8.2 GrammarRAE: modeling structured inputs . . . 24
3.9 Conclusion . . . 25

4 Gaussian mixture as Priors of VAEs **27**
4.1 Resisting adversarial attacks using Gaussian mixture variational autoencoders . 27
4.2 Adversarial Attacks and Defenses . . . 27
4.3 Approach . . . 28
4.4 Related Works . . . 29
4.5 GMM-VAE . . . 29
4.5.1 Modifying the Evidence Lower Bound . . . 29
4.5.2 Supervision using a Gaussian Mixture Prior . . . 31
4.6 Resisting adversarial attacks . . . 32

4.7 Reclassification . . . 33
4.8 Experiments . . . 33
4.8.1 Implementation details . . . 33
4.8.2 Qualitative evaluation. . . . 34
4.8.3 Numerical results. . . . 34
4.8.4 Adversarial attacks on encoder. . . . 35
4.8.5 White-box adversarial attack. . . . 35
4.8.6 Fooling images. . . . 36
4.8.7 Entropy of the posterior . . . 38
4.9 Discussion . . . 39
4.10 Conclusion . . . 39

5 Controlling the Generative Adversarial Networks 41
5.1 Inductive Bias . . . 41
5.2 Conditional GANs . . . 42
5.3 Generative Interpretable Faces . . . 43
5.3.1 Introduction . . . 43
5.3.2 Related Work . . . 44
5.3.3 Preliminaries . . . 47
5.3.4 Method . . . 48
5.3.5 Experiments . . . 51
5.3.6 Discussion . . . 55
5.3.7 Conclusion . . . 56

6 Invertible GANs (InvGAN) 59
6.1 Introduction . . . 59
6.2 Related work . . . 60
6.3 Method . . . 61
6.3.1 Architecture . . . 62
6.3.2 Objective . . . 62
6.3.3 Dealing with resolutions higher than training resolution . . . 65
6.4 Experiments . . . 65
6.4.1 Semantically consistent inversion using InvGAN . . . 66
6.4.2 Suitability for image editing . . . 68
6.4.3 Tiling to boost resolution . . . 68
6.4.4 Ablation studies . . . 69
6.4.5 Discussion and future work . . . 72
6.4.6 Conclusion . . . 73

7 Conclusion 75
7.1 Contributions . . . 75
7.1.1 RAE . . . 75
7.1.2 GMM-VAE . . . 75
7.1.3 GIF . . . 76
7.1.4 InvGAN . . . 76

7.2 Open Questions 76
7.2.1 Likelihood-based vs adversarial models 77
7.2.2 Quantifying generative capacity 77
7.2.3 Training Instability of GANs 77
7.3 Condition cross-talk 77
7.4 Modular Generative models 78
7.5 Future Directions 78
7.5.1 GAN as an Energy-Based Model 78
7.5.2 Assemble GAN 80
7.6 Discussions 80

A Regularized autoencoders **83**
A.1.1 A Probabilistic Derivation of Regularization 83
A.1.2 Network architecture, Training Details and Evaluation 84
A.1.3 Evaluation Setup 85
A.1.4 Evaluation by Precision and Recall 86
A.1.5 More Qualitative Results 91
A.1.6 Investigating Overfitting 93
A.1.7 Visualizing Ex-Post Density Estimation 94
A.1.8 Combining multiple regularization terms 96

B Generative interpretable Faces **97**
B.1.1 Vector conditioning architecture 97
B.1.2 Image centering for extreme rotations 97

C Invertible GANs **99**

Bibliography **103**

List of Figures

3.1 (Left) Test reconstruction quality for a VAE trained on MNIST with different numbers of samples in the latent space as in Eq. (3.7) measured by FID (lower is better). Larger numbers of Monte-Carlo samples clearly improve training, however, the increased accuracy comes with larger requirements for memory and computation. In practice, the most common choice is, therefore $k = 1$. (Right) Reconstruction and random sample quality (FID, y-axis, lower is better) of a VAE on MNIST for different trade-offs between $\mathcal{L}_{\mathsf{REC}}$ and $\mathcal{L}_{\mathsf{KL}}$ (x-axis, see Eq. (3.5)). Higher weights for $\mathcal{L}_{\mathsf{KL}}$ improve random samples, but hurt reconstruction. This is especially noticeable towards the optimality point ($\beta \approx 10^1$). This indicates that enforcing structure in the VAE latent space leads to a penalty in quality. 17
3.2 Qualitative evaluation of sample quality for VAEs, WAEs, 2sVAEs, and RAEs on CelebA. RAE provides slightly sharper samples and reconstructions while interpolating smoothly in the latent space. Corresponding qualitative overviews for MNIST and CIFAR-10 are provided in Appendix A.1.5 in the interest of completeness. GT here represents groundtruth. 21
3.3 Generating structured objects by GVAE, CVAE and GRAE. (Upper left) Percentage of valid samples and their average mean score (see text, Sec. 3.8.2). The three best expressions (lower left) and molecules (upper right) and their scores are reported for all models. 24

4.1 (a) The model at training time. All the inputs are in green, while all the losses are in brown. (b) Model pipeline at inference time. The red dot shows that the attacker is successful in fooling the encoder by placing its output in the wrong class. However, it results in a high reconstruction error, since the decoder generates an image of the target class. 30
4.2 Generated images from different classes of MNIST, COIL-100, SVHN. 34
4.3 We run FGSM with varying ε on the models trained on MNIST data in both supervised and semi-supervised scenarios. Although the error rate is higher for the semi-supervised network, the rejection ratio rises monotonically for both networks with increasing ε, and the error rate for the supervised model stays below 5%. 36
4.4 White box attack on MNIST and COIL dataset. (a) Targeted attack on MNIST. (b) Targeted attack on COIL. (c) Targeted fooling sample attack on MNIST. The first row represents the images to which the white-box attack converged, and the second row represents the corresponding reconstructed images. 37

4.5 In Fig. 4.5a We run FGSM with varying ε on the model with variable encoder distribution entropy, trained on MNIST data. The rejection rate stays low in this case, while the error rate increases with increasing ε. In Fig. 4.5b We present the reclassification accuracy for samples generated using FGSM on the MNIST dataset. 38

5.1 Face images generated by controlling FLAME [Li et al., 2017] parameters, appearance parameters, and lighting parameters. For shape and expression, two principal components are visualized at ± 3 standard deviations. The pose variations are visualized at $\pm\pi/8$ (head pose) and at $0, \pi/12$ (jaw pose). For shape, pose, and expression, the two columns are generated for two randomly chosen sets of appearance, lighting, and style parameters. For the appearance and lighting variations (right), the top two rows visualize the first principal components of the appearance space at ± 3 standard deviations, the bottom two rows visualize the first principal component of the lighting parameters at ± 2 standard deviations. The two columns are generated for two randomly chosen style parameters. 45

5.2 StyleGAN synthesis block. [A] represents a learned affine transform produced from the style vector, and [B] represents a broadcasting operation that is used to inject noise responsible for creating localized stochasticity in the generated image. As will be found later, (Fig. 5.3), we modify this noise broadcasting operation to inject the conditioning signal into the generator. 48

5.3 Our generator architecture is based on StyleGAN2 [Karras et al., 2019b], A is a learned affine transform, and B stands for per-channel scaling. We make several key changes, such as introducing 3D model generated condition through the noise injection channels and introduce texture consistency loss. We refer to the process of projecting the generated image onto the FLAME mesh to obtain an incomplete texture map as *texture stealing*. 49

5.4 Impact of individual parameters, when being exchanged between two different generated images one at a time. From left to right we progressively exchange style, shape, expression, head and jaw pose, appearance, and lighting of the two parameter sets. We progressively change the color of the parameter symbol that is effected in every column to red. 52

5.5 Images obtained by randomly sampling FLAME, appearance parameters, style parameters, and lighting parameters. Specifically, for shape, expression, and appearance parameters, we sample parameters of the first three principal components from a standard normal distribution and keep all other parameters at zero. We sample pose from a uniform distribution in $[-\pi/8, +\pi/8]$ (head pose) for rotation around the y-axis, and $[0, +\pi/12]$ (jaw pose) around the x-axis . . . 53

5.6 Combination of GIF and an existing speech-driven facial animation method by generating face images for FLAME parameters obtained from VOCA [Cudeiro et al., 2019]. Sample frames to highlight jaw pose variation. Please see the video for the full animation at – `https://youtu.be/-ezPAHyNH9s?&start=230&end=246`. 54

5.7 An example of the experiment window shown to the AMT participants to compare between GIF and an ablated version of itslef. 54

5.8 An example of the experiment window shown to the AMT participants to quantify Geometry-style disentanglement. It contains a random style vector and a flame parameter from the training set. 55
5.9 (5.9a) Preference frequency of styles on a 5-Point Likert Scale. Note that almost all style vectors, represented with different colors here, get a similar distribution of likeness ratings indicating that they do not influence the perceived FLAME conditioning. (5.9b) To evaluate shape error, we generate 1024 random faces from GIF and re-infer their shape using DECA [Feng et al., 2021]. Using FLAME, we compute face region vertices twice: once with GIF's input condition and once with re-inferred parameters. Finally, the RMSE between these face region vertices are computed. The process is repeated for expression and pose. 56

6.1 We train InvGAN following a regular GAN. We use a second output head in the discriminator besides the real fake decision head, to infer the latent-code z of a given image. Here ⃠ denotes no gradient propagation during the back propagation step. It also denotes 'no training' when it is placed on a model. We use red to show data flow corresponding to real images. 64
6.2 InvGAN reconstructions of CelebA at 128×128 resolution. Alternating (from left to right) original and reconstructed images. We show more such examples in Appendix C 66
6.3 Benchmark image editing tasks on FFHQ (128 px). Style mixing: We transfer the first $0, 1, 2, \ldots, 11$ style vectors from one image to another. For the other image editing tasks, pairs of images are input image (left) and reconstruction (right). 69
6.4 Super resolution using extrapolation in the tiled latent space. From left, we visualize the original image, the low-resolution version of it, the reconstruction of the low-resolution version, and progressive extrapolation to achieve deblurring. 70
6.5 Tiled reconstruction of random (a) FFHQ Images and (b) ImageNet images. The left column shows the real images, the second shows the patch by patch reconstructions, and the third shows the absolute pixel-wise differences. Note that interestingly, though the patches are reconstructed independently of each other, the errors lie mostly on the edges of the objects in the images, arguably the most information dense region of the images. 71
6.6 Inversion of held out test samples. Columns are in groups of three: the first column holds real images, second their reconstruction and third the absolute pixel-wise difference. (a) Inversion using naive model, *i.e.*,only z–reconstruction loss is used, (b) inversion using model that uses latent codes from real samples, *i.e.*,the augmented naive model. (c) our full model. Notice how the imperfections in the reconstructions highlighted with red boxes gradually turns green as the model improves. 72

7.1 (5.9a) Simple shapes data set. From left in alternative columns, the foreground objects and their bounding box. The bounding boxes are used as conditioning for the generator. 7.1b Two instances of the generative process of Assemble GAN. 81

A.1 PRD curves of all RAE methods (left), reflects a similar story as FID scores do. RAE-SN seems to perform the best in both precision and recall metric. PRD curves of all traditional VAE variants (middle). Similar to the conclusion predicted by FID scores, there is no clear winner. PRD curves for the WAE (with isotropic Gaussian prior), WAE-GMM model with ex-post density estimation by a 10-component GMM and RAE+SN-GMM (right). This finer grained view shows how the WAE-GMM scores higher recall but lower precision than a RAE+SN-GMM while scoring comparable FID scores. Note that ex-post density estimation greatly boosts the WAE model in both PRD and FID scores. 87

A.2 PRD curves of all methods on image data experiments on MNIST. For each plot, we show the PRD curve when applying the fixed or the fitted one by ex-post density estimation (XPDE). XPDE greatly boosts both precision and recall for all models. 88

A.3 PRD curves of all methods on image data experiments on CIFAR10. For each plot, we show the PRD curve when applying the fixed or the fitted one by ex-post density estimation (XPDE). XPDE greatly boosts both precision and recall for all models. 89

A.4 PRD curves of all methods on image data experiments on CELEBA. For each plot, we show the PRD curve when applying the fixed or the fitted one by ex-post density estimation (XPDE). XPDE greatly boosts both precision and recall for all models. 90

A.5 Qualitative evaluation for sample quality for VAEs, WAEs and RAEs on MNIST. Left: reconstructed samples (top row is ground truth). Middle: randomly generated samples. Right: spherical interpolations between two images (first and last column). 91

A.6 Qualitative evaluation for sample quality for VAEs, WAEs and RAEs on CIFAR-10. Left: reconstructed samples (top row is ground truth). Middle: randomly generated samples. Right: spherical interpolations between two images (first and last column). 92

A.7 Nearest neighbors to generated samples (leftmost image, red box) from training set. It seems that the models have generalized well and fitting only 10 Gaussians to the latent space prevents overfitting. 93

A.8 Different density estimations of the 2-dimensional latent space of a VAE learned on MNIST. The blue points are 2000 test set samples while the orange ones are drawn from the estimator indicated in each column: isotropic Gaussian (left), multivariate Gaussian with mean and covariance estimated on the training set (center) and a 10-component GMM (right). This clearly shows the aggregated posterior mismatch w.r.t. to the isotropic Gaussian prior imposed by VAEs and how ex-post density estimation can help fix the estimate. 94

A.9 Different density estimations of the 16-dimensional latent spaces learned by all models on MNIST (see Table 1) here projected in 2D via T-SNE. The blue points are 2000 test set samples while the orange ones are drawn from the estimator indicated in each column: isotropic Gaussian (left), multivariate Gaussian with mean and covariance estimated on the training set (center) and a 10-component GMM (right). Ex-post density estimation greatly improves sampling the latent space. 95

B.1 Architecture of Vector conditioning model. Here $+\!\!+$ represents a concatenation. 97
B.2 In order for the eyes to be places at a given pixel location a profile view causes the face image to be highly zoomed in. This causes the generated images to become unrealistic. 98

C.1 Tiled reconstruction of a video sequence. (a) original sequence, (b) reconstructed sequence, (c) up-sampled in time sequence. 99
C.2 InvGAN reconstructions of CelebA at 128×128 resolution. Alternating (from left to right) original and reconstructed images. 100
C.3 Benchmark image editing tasks on CelebA (128 px). Style mixing: We transfer the first $0, 1, 2, \ldots, 11$ style vectors from one image to another. For the other image editing tasks, pairs of images are input image (left) and reconstruction (right). 102

List of Tables

3.1 Evaluation of all models by FID (lower is better, best models in bold). We evaluate each model by REC.: test sample reconstruction; $\mathcal{N}$: random samples generated according to the prior distribution $P(Z)$ (isotropic Gaussian for VAE / WAE, another VAE for 2SVAE) or by fitting a Gaussian to $q_\delta(Z)$ (for the remaining models); GMM: random samples generated by fitting a mixture of 10 Gaussians in the latent space; Interp.: mid-point interpolation between random pairs of test reconstructions. The RAE models are competitive with or outperform previous models throughout the evaluation. Interestingly, interpolations do not suffer from the lack of explicit priors on the latent space in our models. 22

4.1 Comparison between the performance of the state-of-the-art (SOTA) models and our model. We show that our method, even without much fine-tuning focused on achieving classification accuracy, is competitive with the SOTA. MNIST SOTA is as reported by Wan et al. [2013], SVHN SOTA is as given by Lee et al. [2016] and the SOTA for COIL-100 is given by Wu et al. [2015]. Wh. Th. here represents without threshold and Th. represents with threshold. . 34

5.1 (5.1a) AMT ablation experiment. Preference percentage of GIF generated images over ablated models and vector conditioning model. Participants were instructed to “pay particular attention to shape, pose, and expression and ignore image quality”. (5.1b) FID scores of images generated by GIF and ablated models (lower is better). Note that this score only evaluates image quality and does not judge how well the models obey the underlying FLAME conditions. . 54

6.1 Here we report random sample FID (RandFID), FID of reconstructed random samples (RandRecFID), FID of reconstructed test set samples (TsRecFID), FID of the linear middle interpolation of test set images (IntTsFID) and reconstruction per pixel per color channel mean absolute error when images are normalized between ± 1, also from test set. The last three rows are our models. All FID scores are here evaluated against the training set using 500 and, 50000 samples. They are separated by ‘/’. For the traditional MSE optimization based and In-Domain GAN inversion, the MSE errors are converted to MAE by taking square root and averaging over the color channels and accounting for the re-normalization of pixel values between ± 1. Runtime is given in seconds per image. We ran them on a V100 32GB GPU and measured wall clock time. . . . 67

6.2 Mean average precision for a ResNet50 attribute classifier on CelebA, averaged over 20 attributes. We report the performance for training on the original dataset, the reconstructed dataset using the tiling-based method pre-trained on ImageNet and the reconstruction on InvGAN trained on the CelebA training set directly. 68

A.1 Detailed network architecture used for our experiments for different datasets. . 85
A.2 Evaluation of random sample quality by precision / recall [Sajjadi et al., 2018] (higher numbers are better, best value for each dataset in bold). It is notable that the proposed ex-post density estimation improves not only precision, but also recall throughout the experiment. For example, WAE seems to have a comparably low recall of only 0.88 on MNIST, which is raised considerably to 0.95 by fitting a GMM. In all cases, GMM gives the best results. Another interesting point is the low precision but high recall of all models on CIFAR-10 – this is also visible upon inspection of the samples in Fig. A.6. 86
A.3 Comparing multiple regularization schemes for RAE models. The improvement in reconstruction, random sample quality and interpolated test samples is generally comparable, but hardly much better. This can be explained with the fact that the additional regularization losses make tuning their hyperparameters more difficult, in practice. 96

C.1 Random sample FID (RandFID), FID of reconstructed random samples (RandRecFID). FID scores are here evaluated using 500 and 50000 samples. They are separated by '/'. 101

Chapter 1

Introduction

Graphical models capture the causal process by which the observed data is generated [Pearl, 1988]. Therefore, these models are sometimes also called generative models. Here and throughout the rest of this thesis, we relax the notion of a generative model to the following definition. Generative models are the broad category of models that have either implicit or explicit notion of the probability distribution of the data they operate on. They often yield a mechanism to efficiently sample from this distribution. That is, we do not necessarily intend to recover the true causal factors of data generation, but will be content with any set of parameters that represent the data distribution. Naturally, the training mechanism of such a method can be thought of as a density estimation task. Therefore, deep generative models, are an extension of well established density estimators, *e.g.*, parzen-window-estimator [Parzen, 1962], kernel-density-estimator [Rosenblatt, 1956], vector quantization [Fischer, 1986], re-scaled histogram [Wikipedia, 2022a], *etc*. What received renewed focus in the field, however, is the incorporation of an efficient way of sampling from such estimators, especially when dealing with high dimensional data. This often was studied separately from the density-estimation effort in the past. Another crucial point where modern generative models differ from the traditional density estimation methods is that they often represent the probability density only implicitly. That is, for a method to be called a generative model, it must allow sampling but may not necessarily allow density evaluation at a given point.

Modern ways of learning a generative model are Variational Autoencoders (VAEs), [Kingma and Welling, 2014], Generative Adversarial Networks (GAN) [Goodfellow et al., 2014], Normalizing Flows [Tabak and Vanden-Eijnden, 2010], Diffusion models [Sohl-Dickstein et al., 2015], and Auto-regressive models [Aäron et al., 2016] to name a few. To obtain a tractable training objective for each of these models, one must make some key assumptions. Chapter 3 and 4 of this thesis focu and sometimes challenge some of these assumptions in VAEs. This exploration leads to an application of VAEs in adversarial robustness in Chapter 4.

Recent focus on random sampling has led to the specialized study of conditional density estimation as a separate subfield. This distinction between conditional and unconditional density estimation leads to especially distinct models, applications, and evaluation strategies. One of the key factors that drives design choices of models is condition representation. This is the case because often the conditions have drastically different characteristics as compared to the data they influence, *e.g.*, class-categorical variable conditioning image-data. This thesis poses conditional distribution learning as a way of learning one-to-many mapping. Such mappings violate the definition of a function. Based on this observation, we dive into the effects of representation of the conditional variables. Finally, we study a generalized framework of density estimation known as Energy-based models [Song and Kingma, 2021]. We briefly introduce them in Sec 2.4.3 and later in Sec. 7.5.1 we reinterpret GANs as energy-based models and draw inspiration for future work.

1.1 Motivation

Following the phenomenal success of deep neural network-based classification models in 2012 [Krizhevsky et al., 2012], artificial neural networks (ANNs) have attracted a lot of attention. The flip side of classification, which can also be regarded as pattern recognition module, is pattern generation. Intuitively, if one iteratively changes an input image, starting from a random initialization, such that a classifier's output converges to a specific class, the modified input would look like a member of the class indicated by the label. However, this is not the case in practice. In practice, the output label goes to the desired label, but the input still looks like random noise. These are often called fooling samples [Nguyen et al., 2015]. Moreover, when started from an image of a random class, the label changes to any other desired class, but the input image still looks unchanged converging to the so-called adversarial samples [Goodfellow et al., 2015a]. This prohibits classifiers from being used as generative models. However, success of generative models, especially GANs, on image data, showcases complimentary strengths. Given this complimentary prowess of generative and classification models, the following questions arise naturally—i) Can generative models be combined with discriminative models to enhance each other's performance? ii) Can one effectively use the representations learned by generative models? iii) Can generative models be used in semi and unsupervised settings. We, in this thesis, intend to shed some light on these questions.

One of the most popular and theoretically motivated modern generative models is a variational autoencoder (VAE) [Kingma and Welling, 2014]. However, learning under the VAE paradigm still poses unanswered theoretical questions and considerable practical challenges. We devise an alternative generative framework that is simpler, easier to train, and deterministic; yet it retains many of the advantages of VAEs [Ghosh et al., 2020b]. We observe that sampling a stochastic encoder in a Gaussian VAE[1] can be interpreted as simply injecting noise into the input of a deterministic decoder. We investigate how substituting this kind of stochasticity, with other explicit and implicit regularization schemes, can lead to an equally smooth and meaningful latent space without having to force it to conform to an arbitrarily chosen prior. To retrieve a sampling mechanism, we introduce an ex-post density estimation step that can be readily applied to the proposed framework as well as existing VAEs. We call models thus learned Regularized Autoencodes (RAEs). Although this greatly improves training stability and simplifies VAE based generative framework, RAEs still lag behind GANs in terms of image quality. The apparent failure of likelihood-based methods on high-resolution images makes one wonder if high-resolution images might be fundamentally different from their lower resolution counterpart, *e.g.*, CIFAR [Krizhevsky and Hinton, 2009], MNIST [LeCun et al., 1998b], etc.

Over the years, although ANN classifiers have achieved super human performance, one baffling aspect of them had been their vulnerability towards fooling and adversarial samples [Goodfellow et al., 2015b]. Stranger still is the ease of finding such samples. So far, all efforts to harden classifiers against such attacks have seen limited success. "Adversarial samples" and

[1]We call a VAE with Gaussian prior a Gaussian VAE

“Fooling samples”, have been tackled separately so far due to the difficulty posed when considered together. We harness the power of generative models and show how one can defend against them both under a unified framework. Our model has the form of a variational autoencoder, [Kingma and Welling, 2014], which we discuss in detail in Sec. 4.1. Moreover, we extend the VAE framework to have a Gaussian mixture prior over the latent variable. We show how selective classification can be performed using this model, thereby causing the objective that finds adversarial samples to entail a conflict. The proposed method leads to the rejection of adversarial samples instead of misclassification, while maintaining high precision and recall on test data. It implicitly provides an adversarially robust classifier, which can be trained in either a supervised or semi-supervised way.

Apart from bolstering classifier performance and estimating probability densities, high quality generative models can augment or even replace the traditional graphics pipeline. With the long-term sight of many set on a highly realistic metaverse, the strengths and weakness of the generative models are being studied intensely. In this context, we study specifically the conditional generation of human faces. We choose this sub problem since evaluation of condition association in a generative process is an active research area. Hence, evaluation has to rely on human judgement. Because of evolutionary reasons, we are highly sensitive towards detailed expression and subtlety in the human face. Furthermore, photo-realistic visualization and animation of expressive human faces find numerous application in gaming and computer graphics. Despite the aforementioned motivation and abundance of effort in this field, photorealistic rendering of human faces remains a challenge. Traditional 3D face modelling methods provide parametric control but generates unrealistic images, on the other hand, generative 2D models like GANs (Generative Adversarial Networks) output photo-realistic face images, but lack explicit control. Recent methods offer partial control, either by attempting to disentangle different generative factors in an unsupervised manner, or by adding control mechanisms post hoc to a pre-trained model. Unconditional GANs, however, may entangle factors of interest that are hard to undo later. In Sec. 5.3 we introduce generative interpretable faces (GIF) and study in detail how one can address such issues.

Although GIF enables us to perform conditional sampling using GANs, we still lack an inference mechanism. This blocks us from harnessing the powerful GAN latent space for downstream tasks, *e.g.*, semantic data editing, in-painting, transfer learning *etc*. This greatly limits the practical utility of GANs. Despite numerous efforts to train an inference model and to design an iterative method to invert a pre-trained generator, previous methods are specific to datasets (*e.g.*, human face images) and architectures (*e.g.*, StyleGAN). These methods are nontrivial to extend to novel datasets and architectures. We propose a general framework that is agnostic to these factors, [Ghosh et al., 2022]. Our key insight is that, by training the inference and the generative model together, we allow them to adapt to each other and to converge better. Our model InvGAN, short for Invertible GAN, successfully embeds real images in the latent space of a high quality generative model. This allows us to perform image in-painting, merging, interpolation, and data augmentation.

1.2 Outlook

In the remainder of the thesis, we explore in detail all the components described above. Since this thesis naturally splits into well segmented modules, we introduce the mathematical tools, the prior art, and the techniques employed at the beginning of the corresponding sections.

In Chapter 2 we introduce the general distinction of different kinds of mappings, specifically we explore modeling of one-to-many mappings using neural networks. A one-to-many mapping, however, violates the definition of a function. Thus, many modern methods that simply ignore this suffer from inter-sample averaging. This is of particular interest to the supervised regression-based inference community as, modeling a many one-to-many mapping in such a setting will only produce models with reduced performance even with powerful models and abundance of data. This can, however, be fixed easily by learning conditional probability densities. These in turn heavily rely on the unconditional density estimation methods. In chapter 2 we introduce the most relevant variants of modern density estimation methods, especially when handling extremely high dimensional data such as images.

In Chapter 3, we take a more in-depth look at VAEs, one of the most successful and widely used generative models. We investigate and distinguish between the most essential machinery, and not so essential design choices within VAEs. This study leads to the relaxation of the requirement for a prior latent distribution in our proposed Regularized Autoencoders (RAEs). RAEs enable us to combine discriminative and generative models. We show that such a combination leads to a hardened model against adversarial attacks.

Chapter 3 lets us use an invertible generative model, specifically a RAE, as a generalized relation-learning mechanism, including one to many mappings. However, despite their elegant formulation, VAEs struggle to achieve state-of-the-art performances under certain metrics *e.g.*, reconstruction sharpness. Generative Adversarial Networks (GANs), conclusively solve such problems. On the other hand, they introduce their own new set of problems, *e.g.*, unstable training dynamics and poor data coverage. In Chapter 5, we study targeted representational and architectural biases in order to solve several real life problems, *e.g.*, disentanglement, conditional generation, etc. Specifically, we study photorealistic generation of images of human faces conditioned on semantically meaningful parameters, such as shape, pose and skin tone.

In Sec. 6 we address the problem posed by the blurring effect of auto-encoders. It is clear that for many real life applications a meaningful latent embedding of images is necessary, yet no method is known to produce satisfactory results on high-resolution images. Our method therefore inverts a GAN. We demonstrate it on in-painting, image merging, frame interpolation, etc. Furthermore, even though GANs show incredible potential, they are not suitable for generation of super high-resolution ($> 1024 \times 1024$ resolution) images primarily due to explosion of memory requirement. We sidestep this issue by segmenting the generated image and reusing our ex-post-density-estimation idea from Sec.3.6

Finally, in Chapter 7, we summarize our contribution and key findings. More importantly, we specify many open problems that, to us, feel as approachable with current technology. We also detail two future directions of research that are slightly more divergent. One of these is directed at supervised learning of a decomposition of generative factors. The other is directed at solving the instability and coverage problems of the GAN framework. Furthermore, it predicts the existence of a stronger generative model.

Chapter 2

Generative Models

2.1 Functions and Relations

A function $f : \mathcal{X} \rightarrow \mathcal{Y}$ can be thought of as a set of ordered pairs (x, y) with $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, such that for every element of $\mathcal{X}$, there exist one and only one element of $\mathcal{Y}$. Here $\mathcal{X}$ is called the domain and $\mathcal{Y}$ the codomain of the function [Wikipedia, 2022b]. This definition, however, somewhat restricts the possible binary relations the elements of the two sets might have. As such, a binary relationship R between two sets $\mathcal{X}$ and $\mathcal{Y}$ is defined as a subset of the Cartesian product of the respective sets, *i.e.*, $R \subseteq \mathcal{X} \times \mathcal{Y}$. Therefore, in the general case, one or more elements of the domain can be mapped to one or more elements in the codomain. Such relations, however, cannot be modelled using traditional neural networks as they are members of the space of continuous functions. One way of extending neural networks such that they can model not only functions but generalized relations is to use them to represent conditional probability densities and to have a mechanism to sample from them. In the following sections, we explore several ways to achieve this. To do so, however, we must study unconditional density estimation. Extending it to the conditional setting is often a relatively small step. Keeping convenience and continuity in mind, we present the conditional versions of unconditional density estimation techniques, together.

2.2 Compression-based models

A latent variable model in statistics is a model that relates a set of observable variables to a set of unobserved or latent variables. It is often assumed that the realizations of the observed variable are the results of a particular setting of the latent variables. We also assume that two different realizations of the observed variable have noting to do with each other if the latent variables are accounted for. Imagine that one asks an artist to compose a painting. The painter before making any stroke on his/her canvas must first decide on the content of the drawing, *e.g.*, the objects, their layout, lighting, inter-object interactions, *etc*. Yet when complete we do not see these decisions directly (latent variable), we can only observe the full images (observable variable). We can, however, think back in time and say, that, the painter must have wanted to draw a landscape or a seascape during sunset before he/she started, but we do not get to observe the artist's 'state-of-mind' directly. That stays 'latent'.

Since the methods described in this thesis builds mainly upon two generative models, namely Variational autoencoders (VAEs) [Kingma and Welling, 2014] and Generative Adversarial networks (GANs) [Goodfellow et al., 2014], we shall introduce them here in detail. As mentioned before, both of these methods, implicitly model the probability distribution of the training data and enable efficient sampling. VAEs, in addition, let one compute a lower bound of the likelihood of training data.

2.2.1 Variational Autoencoders

Since their debut in 2013, VAEs quickly became one of the most popular approaches to learning probability density of a data set. This popularity is driven by VAEs ability to utilize neural networks, they can be trained with simple gradient descent mechanism, and they are theoretically grounded. The most popular application area is to model the distribution of a data set of images. Say $\mathcal{X} = x_{i}{}_{i=1}^{N}$ is a collection of images of a particular kind, that we wish to model by having an implicit estimation of its probability distribution $P_{data}(X)$. Furthermore, say the objective is i) to be able to sample from $P_\theta(X)$ that matches $P_{data}(X)$ and ii) given a realization x_i of the random variable X, we wish to compute the probability that it belongs to the training set. Of course the end objective depends upon a concrete application, however, to develop the general mathematical machinery this objective will suffice. Traditionally, methods [Parzen, 1962, Rosenblatt, 1956, Fischer, 1986] to estimate the probability density of real-world data have suffered some or all of the following drawbacks, i) strong assumptions about the data are required, ii) severe approximations needed to make key quantities tractable, iii) computationally too expensive. VAEs manage to side step almost all of them, or at least drastically alleviate the severity.

VAEs are latent variable models. In reference to the painter's analogy in Sec. 2.2, one can imagine that VAEs first choose the content of the generation and then progressively refine the arrangements of the pixels. Formally, we chose a latent variable Z, that is distributed according to a prior distribution $P(Z)$. This latent variable is usually chosen to be lower dimensional than the original data. We parametrize an evidence model as a conditional distribution $P_\theta(X|Z)$ by a normal distribution $\mathcal{N}(\mu_z, I)$ with unit variance I and mean $\mu = D_\theta(z)$ is a deterministic function $D_\theta(z) : \mathcal{Z} \rightarrow \mathcal{X}$. We obtain the data distribution by marginalizing over the latent variable. This process can be written concretely as in Eq. (2.1). Here θ is a parameter vector that defines the function D.

$$P_\theta(X) = \int_{\mathcal{Z}} P_\theta(X|z)P(z)dz \tag{2.1}$$

Given a dataset $\mathcal{X}$, a straightforward strategy at this point to find the optimal parameters of the function $f_\theta(z)$ would be to perform an $\text{argmax}_\theta \sum_{x\in\mathcal{X}} log(P_\theta(x))$. However, as given by Eq. (2.1) we need to compute an integral for likelihood computation of every point in the data set. This is computationally infeasible. To make it sample efficient for every individual data point, $x_i \in \mathcal{X}$ we perform importance sampling from a distribution $Q_\phi(Z|x_i)$ with tunable parameters given by ϕ. Here, we sample z from a distribution that has information about the data point whose likelihood we shall compute. Therefore, we hope that if we compute $Q_\phi(Z|x_i)$ "intelligently" by choosing a suitable ϕ, we can drastically improve sample efficiency. Guided by this strategy, we rewrite Eq. (2.1) as in Eq. (2.2).

$$\begin{aligned}
log(P_\theta(X)) = log\left(\int_{\mathcal{Z}} P_\theta(X|z)P(z)dz\right) &= log\left(\int_{z\sim Q_\phi(Z|X)} P_\theta(X|z)\frac{P(z)}{Q_\phi(z|X)}dz\right) \\
&\geq \int_{z\sim Q_\phi(Z|X)} log\left(P_\theta(X|z)\frac{P(z)}{Q_\phi(z|X)}\right)dz \\
&= \int_{z\sim Q_\phi(Z|X)} log\left(P_\theta(X|z)\right)dz + \int_{z\sim Q_\phi(Z|X)} log\left(\frac{P(z)}{Q_\phi(z|X)}\right)dz \\
&= \mathbb{E}_{z\sim Q_\phi(Z|X)} log\left(P_\theta(X|z)\right) - \mathbb{KL}(Q_\phi(Z|X)||P(Z))
\end{aligned} \tag{2.2}$$

Here we use Jensen's inequality for a concave function[1], to send the logarithm operator inside the integral. Also, here $\mathbb{KL}$ represents the KL divergence and $\mathbb{E}$ represents the expectation operation. In practice, we choose $P(Z)$, $Q_\phi(Z|X)$ and $P_\theta(X|z)$ to be normal distributions such that the KL divergence term can be analytically computed. Finally, we perform a one sample approximation of the remaining integral. Furthermore, if we choose $P_\theta(X|z) = \mathcal{N}(D_\theta(z), \beta^{-2/k} * I)$, we get the β-VAE [Higgins et al., 2017] objective, which is the most commonly implemented flavor of VAEs.

Although successful, one of the major weakness of VAEs when applied to a dataset of images is that the generated images tend to be blurry. This primarily stems from the assumption that $P_\theta(X|z)$ is normally distributed, resulting in an L_2 distance in the pixel space in the objective function. Moreover, this problem is non-trivial to fix, since one needs a notion of $P_{data}(X)$ to derive a sensible distance function between two realizations of this random variable. However, our objective was to estimate $P_{data}(X)$ in the first place.

2.2.2 Generative Adversarial networks (GAN)

As compared to VAEs, GANs, first proposed by Goodfellow et al. [2014], take a more direct approach to data generation. It recognizes that one key aspect of density estimation is the ability to sample from the true data distribution $P_{data}(X)$. Many a time it is perhaps more important than the ability to evaluate the density function at an arbitrary point. Keeping this in mind, GANs represent their training distribution as the distribution of $G_\theta(z)$ when $z \sim P(Z)$. Where $G_\theta : \mathcal{Z} \to \mathcal{X}$ is a deterministic function parametrized by θ and Z is a random latent variable distributed as an assumed prior distribution $P(Z)$. To train a GAN, we make the assumption that, if a high-capacity function $D_\phi : \mathcal{X} \to \{0,1\}$, also called the discriminator, cannot distinguish between original samples $x_{True} \sim P_{data}(X)$ and generated ones $x_G = G_\theta(z)$, $z \sim P(Z)$, then G_θ, also called the generator, captures the essence of the true data distribution. This intuition leads to the following objective function given in Eq. (2.3)

$$\underset{\theta}{\text{argmin}}\left\{\underset{\phi}{\text{argmax}}\left\{log\left(1 - D_\phi\left(G_\theta(Z)\right)\right) + log\left(D_\phi(x \sim P_{data}(X))\right)\right\}\right\} \tag{2.3}$$

[1]logarithm is a concave function.

When the min-max game is executed alternatively, the objective function in Eq. (2.3) induces a two player game that has a Nash equilibrium when the generated samples follow the true data distribution. For a proof, please refer to [Goodfellow et al., 2014]. This, in practice, turned out to be incredibly useful, since when high capacity neural networks are used as a generator and a discriminator, GANs can model[2] extremely high dimensional data in never seen before quality.

One of the greatest advantages of the GAN framework is that one does not need to design a distance function $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{R}$. Here $\mathcal{R}$ is real number space, which requires knowledge of $P_{data}(X)$. However, this knowledge is not available a priori. Thus, GANs successfully avoid the chicken and egg problem that often plagues autoencoding based methods *e.g.*, in VAEs. On the other hand, GANs provide only a sampling mechanism, *i.e.*, no inference model or likelihood evaluation model is provided. In this sense, a GAN can not be counted as a compression-based model. However, it is assumed that the generator range contains all the data points we are interested in. Therefore, we assume that there exists a compressed latent code for every data point. Because of this reason, we discuss them under the banner of compression-based models.

2.3 Denoising models

Strictly speaking, for this thesis we will not need the diffusion, normalizing flow, autoregressive or energy-based models. However, they are very significant direction of research in the quest of density estimation of high dimensional data. Therefore, for completeness, we will introduce them briefly and point the reader to significant pieces of work for in depth exploration. Furthermore, we will use some knowledge of energy-based models (EBMs) in Sec. 7.5.1, where we link GANs with EBMs. Hence, having an overview will improve clarity.

The ability to sample from a density rather than evaluating it also motivates the diffusion-based probabilistic model, sometimes called just diffusion models. A diffusion process can be modeled as a Markov chain that gradually adds noise to data until it is indistinguishable from noise, *i.e.*, until the signal is destroyed. The idea is to learn to reverse every step of the aforementioned Markov chain. Therefore, the sampling process starts with a random noise vector and slowly attempts to denoise a signal out of it.

2.3.1 Diffusion models

A Markovian noising process of T steps, converts a data point, x_0, gradually through, $x_1, x_2 ...$ to x_T. A particularly easy process uses Gaussian noise with some variance schedule β_t; $t \in [1, T]$. This process, called the forward process, can be described by Eq. (2.4)

$$q(X_t|X_{t-1}) = (\sqrt{1-\beta_t})X_{t-1} + \mathcal{N}(0, \beta_t * I) \tag{2.4}$$

The key idea is to learn a joint density $P_\theta(X_{0:T})$ of the random variables $X_0, X_1 ... X_T$, such that one can evaluate or sample the reverse process $P_\theta(X_{t-1}|X_t)$. It is defined as a Markov process starting at X_T and $P(x_T)$ is assumed to be a standard normal. The learning is driven by maximizing the log likelihood of the data set. i.e. $\theta^* = \text{argmax}_\theta \, log(P_\theta(X_0))$.
We replace $P_\theta(X_0)$ by the marginalized form $\int P_\theta(X_{0:T})dx_{1:T}$. Here we use the shorthand notation $dx_{1:T}$ to represent $dx_1 dx_2 dx_3 ... dx_T$. We get $\theta^* = \text{argmax}_\theta \, log(\int P_\theta(X_{0:T})dx_{1:T}) =$

[2]only implicitly, in the sense that it lets us sample from the density. Nothing else.

$\text{argmax}_\theta\, log(\int q(X_{1:T}|X_0)\frac{P_\theta(X_{0:T})}{q(X_{1:T}|X_0)}dx_{1:T}) = \text{argmax}_\theta\, log\int_{X_{1:T}\sim q(X_{1:T}|X_0)}\frac{P_\theta(X_{0:T})}{q(X_{1:T}|X_0)}dx_{1:T}$. Now using Jensen's inequality and noting that log is a concave function, we get the variational lower bound as shown in Eq. (2.5).

$$log\left(\int_{X_{1:T}\sim q(X_{1:T}|X_0)}\frac{P_\theta(X_{0:T})}{q(X_{1:T}|X_0)}dx_{1:T}\right) \geq \int_{X_{1:T}\sim q(X_{1:T}|X_0)} log\left(\frac{P_\theta(X_{0:T})}{q(X_{1:T}|X_0)}\right)dx_{1:T} \tag{2.5}$$

Now replacing $P_\theta(X_{0:T})$ by $P(X_T)\prod_{t=1}^{T}P_\theta(X_{t-1}|X_t)$ and $q(X_{1:T}|X_0)$ by $\prod_{t=1}^{T}q(X_t|X_{t-1})$ we arrive at the training objective given by Eq. (2.6)

$$\mathbb{E}_q\left[D_{KL}(q(X_T|X_0)||P(X_T)) + \sum_{t=2}^{T}D_{KL}(q(X_{t-1}|X_t,X_0)||P(X_{t-1}|X_t)) - logP_\theta(X_0|X_1)\right] \tag{2.6}$$

In practice Ho et al. [2020] showed that one need not compute all the $T-1$ divergences of the term $\sum_{t=2}^{T}D_{KL}(q(X_{t-1}|X_t,X_0)||P(X_{t-1}|X_t))$ in Eq. (2.6) and can simply take a random subset of them and optimize with gradient descent.

2.4 Log-likelihood-based methods

Strictly speaking, the diffusion model and variational autoencoders can be regarded as likelihood-based approaches to density estimation. However, a key distinction of them as compared to autoregressive, energy-based and flow-based approaches is that VAEs and diffusion models only compute the lower bound of the likelihood rather than the likelihood itself.

2.4.1 Autoregressive models

Given a vector $\mathbf{x} \in \mathcal{R}^D$ expressed by its D components $x_1, x_2 ... x_D$, autoregressive models assume an ordering (often arbitrary) among them. They further make the key assumption that, given values of the previous components $x_0, x_1 ... x_i$, the current component x_{i+1} is independent of the rest. More concretely, autoregressive models make the i-th order Markovian assumption. This enables the factorization given in Eq. (2.7) of the probability density function of a data point

$$P(X) = P(x_0)\prod_{i=1}^{n-1}P(x_i|x_0,...,x_{i-1}). \tag{2.7}$$

As can be readily seen from Eq. (2.7), the terms in the right-hand side can be naturally expressed using a recurrent neural network. This was first observed by Theis and Bethge [2015]. This framework has been further refined and made computationally more efficient by Van Oord et al. [2016] and Salimans et al. [2017]

2.4.2 Normalizing flow models

Normalizing flow models are a class of models that allows one to efficiently compute the exact likelihood of a dataset. By its design, it also yields an exact inference model, and exact sampling mechanism. Given an invertible function, $f : \mathcal{R}^n \to \mathcal{R}^n$, such that $x = f(z)$ and $z = f^{-1}(x)$, we can write the density function of the resultant variable Z given the density, $P(X)$, of the input random variable X as shown in Eq. (2.8)

$$P(X) = P(f^{-1}(X)) * \left| det \frac{\partial f(Z)}{\partial Z} \right|^{-1}. \tag{2.8}$$

Now, if we assume a prior on the derived latent variable Z, we can compute the likelihood of our data set under this prior. Next, by maximizing it, we can learn the parameters of the function f. For a more detailed review on this topic, we refer the reader to the review paper by Kobyzev et al. [2020]. From this perspective, flow-models seem to be a very attractive class of models. However, a close look at Eq. (2.8) revels that to make normalizing flow networks tractable for large datasets, one must come up with a flexible function whose Jacobean and inverse are easy to compute. This puts considerable constraints on the choice of functions.

2.4.3 Energy-based models

As we saw in the earlier sections of this chapter, to arrive at a tractable objective, generative models have to make assumptions that constrain their architecture and or assume a factorized form of the density function they express. These assumptions play out in practice in different ways. Energy-based models (EBMs) are much less restrictive in this regard. They only represent an unnormalized density value for each point and therefore can be represented by any parameterized function called the energy function $f_\theta(x)$. The density function is then given as in Eq. (2.9)

$$P_\theta(X = x) := \frac{exp(-f_\theta(x))}{\int exp(-f_\theta(x))dx}. \tag{2.9}$$

Here θ is the parameter that can be adjusted according to data. We intend to find the maximum likelihood estimate (MLE) of this parameter. The intractability of the normalizing factor $z_\theta = \int exp(-f_\theta(x))dx$ prohibits us from optimizing the data likelihood directly. If we consider gradient-based optimization at this point, one only requires ∇_θ. It can be shown, [Song and Kingma, 2021] that $\nabla_\theta log z_\theta = -\mathbb{E}_{x \sim P_\theta(X)}[\nabla_\theta exp(-f_\theta(x))]$. Furthermore, the term $\nabla_\theta exp(-f_\theta(x))$ is trivial to compute with auto differentiation. Therefore, the only remaining hurdle is to compute the expectation of the gradient under the density represented by the current state of the model. One way to approximate an expectation is via the Monte Carlo method. Specifically, here, we wish to sample from the current model distribution and perform a finite sample approximation of the aforementioned expectation.

Markov Chain Monte Carlo (MCMC) However, obtaining i.i.d. samples from a high dimensional distribution is not trivial. One popular way is to set up a Markov Chain (MC) whose stationary distribution matches the distribution from which we wish to draw samples. Next, we start the chain from an arbitrary state and let it run for a long time (ideally infinite time). Once the chain reaches stationarity, the states generated can be used as i.i.d samples from the desired distribution. However, certain Markov Chains also known as ill-conditioned MCs require a long time to become stationary. Therefore, a lot of work has been put into designing efficient MCs. Despite that, it remains computationally hard, especially for high dimensional data.

Contrastive Divergence (CD) To fix this computational problem, Hinton [2002] propose the Contrastive Divergence mechanism. Here, an MC is started from a real data point (in contrast to an arbitrary starting point) and is run for a fixed number of steps (usually far fewer than the required to obtain stationarity). Intuitively speaking, the hope here is that at some point the model distribution will be close to the data distribution and hence it will be easier to convert the data samples to samples from the model distribution. However, the samples thus generated still suffer in quality, as the truncated MCMC methods produce biased gradients. Scores of corrective mechanisms to this method have been proposed *e.g.*, [Schulz et al., 2010, Fischer and Igel, 2010, Gao et al., 2020], however, the problem of density estimation remains open.

Score Matching (SM) The integral z_θ might be hard to compute, but clearly $\nabla_x log P_\theta(X)$ is easy to compute, since the normalizing constant z_θ does not depend upon x. Therefore, Fisher divergence $D_F(P_{data}(X)||P_\theta(X)) = \mathbb{E}_{P_{data}(X)}[\frac{1}{2}||\nabla_x(\log P_{data}(x) - log P_\theta(x))||^2]$ between the model distribution and the data distribution can be computed, where $\mathbb{E}_{P_{data}(X)}$ represents the expectation operator under the data distribution. The catch, however, is that it involves computation of the derivative of the true data distribution, which does not have an analytic expression. Several methods have been proposed that avoid computing this quantity. By constructing a noisy data distribution [Vincent, 2011], it was shown that one can arrive at an efficiently computable objective. This method is called denoising score matching. Several other variants of score matching objectives exist with varying practical success.

Noise contrastive Estimation (NCE) The basic idea here is that we can learn an energy-based model by contrasting it with another distribution with known density. Say a noise density is $P_n(X)$, the true data density is $P_{data}(X)$ and $P_\theta(X) = exp(-f_\theta(x) - c)$, where c is the normalizing constant. We consider c part of θ *i.e.*, an optimizable parameter and optimize for it. Let us also introduce a random variable Y with Bernoulli distribution. We then sample noise or from data samples according to this decision variable, Y *i.e.*, we sample from noise if a random draw of Y turns out to be 1, else we choose a data sample. Then the density of noise-data mixture will be given by $P_{n,d}(X) = P(Y=1) * P_n(X) + P(Y=0) * P_{data}(x)$. Given this set up, if we wish to classify a given data point while having access to an estimate of the data distribution $P_\theta(X)$, Bayes' rule says that we should compute the conditional distribution given in Eq. (2.10)

$$P_{n,\theta}(Y|X) = \frac{P_{n,\theta}(X|Y) * P(Y)}{P(Y=1) * P_n(X) + P(Y=0) * P_\theta(X)} = \frac{P_{n,\theta}(X|Y)}{P_n(X) + P_\theta(X)} \tag{2.10}$$

The second part of Eq. (2.10) is derived by setting $P(Y=1) = P(Y=0) = 0.5$. Now, since we can sample pairs of $(x \sim \mathcal{X}_{data},\ y \sim P(Y))$ from true data and noise mixture, we can optimize for θ by performing $\text{argmax}_\theta\, \mathbb{E}_{P_{n,d}}\left[log P_{n,\theta}(Y|X)\right]$. Here $\mathcal{X}_{data}$ represents our dataset, the distribution of which we wish to learn. In practice, the success of this training scheme depends upon the chosen noise distribution $P_n(X)$. The closer the noise distribution to the data distribution (but not exactly the same), the better is the quality of the estimation [Gutmann and Hirayama, 2012].

Besides the above-mentioned techniques, several others exist, which revolve around circumventing the computation of the normalizing factor. We refer the reader to [Song and Kingma, 2021] for a more detailed review on energy-based models.

Chapter 3

From Variational to Deterministic Auto-encoder

As described in Sec. 2.2.1, VAEs, let one model data density via a latent variable model. Such a model finds application in various machine learning tasks such as classification, clustering, constrained optimization, object detection, etc. Therefore, to understand and characterize strength and weakness of VAEs is an important task. As with any latent variable model, the prior distribution of the latent variable influences various aspects of the quality of the model. Prior studies suggest that VAEs are no exception, and understanding their behavior regarding the choice of latent distribution is critical to their successful application [Tolstikhin et al., 2017]. Therefore, we study the latent space of a standard VAE and find ways to control it to achieve different desirable properties. We will now take a look at a technique that lets us construct VAEs with any arbitrary prior distribution of the latent variable, instead of the Gaussian one that had traditionally been necessary for tractability.

3.1 Introduction

As is clear from the previous chapter, generative models simulate the data generation process. At least in the sense that both in real life and in latent variable based generative model, a few influencing factors characterize/ describe a high dimensional data point. For example, if we think about an image of a handwritten digit, a few of the influencing factors could be the thickness of the stroke, writing angle, the size of the digit etc., however one can agree that, the number of such parameters are far fewer than the number of pixels with which we represent the image itself. Furthermore, if we randomly assign values to pixels in an image, we hardly expect to get a picture of an object. This motivates us to hypothesize that every image is a member of a lower dimensional subspace of the pixel space in which they are commonly expressed. This low dimensional space also is sometimes loosely referred to as the data manifold. Having access to such a manifold would let one reason about data probabilistically, access and traverse the data manifold, and ultimately generate new data. It is therefore not surprising that generative models have gained momentum in applications such as computer vision [Sohn et al., 2015, Brock et al., 2018], natural language processing (NLP) [Bowman et al., 2016, Severyn et al., 2017], and chemistry [Kusner et al., 2017, Jin et al., 2018, Gómez-Bombarelli et al., 2018].

VAEs [Kingma and Welling, 2014, Rezende et al., 2014] cast learning representations for high-dimensional distributions as a variational inference problem. Learning a VAE amounts to the optimization of an objective balancing the quality of samples that are autoencoded through a stochastic encoder–decoder pair while encouraging the latent space to follow a fixed prior distribution. Since their introduction, VAEs have become one of the frameworks of choice among the different generative models. VAEs promise theoretically well-founded and more stable training than Generative Adversarial Networks (GANs) [Goodfellow et al., 2014] and more efficient sampling mechanisms than autoregressive models [Larochelle and Murray, 2011, Germain et al., 2015].

However, the VAE framework is still far from delivering the promised generative mechanism, as there are several practical and theoretical challenges yet to be solved. Specifically, it has been observed that VAEs tend to deliver blurry[1] results (samples and reconstruction). This, coupled with the fact that there are no blurry samples in the training set, indicate that the learned model gives weight to data points that are clearly out of the training set, which in turn signals partial failure in the applied method. It has been attributed to two key open problems—1. the mismatch between aggregate posterior and prior 2. a very simplistic likelihood model. The recently published VD-VAE [Child, 2021] and VQ-VAE [van den Oord et al., 2017] claim to solve the blur problem with a very deep architecture and vector quantization technique, respectively. However, a closer inspection reveals that, VQ-VAE reduces the dimensionality of input data only be 1/4X. On the other hand, VD-VAE has high-resolution skip connections. This limits its usability as a representation learning framework, as the latent code is higher dimensional than the original data representation. Given that GANs find a way to represent high-resolution input data at a much higher compression ratio, we conclude that the two problems stated above remain unsolved. In this chapter, we explore a solution to the posterior–prior mismatch problem. VAEs tend to strike an unsatisfying compromise between sample quality and reconstruction quality. We study this effect in a standard VAE in depth in Sec. 3.3 and Sec. 3.2. In practice, this has been attributed to overly simplistic prior distributions [Tomczak and Welling, 2018, Dai and Wipf, 2019] or, alternatively, to the inherent over-regularization induced by the KL divergence term in the VAE objective [Tolstikhin et al., 2017]. Most importantly, the VAE objective itself poses several challenges as it admits trivial solutions that decouple the latent space from the input [Chen et al., 2017b, Zhao et al., 2017], leading to the posterior collapse phenomenon with powerful decoders [van den Oord et al., 2017]. Furthermore, due to its variational formulation, training a VAE requires approximating expectations through sampling at the cost of increased variance in gradients [Burda et al., 2016, Tucker et al., 2017], making initialization, validation, and annealing of hyperparameters essential in practice [Bowman et al., 2016, Higgins et al., 2017, Bauer and Mnih, 2019]. Lastly, even after a satisfactory convergence of the objective, the learned aggregated posterior distribution rarely matches the assumed latent prior in practice [Kingma et al., 2016, Bauer and Mnih, 2019, Dai and Wipf, 2019], ultimately hurting the quality of generated samples.

All in all, much of the attention around VAEs is still directed towards 'fixing' the aforementioned drawbacks associated with them. In this work, we take a different route: we question whether the variational framework adopted by VAEs is necessary for generative modelling and, in particular, to obtain a smooth latent space. We propose to adopt a simpler, deterministic version of VAEs that scales better, is simpler to optimize, and, most importantly, still produces a meaningful latent space and equivalently good or better samples than VAEs or stronger alternatives, *e.g.*, Wasserstein Autoencoders (WAEs) [Tolstikhin et al., 2017]. We do so by observing

[1]Blurry usually means samples that are the result of an averaging operation between points from different modes of the data. Depending upon the data and the averaging process, this can lead to completely unrealistic generation, *e.g.*, in case of discrete data such as in NLP tasks VAEs might generate non-meaningful samples.

that, under commonly used distributional assumptions, training a stochastic encoder–decoder pair in VAEs does not differ from training a deterministic architecture where noise is added to the decoder's input. We investigate how to substitute this noise injection mechanism with other regularization schemes in the proposed deterministic *Regularized Autoencoders* (RAEs), and we thoroughly analyse how this affects performance. Finally, we equip RAEs with a generative mechanism via a simple ex-post density estimation step on the learned latent space.

In summary, our contributions in this capter are as follows: i) we introduce the RAE framework for generative modelling as a drop-in replacement for many common VAE architectures; ii) we propose an ex-post density estimation scheme which greatly improves sample quality for VAEs, WAEs and RAEs without the need to retrain the models; iii) we conduct a rigorous empirical evaluation to compare RAEs with VAEs and several baselines on standard image datasets and on more challenging structured domains such as molecule generation [Kusner et al., 2017, Gómez-Bombarelli et al., 2018].

With this introduction, let us dive straight into the method. Since having studied the method in the following subsections will help us contrast our work to the related work more naturally, we make an exception here and present the related work section afterwards.

3.2 Practice and shortcomings of VAEs

Recall from Chapter 2, Sec. 2.2.1 that our data set is a collection of high-dimensional i.i.d samples $\mathcal{X} := \{x_i\}_{i=1}^N$ drawn from the true data distribution $P_{data}(X)$ over a random variable X. The aim of a classical VAE is to learn from $\mathcal{X}$ a mechanism to draw new samples $x_{new} \sim P_{data}$. As is usually for latent variable models, the generative process of the VAE is defined as

$$z_{new} \sim P(Z), \qquad x_{new} \sim P_\theta(X|Z = z_{new}) \tag{3.1}$$

Recall also from Chapter 2, Sec. 2.2.1 that $P(Z)$ is a fixed prior distribution over a low-dimensional latent space Z. Recall also that we characterized the likelihood distribution $P_\theta(X|Z)$, with a deterministic function D_θ. Using autoencoder terminology, we call this function D_θ a decoder. The decoding process

$$D_\theta(Z) := x \sim P_\theta(X|Z) = x \sim P(X|D_\theta(z)) \tag{3.2}$$

links the latent space to the input space through the *likelihood* distribution P_θ, where D_θ is an expressive non-linear function parameterized by θ. Recall further from Chapter 2, Sec. 2.2.1 that we needed a $Q_\phi(Z|X)$ distribution to achieve sample efficiency. This distribution is characterized by another function approximator E_ϕ. This maps the input space to the latent space and is also called an encoder. The encoding process is as follows.

$$E_\phi(X) := z \sim Q_\phi(Z|X) = z \sim Q(E_\phi(x)) \tag{3.3}$$

where $Q_\phi(Z|X)$ is the *posterior* distribution given by a function E_ϕ parameterized by ϕ. To train this model, one follows a variational approach and maximizes the evidence lower bound (ELBO). We rewrite Eq. (2.2) as follows.

$$logP_\theta(X) \geq \mathsf{ELBO}(\phi, \theta, X) = \mathbb{E}_{z\sim q_\phi(Z|X)} logP_\theta(X|Z) - \mathbb{KL}\left(Q_\phi(Z|X)||P(Z)\right) \tag{3.4}$$

Now the optimization problem can be written compactly as in Eq. (3.4) over data $\mathcal{X}$ w.r.t. model parameters ϕ, θ.

$$\underset{\phi,\theta}{\text{argmin}}\ \mathbb{E}_{x\sim P_{data}}\ \mathcal{L}_{\mathsf{ELBO}} = \underset{\phi,\theta}{\text{argmin}}\ \mathbb{E}_{x\sim P_{data}}\ \mathcal{L}_{\mathsf{REC}} + \mathcal{L}_{\mathsf{KL}} \tag{3.5}$$

where $\mathcal{L}_{\mathsf{REC}}$ and $\mathcal{L}_{\mathsf{KL}}$ are defined for a sample x as follows:

$$\mathcal{L}_{\mathsf{REC}} = -\mathbb{E}_{z\sim q_{\phi}(Z|X)} log P_{\theta}(X|Z) \qquad \mathcal{L}_{\mathsf{KL}} = \mathbb{KL}(Q_{\phi}(Z|X)||P(Z)) \tag{3.6}$$

Intuitively, the reconstruction loss $\mathcal{L}_{\mathsf{REC}}$ takes into account the quality of autoencoded samples x_i through $D_{\phi}(E_{\theta}(x))$, while the KL-divergence term $\mathcal{L}_{\mathsf{KL}}$ encourages $Q_{\phi}(Z|X)$ to match the prior $P(Z)$ for each, z_i which acts as a regularizer during training [Hoffman and Johnson, 2016].

To fit a VAE to data through Eq. (3.5) one has to specify the parametric forms for the prior, $P(Z)$, posterior, $Q_{\phi}(Z|X)$, and likelihood $P_{\theta}(X|Z)$, and hence the deterministic mappings E_{ϕ} and D_{θ}. In practice, the choice for the above distributions is guided by trading off computational complexity with model expressiveness. In the most commonly adopted formulation of the VAE, the posterior and the likelihood distributions are assumed to be Gaussian:

$$E_{\phi}(X) \sim \mathcal{N}(Z|\mu_{\phi}(X),\ \mathsf{diag}(\sigma_{\phi}(X))) \qquad D_{\theta}(E_{\phi}(X)) \sim \mathcal{N}(X|\mu_{\theta}(Z),\ \mathsf{diag}(\sigma_{\theta}(Z))) \tag{3.7}$$

with means μ_{ϕ}, μ_{θ} and variance parameters $\sigma_{\phi}, \sigma_{\theta}$ are outputs of E_{ϕ} and D_{θ}. In practice, the covariance of the decoder is set to the identity matrix for all z, *i.e.*, $\sigma_{\theta}(Z) = 1$ [Dai and Wipf, 2019]. The expectation of $\mathcal{L}_{\mathsf{KL}}$ in Eq. (3.6) must be approximated via k Monte Carlo point estimates. It is expected that the quality of the Monte Carlo estimate, and hence convergence during learning and sample quality, increases for larger k [Burda et al., 2016]. However, only a 1-sample approximation is generally carried out [Kingma and Welling, 2014] since memory and time requirements are prohibitive for large k. With the 1-sample approximation, $\mathcal{L}_{\mathsf{REC}}$ is computed as the mean squared error between input samples and their mean reconstructions μ_{θ} by a decoder that is deterministic in practice:

$$\mathcal{L}_{\mathsf{REC}} = ||X - \mu_{\theta}(E_{\phi}(X))||_2^2 \tag{3.8}$$

Gradients w.r.t. the encoder parameters ϕ are computed through the expectation of $\mathcal{L}_{\mathsf{REC}}$ in Eq. (3.6) via the reparametrization trick [Kingma and Welling, 2014] where the stochasticity of E_{θ} is relegated to an auxiliary random variable ε that does not depend on ϕ:

$$E(X) = \mu_{\phi}(X) + \sigma_{\phi}(X) \odot \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I) \tag{3.9}$$

where $\odot$ denotes the Hadamard product. An additional simplifying assumption involves fixing the prior $P(Z)$ to be a d-dimensional isotropic Gaussian $\mathcal{N}(Z|0,I)$. For this choice, the KL-divergence for a sample x_i is given in closed form: $2\mathcal{L}_{\mathsf{KL}} = ||\mu_{\phi}(x_i)||_2^2 + d + \sum_i^d \sigma_{\phi}(x_i) - log\sigma_{\phi}(x_i)$.

While the above assumptions make VAEs easy to implement, the stochasticity in the encoder and decoder are still problematic in practice [Makhzani et al., 2016, Tolstikhin et al., 2017, Dai and Wipf, 2019].

In particular, one has to balance the trade-off carefully between the $\mathcal{L}_{\mathsf{KL}}$ term and $\mathcal{L}_{\mathsf{REC}}$ during optimization [Dai and Wipf, 2019, Bauer and Mnih, 2019]. A too-large weight on the $\mathcal{L}_{\mathsf{KL}}$ term can make it dominate $\mathcal{L}_{\mathsf{ELBO}}$, having the effect of *over-regularization*. As this would smooth the latent space, it can directly affect sample quality negatively. Heuristics to avoid this includes manually fine-tuning or gradually annealing the importance of $\mathcal{L}_{\mathsf{KL}}$ during training [Bowman et al., 2016, Bauer and Mnih, 2019]. We also observe this trade-off in a practical experiment, as shown in below in sec 3.3.

3.3 Reconstruction and regularization trade-off

We train a VAE on MNIST while monitoring the test set reconstruction quality by FID. Figure 3.1 (left) clearly shows the impact of more expensive $k > 1$ Monte Carlo approximations of Eq. (3.7) on sample quality during training. The commonly used 1-sample approximation is a clear limitation for VAE training.

Figure 3.1 (right) depicts the inherent trade-off between reconstruction and random sample quality in VAEs. Enforcing structure and smoothness in the latent space of a VAE affects random sample quality in a negative way. In practice, a compromise needs to be made, ultimately leading to subpar performance.

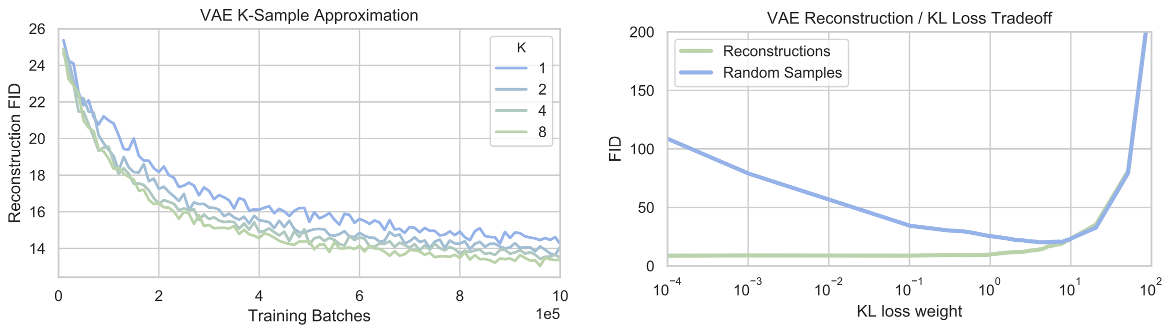


Figure 3.1: (Left) Test reconstruction quality for a VAE trained on MNIST with different numbers of samples in the latent space as in Eq. (3.7) measured by FID (lower is better). Larger numbers of Monte-Carlo samples clearly improve training, however, the increased accuracy comes with larger requirements for memory and computation. In practice, the most common choice is, therefore $k = 1$. (Right) Reconstruction and random sample quality (FID, y-axis, lower is better) of a VAE on MNIST for different trade-offs between $\mathcal{L}_{\mathsf{REC}}$ and $\mathcal{L}_{\mathsf{KL}}$ (x-axis, see Eq. (3.5)). Higher weights for $\mathcal{L}_{\mathsf{KL}}$ improve random samples, but hurt reconstruction. This is especially noticeable towards the optimality point ($\beta \approx 10^1$). This indicates that enforcing structure in the VAE latent space leads to a penalty in quality.

Even after employing the full array of approximations and “tricks” to reach convergence of Eq. (3.5) for a satisfactory set of parameters, there is no guarantee that the learned latent space is distributed according to the assumed prior distribution. In other words, the aggregated posterior distribution $Q_\phi(Z) = \mathbb{E}_{x\sim P_{data}} Q_\phi(Z|x)$ has been shown not to conform well to $P(Z)$ after training [Tolstikhin et al., 2017, Bauer and Mnih, 2019, Dai and Wipf, 2019]. This critical issue severely hinders the generative mechanism of VAEs (cf. Eq. (3.1)) since latent codes sampled from $P(Z)$ (instead of $Q_\phi(Z)$) might lead to sampling from the regions of the latent space that are previously unseen to the decoder during training. This results in generation of out-of-distribution samples. We refer the reader to Appendix A.1.7 for a visual demonstration of this phenomenon on the latent space of VAEs. We analyze solutions to this problem in Sec. 3.6.

3.4 Constant-Variance Encoders

Before introducing our fully-deterministic take on VAEs, it is worth investigating intermediate flavors of VAEs with reduced stochasticity. Analogous to what is commonly done for decoders as discussed in the previous section, one can fix the variance of $Q_\phi(Z|X)$ to be constant for all x_i. This simplifies the computation of E from Eq. (3.9) to

$$E^{CV}(X) = \mu_\phi(X) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma I) \tag{3.10}$$

where σ is a fixed scalar. Then, the KL loss term in a Gaussian VAE simplifies (up to a constant) to $\mathcal{L}_{\mathsf{KL}}^{\mathsf{CV}} = ||\mu_\phi(X)||_2^2$. We name this variant Constant-Variance VAEs (CV-VAEs). While CV-VAEs have been adopted in some applications such as variational image compression [Ballé et al., 2017] and adversarial robustness [Ghosh et al., 2019], to the best of our knowledge, there is no systematic study of them in the literature. We will fill this gap in our experiments in Sec. 3.8. Lastly, note that now σ in Eq.(3.10) is not learned along the encoder as in Eq. (3.9). Nevertheless, it can still be fitted as an hyperparameter, *e.g.*, by cross-validation, to maximise the model likelihood.

We provide a more flexible solution to deal with the mismatch between the prior $P(Z)$ and posterior $Q_\phi(Z|X)$ distributions over Z via ex-post density estimation in Sec. 3.6.

3.5 Deterministic Regularized Autoencoders

Autoencoding in VAEs is defined in a probabilistic fashion: E_ϕ and D_θ map data points not to a single point, but rather to parameterized distributions (cf. Eq. (3.7)). However, common implementations of VAEs as discussed in Sec. 3.2 admit a simpler, deterministic view of this probabilistic mechanism. A glance at the autoencoding mechanism of the VAE is revealing.

The encoder deterministically maps a data point X to mean $\mu_\phi(X)$ and variance $\sigma_\phi(X)$ in the latent space. The input to the decoder is then simply the mean $\mu_\phi(X)$ *augmented with Gaussian noise* scaled by $\sigma_\phi(X)$ via the reparametrization trick (cf. Eq. (3.9)). In the CV-VAE, this relationship is even more obvious, as the magnitude of the noise is fixed for all data points (cf. Eq. (3.10)). In this light, a VAE can be seen as a *deterministic* autoencoder where (Gaussian) noise is added to the decoder's input.

We argue that this noise injection mechanism is a key factor in having a regularized decoder. Using random noise injection to regularize neural networks is a well-known technique that dates back several decades [Sietsma and Dow, 1991, Guozhong, 1996]. It implicitly helps to smooth the function learned by the network at the price of increased variance in the gradients during training. In turn, decoder regularization is a key component in generalization for VAEs, as it improves random sample quality and achieves a smoother latent space. Indeed, from a generative perspective, regularization is motivated by the goal to learn a smooth latent space where similar data points are mapped to similar latent codes, and small variations in the latent code lead to reconstructions by the decoder that vary only slightly.

We propose to *substitute noise injection with an explicit regularization scheme for the decoder.* This entails the substitution of the variational framework in VAEs, which enforces regularization on the encoder posterior through $\mathcal{L}_{\mathsf{KL}}$, with a deterministic framework that applies other flavors of decoder regularization. By removing noise injection from a CV-VAE, we are effectively left with a deterministic autoencoder (AE). Coupled with explicit regularization for the decoder, we obtain a *Regularized Autoencoder* (RAE). Training a RAE thus involves minimizing the simplified loss

$$\mathcal{L}_{\mathsf{RAE}} = \mathcal{L}_{\mathsf{REC}} + \beta \mathcal{L}^{\mathsf{RAE}}_{\mathbf{Z}} + \lambda \mathcal{L}_{\mathsf{REG}} \tag{3.11}$$

where $\mathcal{L}_{\mathsf{REG}}$ represents the explicit regularizer for the discriminator D_θ (discussed in Sec. 3.5.1) and $\mathcal{L}^{\mathsf{RAE}}_{\mathbf{Z}} = 1/2||Z||_2^2$ (resulting from simplifying $\mathcal{L}^{\mathsf{CV}}_{\mathsf{KL}}$) is equivalent to constraining the size of the learned latent space, which is still needed to prevent unbounded optimization. Finally, β and λ are two hyperparameters that balance the different loss terms. We arrive at the RAE loss function (Eq. (3.11)) from a probabilistic perspective in Appendix A.1.1.

Note that for RAEs, no Monte Carlo approximation is required to compute $\mathcal{L}_{\mathsf{REC}}$. This reliefs the need for more samples from $Q_\phi(Z|X)$ to achieve better image quality (cf. Sec. 3.3). Moreover, by abandoning the variational framework and the $\mathcal{L}_{\mathsf{KL}}$ term, there is no need in RAEs for a fixed prior distribution over Z. Doing so, however, loses a clear generative mechanism for RAEs to sample from Z. We propose a method to regain random sampling ability in Sec. 3.6 by performing density estimation on Z ex-post. This step that is any way needed in regular VAEs to alleviate the posterior mismatch issue.

3.5.1 Regularization Schemes for RAEs

Among possible choices for $\mathcal{L}_{\mathsf{REG}}$, a first obvious candidate is Tikhonov regularization [Tikhonov and Arsenin, 1977], since it is known to be related to the addition of low-magnitude input noise [Bishop, 2006]. Training a RAE within this framework thus amounts to adopting, $\mathcal{L}_{\mathsf{REG}} = \mathcal{L}_{\mathsf{L}_2} = ||\theta||_2^2$ which effectively applies weight decay on the decoder parameters θ.

Another option comes from the recent GAN literature where regularization is a hot topic [Kurach et al., 2018] and where injecting noise to the input of the adversarial discriminator has led to improved performance in a technique called *instance noise* [Sønderby et al., 2017]. To enforce Lipschitz continuity on adversarial discriminators, weight clipping has been proposed [Arjovsky et al., 2017], which is, however, known to significantly slow down training. More successfully, a *gradient penalty* on the discriminator can be used similar to [Gulrajani et al., 2017, Mescheder et al., 2018], yielding the objective $\mathcal{L}_{\mathsf{REG}} = \mathcal{L}_{\mathsf{GP}} = ||\nabla D_\theta(E_\phi(X))||_2^2$ which bounds the gradient norm of the decoder w.r.t. its input.

Additionally, spectral normalization (SN) has been successfully proposed as an alternative way to bound the Lipschitz norm of an adversarial discriminator [Miyato et al., 2018]. SN normalizes each weight matrix θ_ℓ in the decoder by an estimate of its largest singular value: $\theta^{\mathsf{SN}}_\ell = \theta_\ell / \mathsf{s}(\theta_\ell)$ where $\mathsf{s}(\theta_\ell)$ is the current estimate obtained through the power method.

In light of the recent successes of deep networks *without* explicit regularization [Zagoruyko and Komodakis, 2016, Zhang et al., 2017a], it is intriguing to question the need for explicit regularization of the decoder to obtain a meaningful latent space. The assumption here is that techniques such as dropout [Srivastava et al., 2014], batch normalization [Ioffe and Szegedy, 2015], adding noise during training [Guozhong, 1996] implicitly regularize the networks enough.

Therefore, as a natural baseline to the $\mathcal{L}_{\text{RAE}}$ objectives introduced above, we also consider the RAE framework without $\mathcal{L}_{\text{REG}}$ and $\mathcal{L}_{\mathbf{Z}}^{\text{RAE}}$, *i.e.*, a standard deterministic autoencoder optimizing $\mathcal{L}_{\text{REC}}$ only.

To complete our 'autopsy' of the VAE loss, we additionally investigate deterministic autoencoders with decoder regularization, but without the $\mathcal{L}_{\mathbf{Z}}^{\text{RAE}}$ term, as well as possible combinations of different regularizers in our RAE framework (cf. Table A.3 in Appendix A.1.8).

Lastly, it is worth questioning if it is possible to formally derive our RAE framework from first principles. We answer this affirmatively, and show how to augment the ELBO optimization problem of a VAE with an explicit constraint, while not fixing a parametric form for $Q_\phi(Z|X)$. This indeed leads to a special case of the RAE loss in Eq. (3.11). Specifically, we derive a regularizer like $\mathcal{L}_{\text{GP}}$ for a deterministic version of the CV-VAE. Note that this derivation legitimates bounding the decoder's gradients and as such it justifies the spectral norm regularizer as well since the latter enforces the decoder's Lipschitzness. We provide the full derivation in Appendix A.1.1.

3.6 Ex-Post Density Estimation

By removing stochasticity and ultimately, the KL divergence term $\mathcal{L}_{\text{KL}}$ from RAEs, we have simplified the original VAE objective at the cost of detaching the encoder from the prior $P(Z)$ over the latent space. This implies that i) we cannot ensure that the latent space Z is distributed according to a simple distribution (*e.g.*, isotropic Gaussian) anymore and consequently, ii) we lose the simple mechanism provided by $P(Z)$ to sample from Z as in Eq. (3.1).

As discussed in Sec. 3.2, issue i) is a drawback that is present in the VAE framework in any case, as reported in several works [Hoffman and Johnson, 2016, Rosca et al., 2018, Dai and Wipf, 2019]. To fix this, some works extend the VAE objective by encouraging the aggregated posterior to match $P(Z)$ [Tolstikhin et al., 2017] or by utilizing more complex priors [Kingma et al., 2016, Tomczak and Welling, 2018, Bauer and Mnih, 2019].

To overcome both i) and ii), we instead propose to employ *ex-post density estimation* over Z. We fit a density estimator denoted as $q_\delta(Z)$ to $\{z = E_\phi(x) | x \in \mathcal{X}\}$ by choosing an objective suitable for the density estimation method *e.g.*, by maximizing log-likelihood. This simple approach not only fits our RAE framework well, but it can also be readily adopted for any VAE or variants thereof such as the WAE [Tolstikhin et al., 2017] as a practical remedy to the aggregated posterior mismatch without adding any computational overhead to the costly training phase.

The choice of $q_\delta(Z)$ needs to trade-off *expressiveness* – to provide a good fit of an arbitrary space for Z – with *simplicity*, to improve generalization. For example, placing a Dirac distribution on each latent point z would allow the decoder to output only training sample reconstructions which have a high quality, but dos not generalize. Striving for simplicity, we employ and compare a full covariance multivariate Gaussian with a 10-component Gaussian mixture model (GMM) in our experiments and determine its parameters by maximize log-likelihood of the latent codes of the training set.

3.7 Related works

Many works have focused on diagnosing the VAE framework, the terms in its objective [Hoffman and Johnson, 2016, Zhao et al., 2017, Alemi et al., 2018], and ultimately augmenting it to solve optimization issues [Rezende and Viola, 2018, Dai and Wipf, 2019]. With RAE, we argue that a simpler deterministic framework can be competitive for generative modeling.

Deterministic denoising [Vincent et al., 2008] and contractive autoencoders (CAEs) [Rifai et al., 2011] have received attention in the past for their ability to capture a smooth data manifold. Heuristic attempts to equip them with a generative mechanism include MCMC schemes [Rifai et al., 2012, Bengio et al., 2013b]. However, they are hard to diagnose for convergence, require a considerable effort in tuning [Cowles and Carlin, 1996], and have not scaled beyond MNIST [LeCun et al., 1998b], leading to them being superseded by VAEs. While computing the Jacobian for CAEs [Rifai et al., 2011] is close in spirit to $\mathcal{L}_{\mathsf{GP}}$ for RAEs, the latter is much more computationally efficient.

Approaches to cope with the aggregated posterior mismatch involve fixing a more expressive form for $P(Z)$ [Kingma et al., 2016, Bauer and Mnih, 2019], therefore, altering the VAE objective and requiring considerable additional computational effort. Estimating the latent space of a VAE with a second VAE [Dai and Wipf, 2019] reintroduces many of the optimization shortcomings discussed for VAEs and is much more expensive in practice compared to fitting a simple $q_\delta(Z)$ after training.

Adversarial Autoencoders (AAE) [Makhzani et al., 2016] add a discriminator to a deterministic encoder–decoder pair, leading to sharper samples at the expense of higher computational overhead and the introduction of instabilities caused by the adversarial nature of the training process.

Wasserstein Autoencoders (WAE) [Tolstikhin et al., 2017] have been introduced as a generalization of AAEs by casting autoencoding as an optimal transport (OT) problem. Both stochastic and deterministic models can be trained by minimizing a relaxed OT cost function employing either an adversarial loss term or the maximum mean discrepancy score between $P(Z)$ and $q_\phi(Z)$ as a regularizer in place of $\mathcal{L}_{\mathsf{KL}}$. Within the RAE framework, we look at this problem from a different perspective: instead of explicitly imposing a simple structure on the latent space that might impair the ability to fit high-dimensional data during training, we propose to model the latent space by an ex-post density estimation step.

The most successful VAE architectures for images and audio so far are variations of the VQ-VAE [van den Oord et al., 2017, Razavi et al., 2019]. Despite the name, VQ-VAEs are neither stochastic, nor variational, but they are deterministic autoencoders. VQ-VAEs are similar to RAEs in that they adopt ex-post density estimation. However, VQ-VAEs necessitate complex discrete autoregressive density estimators and a training loss that is non-differentiable due to quantizition of the latent code.

Lastly, RAEs share some similarities with GLO [Bojanowski et al., 2018]. However, differently from RAEs, GLO can be interpreted as a deterministic AE without and encoder, and when the latent space is built "on-demand" by optimization. On the other hand, RAEs augment deterministic decoders as in GANs with deterministic encoders.

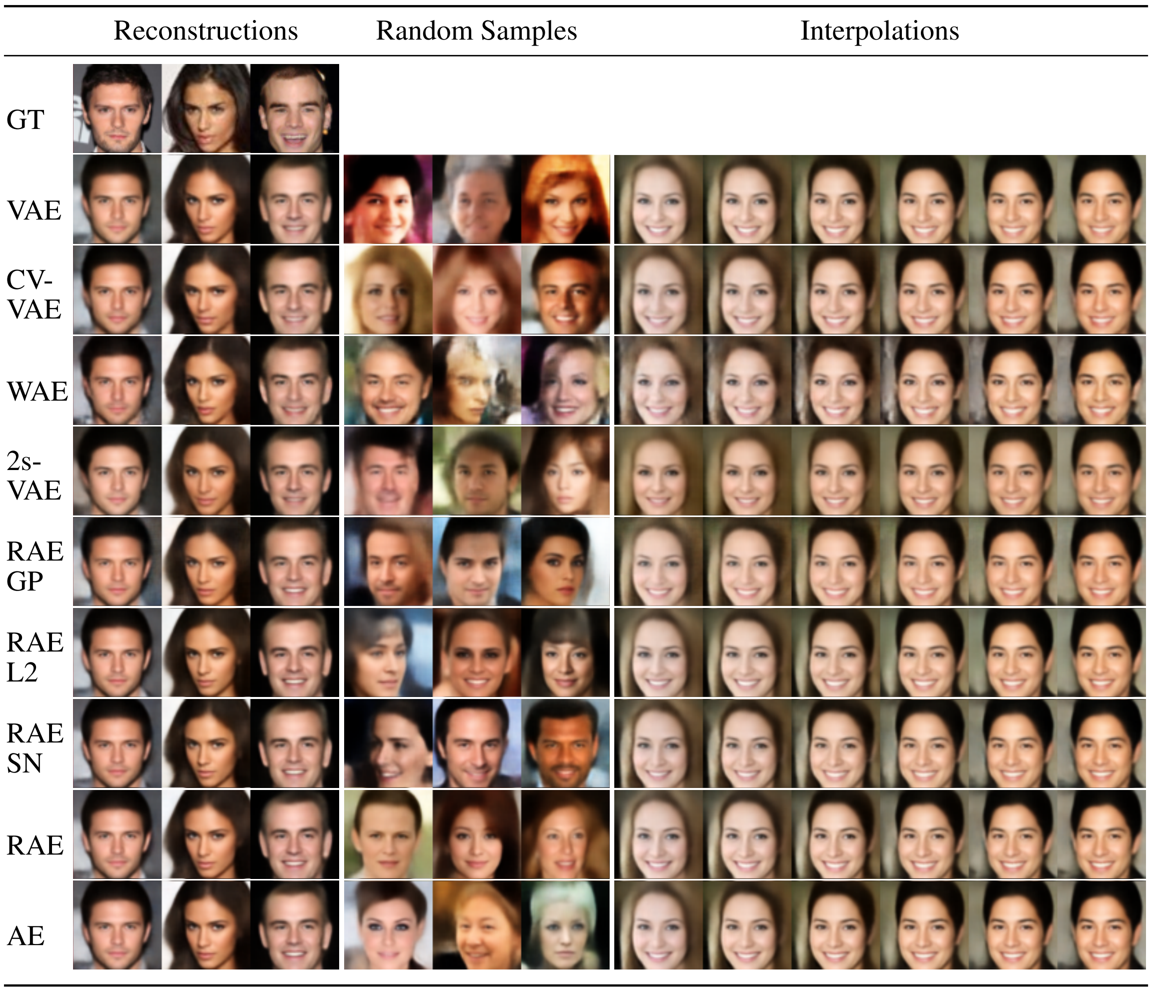


Figure 3.2: Qualitative evaluation of sample quality for VAEs, WAEs, 2sVAEs, and RAEs on CelebA. RAE provides slightly sharper samples and reconstructions while interpolating smoothly in the latent space. Corresponding qualitative overviews for MNIST and CIFAR-10 are provided in Appendix A.1.5 in the interest of completeness. GT here represents groundtruth.

3.8 Experiments

Our experiments are designed to answer the following questions: **Q1:** Are sample quality and latent space structure in RAEs comparable to VAEs? **Q2:** How do different regularizations impact RAE performance? **Q3:** What is the effect of ex-post density estimation on VAEs and its variants?

3.8.1 RAEs for image modeling

We evaluate all regularization schemes from Sec. 3.5.1: RAE-GP (RAE with gradient penalty), RAE-L2 (RAE with weight decay), and RAE-SN (RAE with spectral normalization). For a thorough ablation study, we also consider only adding the latent code regularizer $\mathcal{L}_{\mathbf{z}}^{\mathsf{RAE}}$ to $\mathcal{L}_{\mathsf{REC}}$ (RAE), and an autoencoder without any explicit regularization (AE). We check the effect

	MNIST				CIFAR				CelebA			
	Rec.	Samples			Rec.	Samples			Rec.	Samples		
		$\mathcal{N}$	GMM	Interp.		$\mathcal{N}$	GMM	Interp.		$\mathcal{N}$	GMM	Interp.
VAE	18.26	19.21	17.66	18.21	57.94	106.37	103.78	88.62	39.12	48.12	45.52	44.49
CV-VAE	15.15	33.79	17.87	25.12	37.74	94.75	86.64	69.71	40.41	48.87	49.30	44.96
WAE	**10.03**	20.42	9.39	**14.34**	35.97	117.44	93.53	76.89	**34.81**	53.67	42.73	40.93
2sVAE	20.31	**18.81**	–	18.35	62.54	109.77	–	89.06	42.04	49.70	–	47.54
RAE-GP	14.04	22.21	11.54	15.32	32.17	83.05	76.33	64.08	39.71	116.30	45.63	47.00
RAE-L2	10.53	22.22	**8.69**	14.54	32.24	**80.80**	**74.16**	62.54	43.52	51.13	47.97	45.98
RAE-SN	15.65	19.67	11.74	15.15	**27.61**	84.25	75.30	63.62	36.01	**44.74**	**40.95**	**39.53**
RAE	11.67	23.92	9.81	14.67	29.05	83.87	76.28	63.27	40.18	48.20	44.68	43.67
AE	12.95	58.73	10.66	17.12	30.52	84.74	76.47	**61.57**	40.79	127.85	45.10	50.94
AE-L2	11.19	315.15	9.36	17.15	34.35	247.48	75.40	61.09	44.72	346.29	48.42	56.16

Table 3.1: Evaluation of all models by FID (lower is better, best models in bold). We evaluate each model by REC.: test sample reconstruction; $\mathcal{N}$: random samples generated according to the prior distribution $P(Z)$ (isotropic Gaussian for VAE / WAE, another VAE for 2SVAE) or by fitting a Gaussian to $q_\delta(Z)$ (for the remaining models); GMM: random samples generated by fitting a mixture of 10 Gaussians in the latent space; Interp.: mid-point interpolation between random pairs of test reconstructions. The RAE models are competitive with or outperform previous models throughout the evaluation. Interestingly, interpolations do not suffer from the lack of explicit priors on the latent space in our models.

of applying one regularization scheme while not including the $\mathcal{L}_{\mathbf{Z}}^{\text{RAE}}$ term in the AE-L2 model.

As baselines, we employ the regular VAE, constant-variance VAE (CV-VAE), Wasserstein Autoencoder (WAE) with the maximum mean discrepancy (MMD) loss as a state-of-the-art method, and the recent 2-stage VAE (2sVAE) [Dai and Wipf, 2019] which performs a form of ex-post density estimation via another VAE. For a fair comparison, we use the same network architecture for all models. We omit the details about the architecture and training here, as they are orthogonal to the topic at hand and are given in Appendix A.1.2 for completeness.

We measure the following quantities: held-out sample reconstruction quality, random sample quality, and interpolation quality. While reconstructions give us a lower bound on the best quality achievable by the generative model, random sample quality indicates how well the model generalizes. Finally, interpolation quality sheds light on the structure of the learned latent space. The evaluation of generative models is a nontrivial research question [Theis et al., 2016, Sajjadi et al., 2017, Lucic et al., 2018]. We report here the ubiquitous Fréchet Inception Distance (FID) [Heusel et al., 2017b]. Since precision and recall scores (PRD) [Sajjadi et al., 2018][2] correlates strongly to FID metric, we provide them in Appendix A.1.4 in the interest of readability.

Table 3.1 summarizes our main results. All of the proposed RAE variants are competitive with the VAE, WAE and 2sVAE w.r.t. generated image quality in all settings. Sampling RAEs achieve the best FIDs across all datasets when a modest 10-component GMM is employed for ex-post density estimation. Furthermore, even when $\mathcal{N}$ is considered as $q_\delta(Z)$, RAEs rank first with the exception of MNIST, where it competes for the second position with a VAE. Our best RAE FIDs are lower than the best results reported for VAEs in the large-scale comparison of [Lucic et al., 2018], challenging even the best scores reported for GANs. While we

[2]PRD provides independent view on precision and recall.

are employing a slightly different architecture than theirs, our models underwent only modest fine-tuning instead of an extensive hyperparameter search. A comparison of the different regularization schemes for RAEs (Q2) yields no clear winner across all settings, as all perform equally well. Striving for a simpler implementation, one may prefer RAE-L2 over the GP and SN variants.

For completeness, we investigate applying multiple regularization schemes at the same time to our RAE models. We report the results of all possible combinations in Table A.3, Appendix A.1.8. There, no significant boost of performance can be spotted when comparing to singly regularized RAEs.

Surprisingly, the implicitly regularized RAE and AE models are shown to be able to score impressive FIDs when $q_\delta(Z)$ is fit through GMMs. FIDs for AEs decrease from 58.73 to 10.66 on MNIST and from 127.85 to 45.10 on CelebA – a value close to the state of the art. This is a remarkable result that follows a long series of recent confirmations that neural networks are surprisingly smooth by design [Neyshabur et al., 2017]. It is also surprising that the lack of an explicitly fixed structure on the latent space of the RAE does not impede interpolation quality. This is further confirmed by the qualitative evaluation on CelebA as reported in Fig. 3.2 and for the other datasets in Appendix A.1.5, where RAE interpolated samples seem sharper than competitors and transitions smoother.

Our results further confirm and quantify the effect of the aggregated posterior mismatch. In Table 3.1, ex-post density estimation consistently improves sample quality across all settings and models. A 10-component GMM halves FID scores from $\sim$20 to $\sim$10 for WAE and RAE models on MNIST and from 116 to 46 on CelebA. This is especially striking since this additional step is much cheaper and simpler than training a second-stage VAE as in 2sVAE (Q3). In summary, the results strongly support the conjecture that the simple deterministic RAE framework can challenge VAEs and stronger alternatives (Q1).

3.8.2 GrammarRAE: modeling structured inputs

We now evaluate RAEs for generating complex structured objects such as molecules and arithmetic expressions. We do this with a twofold aim: i) to investigate the latent space learned by RAE for more challenging input spaces that abide to some structural constraints, and ii) to quantify the gain of replacing the VAE in a state-of-the-art generative model with a RAE.

To this end, we adopt the exact architectures and experimental settings of the GrammarVAE (GVAE) [Kusner et al., 2017], which has been shown to outperform other generative alternatives such as the CharacterVAE (CVAE) [Gómez-Bombarelli et al., 2018]. As in Kusner et al. [2017], we are interested in traversing the latent space learned by our models to generate samples (molecules or expressions) that best fit some downstream metric. This is done by Bayesian optimization (BO) by considering the $\log(1+\mathsf{MSE})$ (lower is better) for the generated expressions w.r.t. some ground truth points, and the water-octanol partition coefficient ($\log P$) [Pyzer-Knapp et al., 2015] (higher is better) in the case of molecules. A well-behaved latent space will not only generate molecules or expressions with better scores during the BO step, but it will also contain syntactically valid ones, *i.e.*,, samples abide to a grammar of rules describing the problem.

Figure 3.3 summarizes our results over 5 trials of BO. Our GRAEs (Grammar RAE) achieve better average scores than CVAEs and GVAEs in generating expressions and molecules. This is visible also for the three best samples and their scores for all models, with the exception

Problem	Model	% Valid	Avg. score
Expressions	GRAE	**100± 0.00**	3.22 ± 0.03
	GCVVAE	99.0 ± 1.00	**2.85 ± 0.08**
	GVAE	99.0 ± 1.00	3.26 ± 0.20
	CVAE	82.0 ± 7.00	4.74 ± 0.25
Molecules	GRAE	72.0 ± 9.00	**-5.62 ± 0.71**
	GCVVAE	**76.0 ± 6.00**	-6.40 ± 0.80
	GVAE	28.0 ± 4.00	-7.89 ± 1.90
	CVAE	16.0 ± 4.00	-25.64 ± 6.35

Model	#	Expression	Score
GRAE	1	$\sin(3)+x$	0.39
	2	$x+1/\exp(1)$	**0.39**
	3	$x+1+2*\sin(3+1+2)$	**0.43**
GCVVAE	1	$x+\sin(3)*1$	0.39
	2	$x/x/3+x$	0.40
	3	$\sin(\exp(\exp(1)))+x/2*2$	**0.43**
GVAE	1	$x/1+\sin(x)+\sin(x*x)$	**0.10**
	2	$1/2+(x)+\sin(x*x)$	0.46
	3	$x/2+\sin(1)+(x/2)$	0.52
CVAE	1	$x*1+\sin(x)+\sin(3+x)$	0.45
	2	$x/1+\sin(1)+\sin(2*2)$	0.48
	3	$1/1+(x)+\sin(1/2)$	0.61

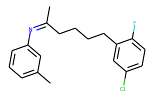
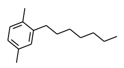
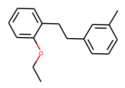
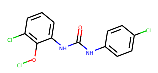
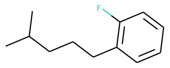
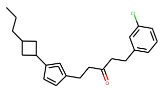
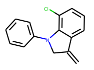
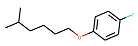
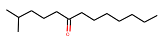
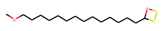
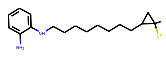
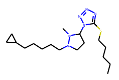
Model	1st	2nd	3rd
GRAE			
Score	**3.74**	**3.52**	**3.14**
GCVVAE			
Score	3.22	2.83	2.63
GVAE			
Score	3.13	3.10	2.37
CVAE			
Score	2.75	0.82	0.63

Figure 3.3: Generating structured objects by GVAE, CVAE and GRAE. (Upper left) Percentage of valid samples and their average mean score (see text, Sec. 3.8.2). The three best expressions (lower left) and molecules (upper right) and their scores are reported for all models.

of the first best expression of GVAE. We include in the comparison also the GCVVAE, the equivalent of a CV-VAE for structured objects, as an additional baseline. We can observe that while the GCVVAE delivers better average scores for the simpler task of generating equations (even though the single three best equations are on par with GRAE), when generating molecules GRAEs deliver samples associated to much higher scores.

More interestingly, while GRAEs are almost equivalent to GVAEs for the easier task of generating expressions, the proportion of syntactically valid molecules for GRAEs greatly improves over GVAEs (from 28% to 72%). This we attribute to the ex-post-density estimation introduced in Sec. 3.6, that significantly reduces mismatch between prior and aggregate posterior.

3.9 Conclusion

While the theoretical derivation of the VAE has helped popularize the framework for generative modeling, recent works have started to expose some discrepancies between theory and practice. We have shown that viewing sampling in VAEs as noise injection to enforce smoothness can enable one to distill a deterministic autoencoding framework that is compatible with several regularization techniques to learn a meaningful latent space. We have demonstrated that such an autoencoding framework can generate comparable or better samples than VAEs while getting around the practical drawbacks tied to a stochastic framework. This result sheds light on the importance of the two main phenomenon identified to be causing the blur problem in VAEs.

Namely, 1. posterior-prior miss match, 2. pixel space L_2 loss. Since RAEs by design do not have posterior-prior miss match and yet produce comparable results as traditional VAEs, we conclude that the weak likelihood model, *i.e.*, pixel space L_2 loss is the main contributor towards VAEs' characteristic blurry reconstruction and samples. We would like to highlight that we have shown that our solution of fitting a simple density estimator on the learned latent space consistently improves sample quality for the proposed RAE framework as well as for VAEs, WAEs, and 2sVAEs which solves the mismatch between the prior and the aggregated posterior in VAEs.

Chapter 4

Gaussian mixture as Priors of VAEs

Although theoretically VAEs as introduced by [Kingma and Welling, 2014] can be constructed with an arbitrary parametric choice of the posterior $Q_\phi(Z|X)$and prior $P(Z)$, practical tractability of $\mathbb{KL}\left(Q_\phi(Z|X)||P(Z)\right)$ mandates that we use Gaussian priors and posteriors. However, as seen in the previous chapter in Sec. 3.1 viewing VAEs from a different perspective helps relax this requirement. In this chapter, we take a look at a special case of VAEs with a Gaussian mixture prior and leverage them to gain adversarial robustness during classification.

4.1 Resisting adversarial attacks using Gaussian mixture variational autoencoders

The vulnerability of deep neural networks to adversarial attacks has generated a lot of interest and concern in the past few years. The fact that these networks can be easily fooled by adding specially crafted noise to the input, such that the original and modified inputs are indistinguishable to humans [Szegedy et al., 2014], clearly suggests that they fail to mimic the human learning process. Even though these networks achieve state-of-the-art performance, often surpassing human level performance [He et al., 2015, Huang et al., 2017] on the test data used for different tasks, their vulnerability is a cause of concern when deploying them in real life applications, especially in domains such as health care [Finlayson et al., 2018], autonomous vehicles [Eykholt et al., 2018] and defense, etc.

4.2 Adversarial Attacks and Defenses

Adversarially crafted samples can be classified into two broad categories, namely (i) adversarial samples [Szegedy et al., 2014] and (ii) fooling samples as defined by [Nguyen et al., 2015]. Loosely speaking, adversarial samples are data points which to a human look clearly to belong to one class, but an otherwise high-performance classifier assigns a different class label. On the other hand, fooling samples are data points which look like random noise to a human, but an otherwise high-performance classifier assign them a class with high confidence. Existence of adversarial samples was first shown by Szegedy et al. Szegedy et al. [2014], while fooling samples [Nguyen et al., 2015], which are closely related to the idea of "rubbish class" images [LeCun et al., 1998a] were introduced by Nguyen et al. [Nguyen et al., 2015]. Evolutionary algorithms were applied to inputs drawn from a uniform distribution, using the predicted probability corresponding to the targeted class as the fitness function [Nguyen et al., 2015] to craft such fooling samples. It has also been shown that Gaussian noise can be directly used to trick classifiers into predicting one of the output classes with very high probability [Goodfellow et al., 2015a].

Adversarial attack methods can be classified into (i) white box attacks [Szegedy et al., 2014, Goodfellow et al., 2015a, Carlini and Wagner, 2017b, Papernot et al., 2016c, Moosavi Dezfooli et al., 2016, Chen et al., 2018, Madry et al., 2018], which use knowledge of the machine learning model (such as model architecture, loss function used during training, etc.) for crafting adversarial samples, and (ii) black box attacks [Papernot et al., 2017, Liu et al., 2017, Papernot et al., 2016b, Chen et al., 2017a], which only require the model for obtaining labels corresponding to input samples. Both kinds of attacks can be further split into two sub categories, (i) targeted attacks, which trick the model into producing a chosen output, and (ii) non-targeted attacks, which cause the model to produce any undesired output [Goodfellow et al., 2015a]. The majority of attacks and defenses have dealt with adversarial samples so far [Szegedy et al., 2014, Gu and Rigazio, 2014, Papernot et al., 2016d], while a relatively smaller literature deals with fooling samples [Nguyen et al., 2015]. However, to the best of our knowledge, no prior method tries to defend against both kinds of samples simultaneously under a unified framework. State-of-the-art defense mechanisms have tried to harden a classifier by one or more of the following techniques: adversarial retraining [Szegedy et al., 2014], preprocessing inputs [Gu and Rigazio, 2014], deploying auxiliary detection networks [Meng and Chen, 2017] or obfuscating gradients [Athalye et al., 2018b]. One common drawback of these defense mechanisms is that they do not eliminate the vulnerability of deep networks altogether, but only try to defend against previously proposed attack methods. Hence, they have been easily broken by stronger attacks, which are specifically designed to overcome their defense strategies [Carlini and Wagner, 2016, Athalye et al., 2018b].

Szegedy et al. [2014] argue that the primary reason for the existence of adversarial samples is the presence of small “pockets” in the data manifold, which are rarely sampled in the training or test set. On the other hand, Goodfellow et al. [2015a] have proposed the “linearity hypothesis” to explain the presence of adversarial samples. We design a classifier such that the adversarial objective, namely small input perturbation, should produce large output change, entails a contradiction. We elaborate this in Sec. 4.6.

4.3 Approach

We design a generative model that finds a latent random variable Z such that the data label Y and the data X become conditionally independent given, Z i.e. $P(X,Y|Z) = P(X|Z) * P(Y|Z)$. We base our generative model on VAEs [Kingma and Welling, 2014], and obtain an inference model that represents $P(Z|X)$ and a generative model that represents $P(X|Z)$. We perform label inference $P(Y|X)$ by computing $P(Y|Z) * P(X|Z)$. We choose the latent space distribution $P(Z)$ to be a mixture of Gaussians, such that each mixture component represents one of the classes in the data. Under this construct, inferring the label given latent encoding, i.e., $P(Y|Z)$ becomes trivial by computing the contribution of the mixture components. Adversarial samples are dealt with by thresholding in the latent and output spaces of the generative model and rejecting the inputs for which $P(X) \approx 0$. In Figure 4.1 we describe our network at test and train time.

In summary, our contributions in this section are as follows: i) We show how VAE’s can be trained with labeled data, using a Gaussian mixture prior on the latent variable in order to perform classification. ii) We perform selective classification using this framework, thereby rejecting adversarial and fooling samples. iii) We propose a method to learn a classifier in a semi-supervised scenario using the same framework, and show that this classifier is also resistant against adversarial attacks. iv) We also show how the detected adversarial samples can be reclassified into the correct class by iterative optimization. v) We verify our claims through experimentation on 3 publicly available datasets: MNIST [LeCun et al., 1998a], SVHN [Netzer et al., 2011] and COIL-100 [Nayar et al., 1996].

4.4 Related Works

A few pieces of work in the existing literature on defense against adversarial attacks have attempted to use generative models in different ways.

Pouya Samangouei [2018] propose training a Generative Adversarial Network (GAN) on the training data of a classifier, and use this network to project every test sample on to the data manifold by iterative optimization. This method does not try to detect adversarial samples, and does not tackle “fooling images”. Further, this defense technique has been recently shown to be ineffective [Athalye et al., 2018b]. Other pieces of work have also shown that adversarial samples can lie on the output manifold of generative models trained on the training data for a classifier [Zhao et al., 2018].

PixelDefend, proposed by Song et al. [2018a] also uses a generative model to detect adversarial samples, and then rectifies the classifier output by projecting the adversarial input back to the data manifold. However, Athalye et al. [2018a] have shown that this method can also be broken by bypassing the exploding/vanishing gradient problem introduced by the defense mechanism.

MagNet [Meng and Chen, 2017] uses autoencoders to detect adversarial inputs, and is similar to our detection mechanism in the way reconstruction threshold is used for detecting adversarial inputs. This defense method does not claim security in the white box setting. Further, the technique has also been broken in the grey box setting by recently proposed attack methods [Carlini and Wagner, 2017a].

Traditional autoencoders do not constrain the latent representation to have a specific distribution like variational autoencoders. Our use of variational autoencoders allows us to defend against adversarial and fooling inputs simultaneously, by using thresholds in the latent and output spaces of the model in conjunction. This makes the method secure to white box attacks as well, which is not the case with MagNet.

Further, even state-of-the-art defense mechanisms [Madry et al., 2018] and certified defenses have been shown to be ineffective for simple datasets such as MNIST [Song et al., 2018b]. We show via extensive experimentation on different datasets how our method is able to defend against strong adversarial attacks, as well as end to end white box attacks.

4.5 GMM-VAE

We consider the dataset $\mathcal{X} = \{x^{(i)}\}_{i=1}^{N}$ consisting of N i.i.d. samples of a random variable X. Let Z be the latent representation from which the data is assumed to have been generated. Similar to Kingma and Welling [2014], we assume that the data generation process consists of two steps: (i) a value $z^{(i)}$ is sampled from a prior distribution $P_{\theta^*}(Z)$; (ii) a value $x^{(i)}$ is generated from a conditional distribution $P_{\theta^*}(X|Z)$. We also assume that the prior $P_{\theta^*}(Z)$ and likelihood $P_{\theta^*}(X|Z)$ come from parametric families of distributions $P_\theta(Z)$ and, $P_\theta(X|Z)$ respectively. In order to maximize the data likelihood $P_\theta(X)$, VAEs [Kingma and Welling, 2014] use an encoder network $Q_\phi(Z|X)$, that approximates $P_\theta(Z|X)$. VAEs maximize the data likelihood by maximizing the so called ELBO as given in Eq. (3.4). Using a Gaussian prior $P(Z)$ and a Gaussian posterior $Q_\phi(Z|X)$, makes it particularly convenient since a closed form expression for their KL-divergence is available.

4.5.1 Modifying the Evidence Lower Bound

VAEs do not enforce any lower or upper bound on encoder entropy $H(Q_\phi(Z|X))$. This can result in blurry reconstruction due to sample averaging in case of overlap in the latent space. On the other hand, unbounded decrease in $H(Q_\phi(Z|X))$ is not desirable either, as in that case, the VAE can degenerate to a deterministic autoencoder leading to holes in the latent space[1]. Hence, we seek an alternative design in which we fix this quantity to a constant value. In order to do so, we express the KL divergence in terms of entropy.

$$\begin{aligned}\mathbb{KL}(Q_\phi(Z|X)\,\|\,P(Z)) &= -\mathbb{E}_{z\sim Q_\phi(Z|X)}(\log P(Z) - \log Q_\phi(Z|X))\\ &= -\mathbb{E}_{z\sim Q_\phi(Z|X)}\log P(Z) + \mathbb{E}_{z\sim Q_\phi(Z|X)}\log Q_\phi(Z|X)\\ &= H(Q_\phi(Z|X),\, P(Z)) - H(Q_\phi(Z|X))\end{aligned} \tag{4.1}$$

where $H(Q_\phi(Z|X), P(Z))$ represents the cross entropy between $Q_\phi(Z|X)$ and $P(Z)$. It can be noted that we need to minimize the KL divergence term. Hence, if we assume that $H(Q_\phi(Z|X))$ is constant, then we can drop this term during optimization (please refer to the next section for details of how $H(Q_\phi(Z|X))$ is enforced to be constant). This lets us replace the KL divergence in a VAE's loss function with $H(Q_\phi(Z|X), P(Z))$. Leading to the following simplified evidence lower bound.

$$\begin{aligned}\mathsf{ELBO}(X,\theta,\phi) &= \mathbb{E}_{z\sim Q_\phi(Z|X)}\log P_\theta(X|Z) - H(Q_\phi(Z|X), P(Z))\\ &= \mathbb{E}_{z\sim Q_\phi(Z|X)}\log P_\theta(X|Z) + \mathbb{E}_{z\sim Q_\phi(Z|X)}\log P(Z)\end{aligned} \tag{4.2}$$

The choice of fixing the entropy of $Q_\phi(Z|X)$ is further justified via experiments in the experiments section. We would like to draw the reader's attention here to how this expression of the ELBO contrasts with the one presented in Eq.(3.4). This is a direct consequence of the assumption that $H(Q_\phi(Z|X))$ is constant.

4.5.2 Supervision using a Gaussian Mixture Prior

In this section, we modify the above evidence lower bound (ELBO) term for supervised learning by including the random variable Y denoting labels. The following expression can be derived for the log-likelihood of the data. Note that this new bound of likelihood is evidence lower bound (ELBO), and hence it looks notationally different from Eq. (4.2) however it must be

[1]We like to inform the reader at this point that RAEs as introduced in Chap 3 were not conceived, while this work was carried out. In fact, this effort directly influenced development of RAEs

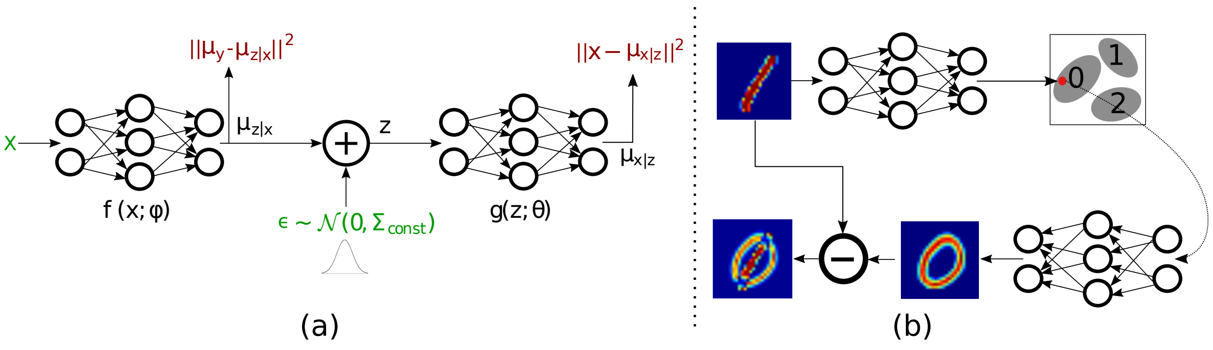

$||\mu_y - \mu_{z|x}||^2$
$||x - \mu_{x|z}||^2$
x
$\mu_{z|x}$
z
$\mu_{x|z}$
f (x; φ)
g(z; θ)
$\epsilon \sim \mathcal{N}(0, \Sigma_{const})$
0
1
2
(a)
(b)

Figure 4.1: (a) The model at training time. All the inputs are in green, while all the losses are in brown. (b) Model pipeline at inference time. The red dot shows that the attacker is successful in fooling the encoder by placing its output in the wrong class. However, it results in a high reconstruction error, since the decoder generates an image of the target class.

understood as a new bound to likelihood that is closely related.

$$\begin{aligned}\log(P_\theta(X,Y)) &= \mathbb{E}_{z\sim Q_\phi(Z|X)}\log(P_\theta(X,Y|Z)) \\ &-\mathbb{KL}(Q_\phi(Z|X)||P(Z)) + \mathbb{KL}(Q_\phi(Z|X)||P_\theta(Z|X,Y))\end{aligned} \tag{4.3}$$

Noting that $\mathbb{KL}(Q_\phi(Z|X)||P(Z)) \geq 0$, and replacing $D_{KL}[Q_\phi(Z|X)||P(Z)]$ with $\mathbb{E}_{z\sim Q_\phi(Z|X)}\log P(Z)$ by assuming $H(Q_\phi(Z|X))$ to be constant (as shown in Eq. (4.2)), we get the following lower bound on the data likelihood.

$$\text{ELBO}(X,Y,\theta,\phi) = \mathbb{E}_{z\sim Q_\phi(Z|X)}\log(P_\theta(X,Y|Z)) + \mathbb{E}_{z\sim Q_\phi(Z|X))}\log P(Z) \tag{4.4}$$

We choose our VAE to use a Gaussian mixture prior for the latent variable Z. We further choose the number of mixture components to be equal to the number of classes k in the training data. The means of each of these components, $\mu_1, \mu_2, ..., \mu_k$ are chosen to be the one-hot encodings of the class labels in the latent space. It can be noted here that although this choice enforces the latent dimensionality to be k, it can be easily altered by choosing the means in a different manner. For example, means of all the mixture components can lie on a single axis in the latent space. Unlike usual VAEs, our encoder network outputs only the mean of the posterior $Q_\phi(Z|X)$. We use the reparameterization trick introduced by Kingma and Welling [2014], but sample the ε (necessary for reparameterization) from $N(0, \Sigma_{constant})$ in order to enforce the entropy of $Q_\phi(Z|X)$ to be constant. Here, each mixture component corresponds to one class and X is assumed to be generated from the latent space according to $P_\theta(X|Z)$ irrespective of Y. Therefore, X and Y become conditionally independent given Z, i.e. $\log(P_\theta(X,Y|Z)) = \log(P_\theta(X|Z)) + \log(P_\theta(Y|Z))$.

$$\begin{aligned}\text{ELBO}(X,Y,\theta,\phi) &= \mathbb{E}_{z\sim Q_\phi(Z|X)}[\log(P_\theta(X|Z)) + \log(P_\theta(Y|Z)) + \log P(Z)] \\ &= \mathbb{E}_{z\sim Q_\phi(Z|X)}[\log(P_\theta(X|Z)) + \log(P_\theta(Z|Y)) + \log P_\theta(Y)]\end{aligned} \tag{4.5}$$

Assuming the classes to be equally likely, the final loss function for an input $x^{(i)}$ with label $y^{(i)}$ becomes the following.

$$\mathcal{L}(x^{(i)},y^{(i)},\varepsilon) = ||x^{(i)} - g(f(x^{(i)})+\varepsilon)||^2 + \alpha||f(x^{(i)}) - \mu_{y^{(i)}}||^2 \tag{4.6}$$

where the encoder is represented by f, the decoder is represented by g and $\mu_{y^{(i)}}$ represents the mean of the mixture component corresponding to $y^{(i)}$. α is a hyper-parameter that trades off between reconstruction fidelity, latent space prior and classification accuracy.

The label $y(i)$ for an input sample $x(i)$ can be obtained following the Bayes Decision rule.

$$\begin{aligned}
&\operatorname*{argmax}_y P_\theta(Y|X) = \operatorname*{argmax}_y P_\theta(X,Y) = \operatorname*{argmax}_y \int_{\mathcal{Z}} P_\theta(X,Y|Z)dz \\
&= \operatorname*{argmax}_y \int_{\mathcal{Z}} P_\theta(X|Z)P_\theta(Y|Z)dz = \operatorname*{argmax}_y \int_{\mathcal{Z}} P_\theta(Z|X)P_\theta(Y|Z)P_\theta(X)dz \\
&= \operatorname*{argmax}_y \int_{\mathcal{Z}} P_\theta(Z|X)P_\theta(Y|Z)dz
\end{aligned} \tag{4.7}$$

$P_\theta(Z|X)$ can be approximated by $Q_\phi(Z|X)$, i.e., the encoder distribution. This corresponds to the Bayes decision rule, in the scenario where there is no overlap among the classes in the input space, ϕ has enough variability and $Q_{\phi^*}(Z|X)$ is able to match $P_{\theta^*}(Z|X)$ exactly.

Semi-supervised learning follows automatically, by using the loss function in Eq. (4.6) for labeled samples, and the loss corresponding to Eq. (4.2) for unlabeled samples. Experiments regarding this is shown in table 4.1.

4.6 Resisting adversarial attacks

In order to successfully reject adversarial samples irrespective of the method of its generation, we use thresholding at the encoder and decoder outputs. This allows us to reject any sample $x(i)$ whose encoding $z(i)$ has low probability under $P_\theta(Z)$, i.e., if the distance between its encoding and the encoding of the predicted class label in the latent space exceeds a threshold value, τ_{enc} (since $P_\theta(Z)$ is a mixture of Gaussians). We further reject those input samples that have low probability under $P_\theta(X|Z)$, i.e., if the reconstruction error exceeds a certain threshold, τ_{dec} (since $P_\theta(X|Z)$ is Gaussian). Essentially, a combination of these two thresholds ensures that $P_\theta(X) = \int_{\mathcal{Z}} P_\theta(X|Z)P_\theta(Z)dz$ is not low.

Both τ_{enc} and τ_{dec} can be determined based on statistics obtained while training the model. In our experiments, we implement thresholding in the latent space as follows: we calculate the Mahalanobis distance between the encoding of the input and the encoding of the corresponding mixture component mean, and reject the sample if it exceeds the critical chi-square value (3σ rule in the univariate case). Similarly, for τ_{dec}, we use the corresponding value for the reconstructions errors. However, in general, any value can be assigned to these two thresholds, and they determine the risk-to-coverage trade-off for this selective classifier.

If the maximum allowed L_p norm of the perturbation η is γ, then the adversary, trying to modify an input $x^{(i)}$ from class c_1, must satisfy the following criteria.

1. $\operatorname{argmin}_{c_i} ||f(x^{(i)}+\eta) - \mu_{c_i}||_2 = c_2$ where $c_2 \neq c_1$
2. $||f(x^{(i)}+\eta) - \mu_{c_2}||_2 \leq \tau_{enc}$
3. $||\eta||_p \leq \gamma$

4. $||x^{(i)} - g(f(x^{(i)} + \eta) + \varepsilon)||_2 \leq \tau_{dec}$ where $\varepsilon \sim N(0, \Sigma_{constant})$

By the first three constraints, the encoding of $x^{(i)}$ and $x^{(i)} + \eta$ must belong to different Gaussian mixture components in the latent space. However, constraint (4) requires the distance between the reconstructions of these two samples from two different mixture components to be less than τ_{dec}, which is extremely hard to satisfy, because of low probability of occurrence of holes in the latent space within τ_{enc} distance from the means.

Similarly, for the case of fooling samples, it can be argued that even if an attacker manages to generate a fooling sample which tricks the encoder, it will be very hard to simultaneously trick the decoder to reconstruct a similar image belonging to the rubbish class.

4.7 Reclassification

Once a sample is detected as adversarial by either or both the thresholds discussed above, we attempt to find its true label using our generative model. By the definition of adversarial images $x^{(i)}_{adv} = x^{(i)}_{org} + \eta$, where $||\eta||_p$ is small. Hence, we can conclude that $||x^{(i)} - x^{(i)}_{adv}||_2 \approx ||x^{(i)} - x^{(i)}_{org}||_2$. Suppose z^* is given by Eq. (4.8)

$$z^* = \underset{z}{\operatorname{argmin}} ||g(z) - x_{org}||_2^2 \tag{4.8}$$

Following the argument stated above, we can approximate $z^* \approx z^*_{adv} = \operatorname{argmin}_z ||g(z) - x_{adv}||^2$. We can now find the label of the adversarial sample as $\operatorname{argmin}_{c_i} ||\mu_{c_i} - z^*_{adv}||^2$. Essentially, for reclassification, we try to find the z in the latent space, which, when decoded, gives the minimum reconstruction error from the adversarial input. However, if Eq. (4.8) returns a z that lies beyond τ_{enc} from the corresponding mean, or if the reconstruction error exceeds τ_{dec}, we conclude that the sample is a fooling sample and reject the sample. It can be noted here that if this network is deployed in a scenario where fooling samples are not expected to be encountered, one can choose not to reject samples during reclassification, thereby increasing coverage. Also, starting from a single value of z can cause the optimization process to get stuck at a local minimum. A better alternative is to run k different optimization processes with $z = \mu_1, \mu_2, \ldots, \mu_k$ as the initial values, and choose the z which gives minimum reconstruction error as z^*_{adv}. Given enough compute power, these k processes can be run in parallel. In our experiments, we follow these two strategies while reclassifying adversarial samples.

4.8 Experiments

We verify the effectiveness of our network through numerical results and visual analysis on three different datasets—MNIST [LeCun et al., 1998b], SVHN [Netzer et al., 2011] and COIL-100 [Nene et al., 1996]. For different datasets, we make minimal changes to the hyperparameters of our network, partly due to the difference in the image size and image type (grayscale/-color) in each dataset.

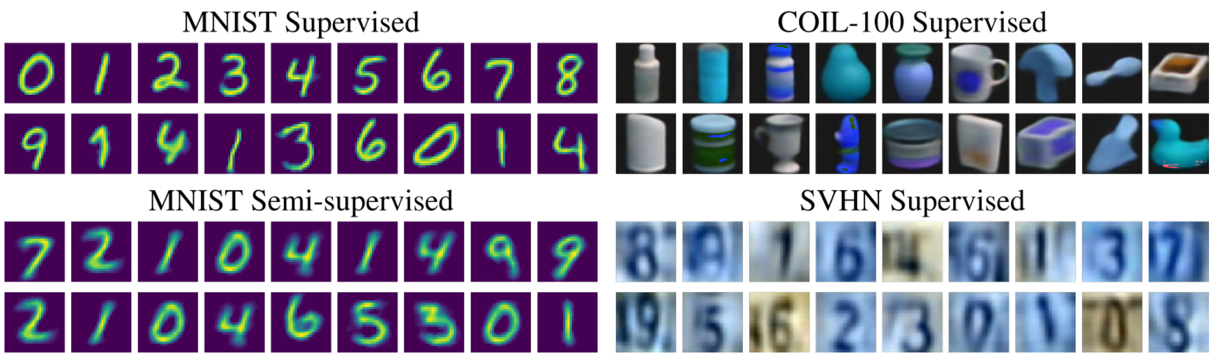


Figure 4.2: Generated images from different classes of MNIST, COIL-100, SVHN.

	Supervised					Semi-supervised			
	Wo. Th.		Th.			Wo. Th.	Th.		
Dataset	SOTA	Acc.	Acc.	Error	Rejec.	Acc.	Acc.	Error	Rejec.
MNIST	99.79%	99.67%	97.97%	0.22%	1.81%	99.1%	98.17%	0.52%	1.31%
SVHN	98.31%	95.06%	92.80%	4.58%	2.62%	86.42%	83.54%	13.64%	2.82%
COIL	99.11%	99.89%	98.40%	0%	1.60%	-	-	-	-

Table 4.1: Comparison between the performance of the state-of-the-art (SOTA) models and our model. We show that our method, even without much fine-tuning focused on achieving classification accuracy, is competitive with the SOTA. MNIST SOTA is as reported by Wan et al. [2013], SVHN SOTA is as given by Lee et al. [2016] and the SOTA for COIL-100 is given by Wu et al. [2015]. Wh. Th. here represents without threshold and Th. represents with threshold.

4.8.1 Implementation details

We use an encoder network with convolution, max-pooling and dense layers to parameterize $Q_\phi(Z|X)$, and a decoder network with convolution, up-sampling and dense layers to parameterize $P_\theta(X|Z)$. We choose the dimensionality of the latent space to be the same as the number of classes for MNIST and COIL-100. However, noting that the size of images is larger for SVHN compared to MNIST, and also because the dataset contains color images, we choose the dimensionality of the latent space for SVHN as 32 instead of 10. The choice of means also varies slightly for this dataset, as we pad zeros to the one-hot encodings of the class labels to allow for the extra latent dimensions. The standard deviation of the encoder distribution is chosen such that the chance of overlap of the mixture components in the latent space is negligible, and the classes are well separated. We use $1/3000$ as the variance for the MNIST dataset, and reduce this value as the latent dimensionality increases for the other datasets. We use the ReLU nonlinearity in our network, and sigmoid activation in the final layer so that the output lies in the allowed range $[0,1]$. We use the Adam by Diederik P. Kingma [2015] optimizer for training.

4.8.2 Qualitative evaluation.

Since our algorithm relies upon the reconstruction error between the generated and the original samples, we first show a few randomly chosen images generated by the network (for both supervised ad semi-supervised scenarios) corresponding to test samples of different classes from the three datasets in Figure 4.2.

4.8.3 Numerical results.

In Table 4.1, we present the accuracy, error and rejection percentages obtained by our method with and without thresholding. For semi-supervised learning, we have taken 100 randomly chosen labeled samples from each class for both MNIST and SVHN during training. It is important to note here that the SOTA for COIL-100 was obtained on a random train-test split of the dataset, and hence, the accuracy values are not directly comparable.

4.8.4 Adversarial attacks on encoder.

We use the encoder part of the network trained on the MNIST dataset to generate adversarial samples using the *Fast Gradient Sign Method (FGSM)* with varying ε values [Goodfellow et al., 2015a]. The corresponding results are shown in Figure 4.3. The behavior is as desired, i.e., with increasing ε, the percentage of misclassified samples rises to a maximum value of only 3.89% and then decreases, while the accuracy decreases monotonically and the rejection percentage increases monotonically. Similar results are obtained for the semi-supervised model, as shown in Figure 4.3, although the maximum error rate is higher in this case. We further tried the FGSM attack from the Cleverhans library [Papernot et al., 2016a] with the default parameters on the SVHN and COIL-100 datasets, and all the generated samples were rejected by the models after thresholding. Similarly, we generated adversarial samples for all three datasets using stronger attacks from Cleverhans with default parameter settings, including the Momentum Iterative Method [Dong et al., 2018] and Projected Gradient Descent [Madry et al., 2018]. In these cases as well, all generated adversarial samples were successfully rejected by thresholding.

This indicates that since all these attacks lack knowledge of the decoder network, they only manage to produce samples that fool the encoder network, but are easily detected at the decoder output. From this set of experiments, we conclude that the only effective method of attacking our model would be to design a complete white-box attack that has knowledge of the decoder loss, as well as the two thresholds. Further, since we do not use any form of gradient obfuscation in our defense mechanism, a complete white-box attacker would represent a strong adversary.

4.8.5 White-box adversarial attack.

We present the results for completely white-box targeted attack on our model for the COIL-100 and MNIST datasets in figures 4.4a and 4.4b. Here, the adversary has complete knowledge of the encoder, the decoder, as well as the rejection thresholds. The results shown correspond to random samples from the first two classes of objects for the COIL-100 dataset, and the classes 2 and 5 for MNIST dataset. We perform gradient descent on the adversarial objective as given

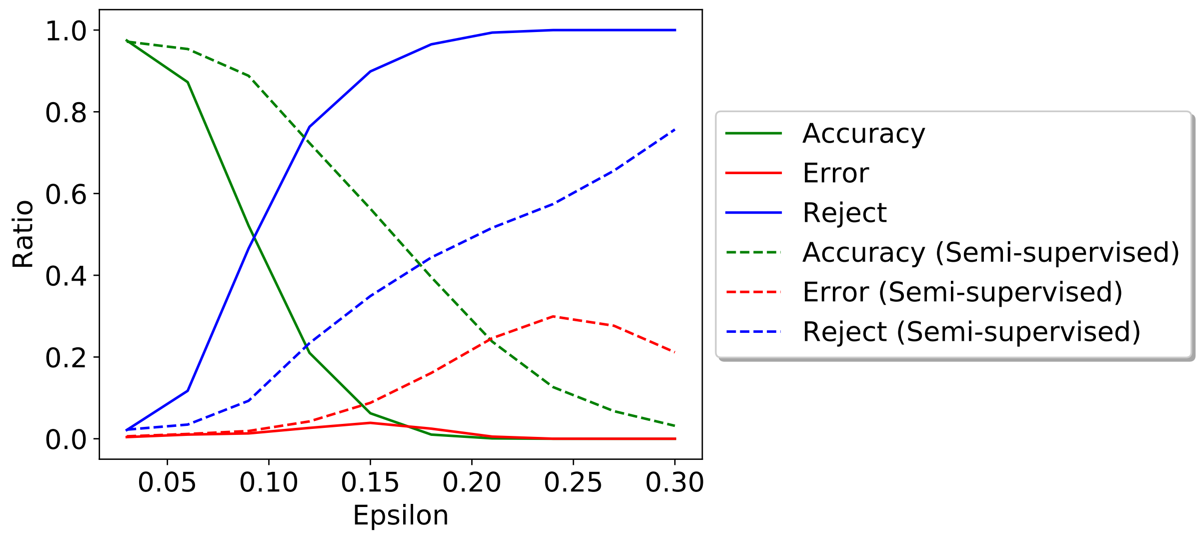


Figure 4.3: We run FGSM with varying ε on the models trained on MNIST data in both supervised and semi-supervised scenarios. Although the error rate is higher for the semi-supervised network, the rejection ratio rises monotonically for both networks with increasing ε, and the error rate for the supervised model stays below 5%.

in Eq. (4.9). The target class is set to 6 for MNIST images from class 2, 9 for MNIST images from class 5, and the class other than that of the source image for the COIL-100 images.

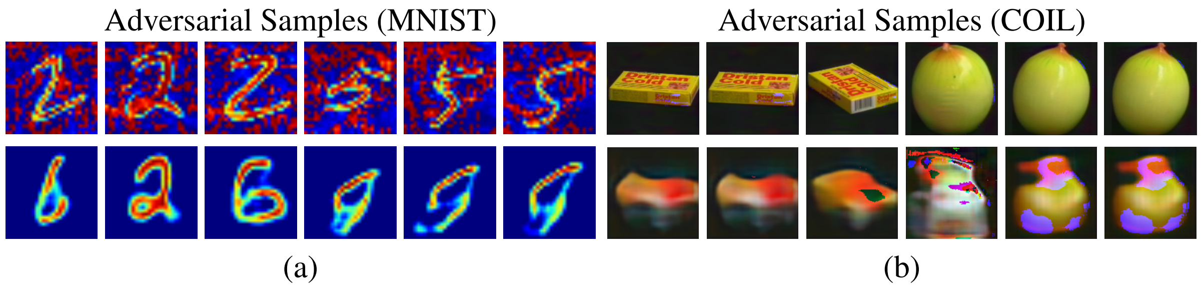
$$\begin{aligned}\underset{\eta}{\operatorname{argmin}}\, \mathcal{L}_{adv} = \underset{\eta}{\operatorname{argmin}}[(||x_o + \eta - g(f(x_o + \eta))||^2/\tau_{dec})^a \\ + (\mu_t - f(x_o + \eta))\Sigma_t(\mu_t - f(x_o + \eta))/\tau_{enc})^b + ||\eta||^2]\end{aligned} \tag{4.9}$$

where x_o is a original image we wish to corrupt, μ_t is the mean of target class, η is the noise added, f, g are the encoder and decoder respectively, and Σ_t denotes target class covariance in latent space. $a > 1$ and $b > 1$ represent constant exponents, that ensure that the adversarial loss grows steeply when the two threshold values are exceeded. Essentially, we aim for low reconstruction error and small change in the adversarial image while moving its embedding close to the target class mean. η is initialized with zeros.

We also ran the white box attack on 100 randomly sampled images from each of the 10 classes for MNIST and SVHN, by setting each of the 9 other classes as the target class. The samples generated by optimizing the adversarial objective in each of these cases were either correctly classified or rejected.

4.8.6 Fooling images.

We take 100 images sampled from the uniform distribution as inputs and optimize the white-box fooling attack objective given by Eq. (4.10), with each of the classes from the MNIST and SVHN datasets as the target classes. In Figure 4.4c, we visualize some of the images to which the attack converged and their reconstructions for the MNIST dataset, with the target classes



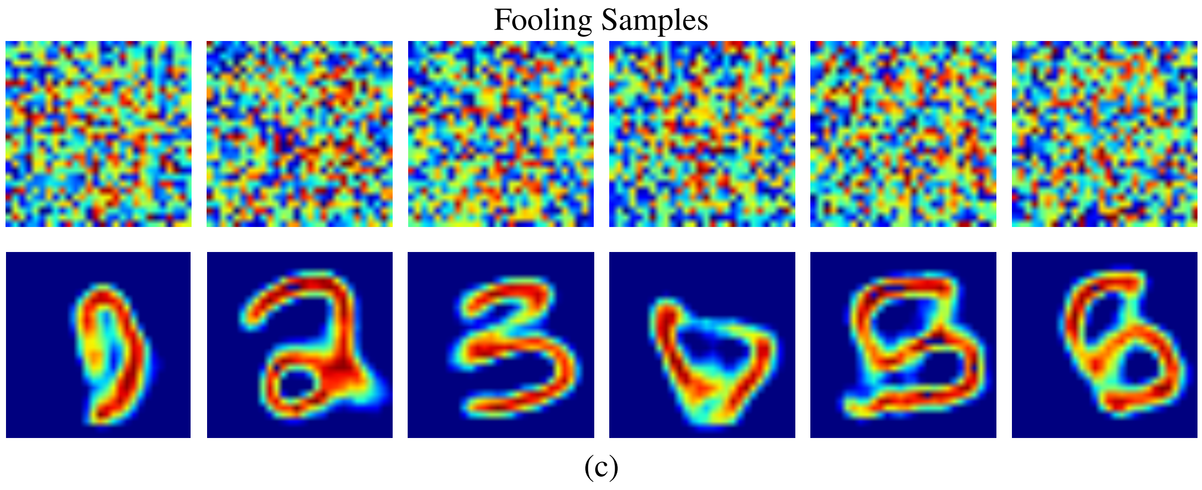


Figure 4.4: White box attack on MNIST and COIL dataset. (a) Targeted attack on MNIST. (b) Targeted attack on COIL. (c) Targeted fooling sample attack on MNIST. The first row represents the images to which the white-box attack converged, and the second row represents the corresponding reconstructed images.

$1, 2, \ldots, 6$.

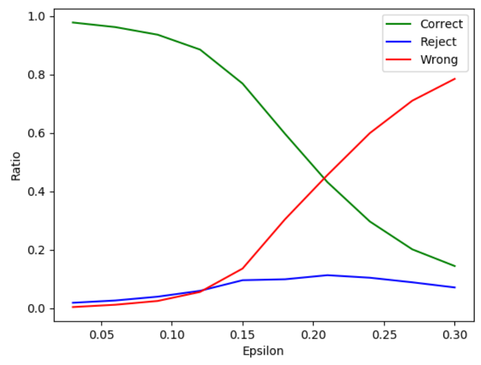
$$\underset{\eta}{\operatorname{argmin}} \mathcal{L}_{fool} = \underset{\eta}{\operatorname{argmin}}[(||\eta - g(f(\eta))||^2/\tau_{dec})^a + ((\mu_t - f(\eta))\Sigma_t(\mu_t - f(\eta))/\tau_{enc})^b] \quad (4.10)$$

Here, η, a, b, f, g, Σ_t and μ_t are as described in Sec. 4.8.5.

It has been shown that fooling samples are extremely easy to generate for state-of-the-art classifier networks [Goodfellow et al., 2015a, Nguyen et al., 2015]. Our technique, by design, gains resilience against such attacks as well. Since by definition, a fooling sample cannot look like a legitimate sample, it can not have small pixel space distance with any real image. This is exactly what can be noticed in the results in Figure 4.4c, where reconstruction errors are very high. Hence, most of the images to which this attack converges are rejected at the decoder, although they had managed to fool the encoder when considered in isolation. For the few cases where the images are not rejected, we observe that the attack method actually converged to a legitimate image of the target class.

Reclassifying Adversarial samples.

In this section, we present the performance of our reclassification technique. Although one could have used our decoder network to perform both 'ordinary' and 'adversarial' sample classification using Eq. (4.8), this process involves an iterative optimization. Hence, we only use it for the detected adversarial samples. The results are summarized in Table 4.5b.



(a)

ε	0.06	0.12	0.18	0.24	0.30
Accuracy	97%	93%	91%	87%	87%

(b)

Figure 4.5: In Fig. 4.5a We run FGSM with varying ε on the model with variable encoder distribution entropy, trained on MNIST data. The rejection rate stays low in this case, while the error rate increases with increasing ε. In Fig. 4.5b We present the reclassification accuracy for samples generated using FGSM on the MNIST dataset.

Following the same reclassification scheme, we also find that the method is able to correctly classify rejected test samples, thereby improving the overall accuracy achieved by the proposed method. For example, among the 181 samples rejected by the supervised model for the MNIST test dataset (as per Table 4.1), 110 samples are now correctly classified, improving the accuracy to 99.07%.

4.8.7 Entropy of the posterior

To compare the performance of the proposed network with the corresponding network with variable entropy of the posterior distribution $Q_\phi(Z|X)$, we ran experiments by letting $H(Q_\phi(Z|X))$ be variable, and keeping all other parameters the same. We tried the FGSM attack against the encoder of the model thus obtained, and observed that the adversarial sample detection capability of the network reduces drastically. This is justified by the fact that the reconstructions tend to be blurry in this case, thereby leading to a high reconstruction threshold. The results are shown in Figure 4.5a.

In order to further study the difference between the two cases, we train both variants of the network on the CelebA dataset, and observe that the Fréchet Inception Distance (FID) [Heusel et al., 2017a] score is significantly better for the model with a constant $H(Q_\phi(Z|X))$ (50.4) than the one with variable $H(Q_\phi(Z|X))$ (58.3). The FID scores are obtained by randomly sampling $10,000$ points from the latent distribution, and comparing the distribution of the images generated from the these points with the training image distribution.

4.9 Discussion

In this work, we have successfully demonstrated how a generative model can be used to gain defensive strength against adversarial attacks on images of relatively high resolution (128×128 for the COIL-100 dataset, for example). However, the proposed network is limited by the generative capability of VAE based architectures, and thus, might not scale effectively to ImageNet scale datasets [Deng et al., 2009]. In spite of this fact, keeping the underlying principles for adversarial sample detection and reclassification as described in this work, recent advances in invertible generative models such as Glow [Kingma and Dhariwal, 2018] can be exploited to scale to more complex datasets. Further, as discussed earlier, the problem of defending against adversarial attacks still remains an unsolved problem even for datasets with more structured images. Hence, our method can be used for practical applications such as secure medical image classification [Finlayson et al., 2018], biometrics identification, etc.

Human perception involves both discriminative and generative capabilities. Similarly, this chapter proposes a modification to VAEs to incorporate discriminative ability, besides using its generative ability to gain robustness against adversarial samples. The input space dimensionality (to the decoder) is drastically smaller compared to the input space dimensionality of image classifiers. Hence, it is much easier to attain dense coverage in the latent space, thereby minimizing the possibility of the occurrence of holes, leading to defensive capability against both adversarial and fooling images. With our construct, selective classification and semi-supervised learning become feasible under the same framework. A possible direction of future research would be to study how effectively the proposed approach can be scaled to more complex datasets by using recently proposed invertible generative modeling techniques.

4.10 Conclusion

We have successfully removed the requirement of a Gaussian prior and yet maintain computational tractability in a VAE framework. In the process, we have reduced variance in the back propagated gradients. We also have developed a technique that forces the latent space to take the shape of a GMM, thereby naturally learning a classifier with rejection capability. The big problem that VAEs are blurry, albeit alleviated, remains an open problem. Generative Adversarial Networks (GANs) however seem to avoid this issue quite well, but they suffer from not having an inference model, *i.e.*, given a real data point there is no easy way to infer the corresponding latent representation. This serves as the motivation for our next chapter. There we see how conditional GANs can alleviate, to some extent, the need for an inference model, and later we integrate an inference model into the GAN framework.

Chapter 5

Controlling the Generative Adversarial Networks

Generative models originate from the desire to estimate the density of high dimensional data. They can also be used as a graphics tool, especially the GAN variant. This is because recent progress has made them capable of sampling high-resolution images with unprecedented realism. So much so that GANs [Goodfellow et al., 2014] are starting to replace and augment traditional graphics pipelines [Zhang et al., 2019, Abdal et al., 2020]. Although the GAN frameworks are remarkable in their sampling capability, they offer little practical usability. Since in their original form, GANs only offer an unconditional sampling mechanism. Practically speaking, all we can do with GANs is to create new random data points which are similar to the training set. This is given by the generator, which maps a lower dimensional, easy to sample, latent variable to its training data. Now, this might be enough for us to judge how well the generator 'understands' the generative mechanism of high dimensional structured data, but, since we cannot access this latent representation for real images, this implicit 'knowledge' distilled in the generator remains 'locked away'. In other words, GANs could be made more useful if we could condition their generation using high-level parameters. By enforcing such conditions, they could be used as a graphics tool, *e.g.*, they can help generate rich images and videos using only very high-level instructions. This serves as the goal for this and the next chapter of this thesis. There exist two avenues to this goal, namely unsupervised and supervised approaches. Understandably, unsupervised methods offer less precise control and often offer control over factors that are not predictable a priori to training. Therefore, in this thesis, we mainly focus on the supervised conditional GANs, while briefly skimming over their unsupervised counterparts.

5.1 Inductive Bias

Unsupervised and semi-supervised methods hope to discover true generative mechanisms, which, in turn, should help downstream tasks. The concept of disentanglement plays a crucial role in the degree of success in this endeavor. The assumption here is that real-world high dimensional data such as video or images are generated from a few underlying semantically meaningful factors and that unsupervised leaning can discover these factors or an easy function of such factors. Although there is no formal definition of disentanglement, it can be roughly defined as a representation of data that separates the distinct informative factors of variations [Bengio et al., 2013a]. Recent work has argued that disentanglement in data representation is an important factor to capture [Bengio et al., 2013a, LeCun et al., 2015, Lake et al., 2017], for several downstream tasks. It has been shown recently, however, that it is impossible to learn to disentangle without data and model inductive biases [Locatello et al., 2019]. There are many ways of inducing inductive biases. We list two commonly used techniques of such efforts below.

Architectural Inductive bias As argued above, it is important to inject inductive biases into neural models for them to be useful in practice. StyleGAN [Karras et al., 2019a] achieves this by gradually increasing the resolution of its internal activations and adaptively normalizing them using the latent code. It has been shown that the activations corresponding to the different scales capture different semantically meaningful elements of the dataset, *e.g.*, hairstyle, skin tone, *etc*.

Inductive bias through latent distribution InfoGAN [Chen et al., 2016], an information theoretic extension of GANs, maximizes the mutual information between a small subset of latent variables and the observed variables. Subsequently, by specifically choosing the distribution of such latent variables, it is able to inject enough inductive bias into GANs such that writing style can be separated from the shape of a handwritten digit in the MNIST dataset [LeCun et al., 1998b].

Although, disentanglement using inductive bias of the learning model lets us gain control over generation, the control thus gained is not precise and is dependent on the learning dynamics. Moreover, it might yield a set of controls that are not relevant to a particular application.

5.2 Conditional GANs

Motivated by these two techniques, we ask ourselves the question—"Does inductive bias imparted by the representation of input variable also play a role in conditional generative models?". Conditional GANs were first introduced by Mirza and Osindero [2014], with the primary motivation of learning multi-modal mapping. Such a framework is incredibly useful where we need to model a one-to-many mapping, *e.g.*, image tagging. As described earlier, a generative model is of little use if it does not give us any control over the generated samples. Furthermore, as described in Sec. 5.1, it is impossible to discover the true generative factors completely unsupervised. Therefore, we have to resort to supervised training. We further show here that even under a fully supervised scenario, inductive bias imparted by different representations of the condition variables significantly impacts the quality of the end result. Conditional GANs provide a framework where we can provide the control we need explicitly. It is remarkably simple to follow the theoretical foundation of a conditional GAN. Let us recall the min-max game of the original GAN framework in Eq. (2.3). This is extended to its conditional version easily if, instead of working with the data space $\mathcal{X}$ and its density $P(X)$, we simply move everything to the joint space of data and its condition $(\mathcal{X}, \mathcal{Y})$. One of the modeling objectives, then, is to estimate the conditional distribution $P(X|Y)$. This gives us the following objective function (Eq.(5.1)) for a conditional GAN

$$\underset{\theta}{\operatorname{argmin}} \left\{ \underset{\phi}{\operatorname{argmax}} \left\{ log\left(1 - D_{\phi}\left(G_{\theta}(Z|Y), Y\right)\right) + log\left(D_{\phi}(x \sim P_D(X|Y), Y)\right) \right\} \right\}. \tag{5.1}$$

Here G_θ is the generator and D_ϕ is the discriminator, with learnable parameters θ and ϕ. Practically, it translates to feeding the condition Y to both the discriminator and the generator. However, the question of how to represent the condition Y, and where (in which layer) to inject it in the discriminator, is a more involved one. Recent work [Mirza and Osindero, 2014, Denton et al., 2015, Reed et al., 2016, Zhang et al., 2017b] concatenates an embedding of the condition Y to an activation vector at an arbitrary layer of the discriminator. There are, however, many other problem-specific ways proposed to inject the condition to the discriminator. AC-GAN by Odena et al. [2017] uses an auxiliary discriminator to modify the objective function to enforce the condition directly. Yet another way of enforcing the condition is to project the output of the features extracted at a certain layer by the discriminator onto the embedding vector of the condition, and then add the result into the downstream features [Miyato and Koyama, 2018]. This method is specifically inspired by the unimodality of the distribution of the condition $P(Y|X)$. What all these efforts highlight is that for best results, the representation of the condition and its injection into the discriminator is rather crucial.

Recalling that GANs, especially conditional GANs, have the potential to replace traditional graphics tools, one can, in theory, replace a traditional 3D rendering pipeline, simply be conditioning a GAN with all the traditional control parameters. To make this proposition concrete, we have to consider a specific problem. In the next section (Sec. 5.3), we study how best to condition a state-of-the-art GAN such that it has the same set of control parameter as a 3D model. We tackle the special case of a human face model.

5.3 Generative Interpretable Faces

5.3.1 Introduction

The ability to generate a person's face has several uses in computer graphics and computer vision. It can be used for several applications such as constructing a personalized avatar for multimedia applications, face recognition, face analysis, etc. Early work focuses on learning a low dimension representation of human faces using (PCA) spaces [Craw and Cameron, 1991, Cootes et al., 1995, 1998, 2001, Turk and Pentland, 1991] or higher-order tensor generalizations [Vasilescu and Terzopoulos, 2002]. Although they provide some semantic control, these methods use linear transformations in the pixel-domain to model facial variations and hence result in blurry images. Further, effects of rotations, *i.e.*, linear transformations in 3D space, are not well parameterized by linear transformations in 2D due to occlusions arising from the 3D rotation, resulting in poor image quality.

To overcome this, [Blanz and Vetter, 1999] introduced the morphable model, a statistical 3D model to parametrize facial shape and appearance variation. Such a model parametrizes rotation and facial variation in 3D and lets us generate images with explicit control by a set of parameters. These can then be combined with classical computer graphics pipelines to generate images of faces. This process roughly follows the following steps. A statistical face model (*e.g.*, [Cao et al., 2013, Paysan et al., 2009, Li et al., 2017]) is used to manipulate the shape, expression, or pose of the facial 3D geometry, which is then combined with a texture map (*e.g.*, [Saito et al., 2017]) and rendered to an image. Rendering of 3D face models lacks photo-realism due to the difficulty of modeling hair, eyes, and the mouth cavity (*i.e.*, teeth or tongue) or the absence of facial details like wrinkles in the geometry of the facial model. Further difficulties arise from modeling material properties of skin and hair and its interactions with light, which cause subsurface scattering, all of which are required for photo-realism of the final rendering. Bridging the realism gap between 3D model renderings and real images is an active research topic. Recent efforts that combine traditional rendering pipelines and learning-based components (*e.g.*, [Kim et al., 2018]) have shown promising results.

On the other hand, generative adversarial networks (GANs) have recently shown great success in generating photo-realistic face images of high resolution [Karras et al., 2018]. Methods like StyleGAN [Karras et al., 2019a] or StyleGAN2 [Karras et al., 2020] even provide high-level control over factors like pose or identity when trained on face images. However, these controlling factors are often entangled, and they are unknown prior to training. The control provided by these models does not allow the independent change of attributes like facial appearance, shape (*e.g.*, length, width, roundness, *etc*) or facial expression (*e.g.*, raise eyebrows, open mouth, *etc*) without changing other factors too. Although these methods have made significant progress in image quality, the provided control is not sufficient for graphics applications like facial editing, face reenactment, or facial animation.

In short, the discipline of generating face images reveals a gap between low-quality face images with explicit control from a 3D face model on one side of the spectrum, and high-quality images from 2D generative models without explicit control on the other side. We close this gap by combining the best of both works, namely, control of a 3D face model, and the image quality of a generative 2D face model. Our key insight is that incorporating 3D geometry from FLAME [Li et al., 2017] (*i.e.*, a publicly available statistical 3D face model) as supervised conditioning to a high quality generative 2D model such as StyleGAN [Karras et al., 2019a] results in a generative 2D face model called GIF (Generative Interpretable Faces) that produces photo-realistic images with explicit control for head pose, facial shape, and expression. Figure 5.1 illustrates generated images for varying FLAME shape, pose, and expression parameters for two different appearance embeddings (*i.e.*, representing different identities).

In summary, our main contributions are 1) a generative 2D face model with FLAME [Li et al., 2017] control, 2) use of conditioning on images rendered from FLAME to incorporate information about 3D face structure, 3) use of texture consistency constraints to improve disentanglement between different parameters and unconditioned factors. Training code and pretrained models are available publicly at `https://github.com/ParthaEth/GIF`

5.3.2 Related Work

Generative 3D face models: Representing and manipulating human faces in 3D has a long-standing history, dating back almost five decades to the parametric 3D face model of Parke [1974]. Blanz and Vetter [1999] propose a 3D morphable model (3DMM), the first generative 3D face model that uses linear subspaces to model shape and appearance variations. This has given rise to a variety of 3D face models to model facial shape [Booth et al., 2018, Dai et al., 2017, Paysan et al., 2009], shape and expression [Amberg et al., 2008, Blanz et al., 2003, Bolkart and Wuhrer, 2015, Cao et al., 2013, Ranjan et al., 2018, Vlasic et al., 2005], shape, expression and head pose [Li et al., 2017], localized facial details [Brunton et al., 2014a, Neog et al., 2016] and wrinkle details [Golovinskiy et al., 2006, Shin et al., 2014]. However, renderings of these models do not reach photo-realism due to the lack of high-quality textures.

To overcome this, Saito et al. [2017] introduce high-quality texture maps, and Slossberg et al. [2018] and Gecer et al. [2019] train GANs to synthesize textures with high-frequency details. These works enhance the realism when combined with detailed texture, but they focus only on a specific region (face region) and ignore hair, teeth, tongue, eyelids, eyes, etc. that are required for real-life applications. While separate part-based generative models of hair [Hu et al., 2017, Saito et al., 2018, Wei et al., 2018], eyes [Bérard et al., 2014], eyelids [Bermano et al., 2015],

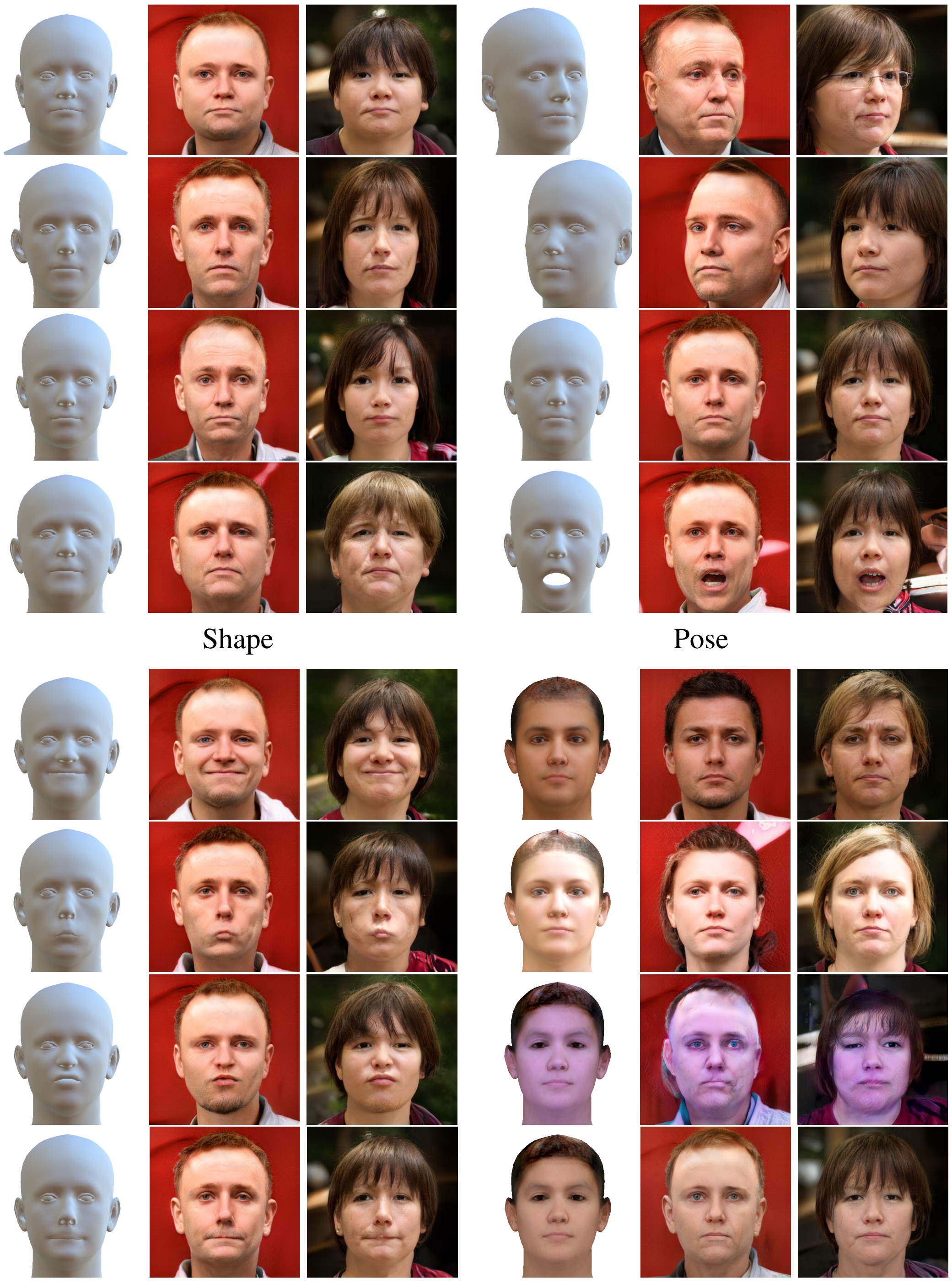


Figure 5.1: Face images generated by controlling FLAME [Li et al., 2017] parameters, appearance parameters, and lighting parameters. For shape and expression, two principal components are visualized at ± 3 standard deviations. The pose variations are visualized at $\pm \pi/8$ (head pose) and at $0, \pi/12$ (jaw pose). For shape, pose, and expression, the two columns are generated for two randomly chosen sets of appearance, lighting, and style parameters. For the appearance and lighting variations (right), the top two rows visualize the first principal components of the appearance space at ± 3 standard deviations, the bottom two rows visualize the first principal component of the lighting parameters at ± 2 standard deviations. The two columns are generated for two randomly chosen style parameters.

ears [Dai et al., 2018], teeth [Wu et al., 2016], or tongue [Hewer et al., 2018] exist, combining these into a complete realistic 3D face model remains an open problem.

Instead of explicitly modeling all face parts, Gecer et al. [2018] use image-to-image translation to enhance the realism of images rendered from a 3D face mesh. Nagano et al. [2018] generate dynamic textures that allow synthesizing expression dependent mouth interior and varying eye gaze. Despite significant progress of generative 3D face models [Brunton et al., 2014b, Egger et al., 2020], they still lack photo-realism.

Our approach, in contrast, combines the semantic control of generative 3D models with the image synthesis ability of generative 2D models. This allows us to generate photo-realistic face images, including hair, eyes, teeth, *etc.* with explicit 3D face model controls.

Generative 2D face models: Early parametric 2D face models like Eigenfaces [Sirovich and Kirby, 1987, Turk and Pentland, 1991], Fisherfaces [Belhumeur et al., 1997], Active Shape Models [Cootes et al., 1995], or Active Appearance Models [Cootes et al., 1998] parametrize facial shape and appearance in images with linear spaces. Tensor faces [Vasilescu and Terzopoulos, 2002] generalize these linear models to higher-order, generating face images with multiple independent factors like identity, pose, or expression. Although these models provided some semantic control, they produced blurry and unrealistic images.

StyleGAN [Karras et al., 2019a], a member of the broad category of GAN [Goodfellow et al., 2014] models, extends Progressive-GAN [Karras et al., 2018] by incorporating a style vector to gain partial control over the image generation, which is broadly missing in such models. However, the semantics of these controls are interpreted only post-training. Hence, it is possible that desired controls might not be present at all. InterFaceGAN [Shen et al., 2020] and StyleRig [Tewari et al., 2020] aim to gain control over a pre-trained StyleGAN [Karras et al., 2019a]. InterFaceGAN [Shen et al., 2020] identifies hyper-planes that separate positive and negative semantic attributes in a GAN's latent space. However, this requires categorical attributes for every kind of control, making it not suitable for a variety of aspects *e.g.*, facial shape, expression etc. Further, many attributes might not be linearly separable. StyleRig [Tewari et al., 2020] learns mappings between 3DMM parameters and the parameter vectors of each StyleGAN layer. StyleRig learns to edit StyleGAN parameters and thereby controls the generated image, with 3DMM parameters. The setup is mainly tailored towards face editing or face reenactment tasks. GIF, in contrast, provides full generative control over the image generation process (similar to regular GANs) but with semantically meaningful control over shape, pose, expression, appearance, lighting, and style.

CONFIG [Kowalski et al., 2020] leverages synthetic images to get ground truth control parameters and leverages real images to make the image generation look more realistic. However, the generated images still lack photo-realism.

HoloGAN [Nguyen-Phuoc et al., 2019] randomly applies rigid transformations to learnt features during training, which provides explicit control over 3D rotations in the trained model. While this is feasible for global transformations, it remains unclear how to extend this to transformations of local parts or parameters like facial shape or expression. Similarly, IterGAN [Galama and Mensink, 2019] also only models rigid rotation of generated objects using a GAN, but these rotations are restricted to 2D transformations.

There are also works that use variational autoencoders [Razavi et al., 2019, van den Oord et al., 2017] and flow-based methods [Kingma and Dhariwal, 2018] to generate images. These provide controllability, but do not reach the image quality of GANs.

Facial animation: A large body of work focuses on face editing or facial animation, which can be grouped into 3D model-based approaches (*e.g.*, [Geng et al., 2019, Kim et al., 2018, Lombardi et al., 2018, Thies et al., 2019, 2016, Ververas and Zafeiriou, 2020]) or 3D model-free methods (*e.g.*, [Bansal et al., 2018, Pumarola et al., 2018, Tripathy et al., 2020, Wu et al., 2018, Zakharov et al., 2019])

Thies et al. [2016] build a subject specific 3DMM from a video, reenact this model with expression parameters from another sequence, and blend the rendered mesh with the target video. Follow-up work use similar 3DMM-based retargeting techniques but replace the traditional rendering pipeline, or parts of it, with learnable components [Kim et al., 2018, Thies et al., 2019]. Slider-GAN [Ververas and Zafeiriou, 2020] and Geng et al. [2019] propose image-to-image translation models that, given an image of a particular subject, condition the face editing process on 3DMM parameters. Lombardi et al. [2018] learn a subject-specific autoencoder of facial shape and appearance from high-quality multi-view images that allow animation and photo-realistic rendering. Like GIF, all these methods use explicit control of a pre-trained 3DMM or learn a 3D face model to manipulate or animate faces in images, but in contrast to GIF, they are unable to generate new identities.

Zakharov et al. [2019] use image-to-image translation to animate a face image from 2D landmarks, ReenactGAN [Wu et al., 2018] and Recycle-GAN [Bansal et al., 2018] transfer facial movement from a monocular video to a target person. Pumarola et al. [2018] learn a GAN conditioned on facial action units for control over facial expressions. None of these methods provide explicit control over a 3D face representation.

All methods discussed above are task-specific, i.e., they are dedicated towards manipulating or animating faces, while GIF, in contrast, is a generative 2D model that is able to generate new identities, and also provides control of a 3D face model. Further, most of the methods are trained on video data [Bansal et al., 2018, Kim et al., 2018, Lombardi et al., 2018, Thies et al., 2019, 2016, Zakharov et al., 2019], in contrast to GIF which is trained from static images only. Regardless, GIF can be used to generate facial animations.

5.3.3 Preliminaries

StyleGAN2

StyleGAN2 [Karras et al., 2019b], a revised version of StyleGAN [Karras et al., 2018], produces photo-realistic face images at 1024×1024 resolution. Similar to StyleGAN, StyleGAN2, is controlled by a style vector Z. This vector is first transformed by a mapping network of 8 fully connected layers to an intermediate space. We simply call this space the W-space. These intermediate activations are then used to transform the activations of the progressively growing resolution blocks using adaptive instance normalization (AdaIN) layers. Although StyleGAN2 provides some high-level control, it still lacks explicit and semantically meaningful control. Our work addresses this shortcoming by distilling a conditional generative model out of StyleGAN2 and combining this with inputs from FLAME.

In the interest of completeness and ease of reproducibility, let us describe StyleGAN architecture in some detail. We recommend the experienced readers who are familiar with StyleGAN or those who are looking for only the bigger picture to skip to the next subsection titled FLAME. The StyleGAN architecture differentiates itself from other GNAs mainly in its Generator design. Therefore, we will only discuss the generator architecture here. The generator of

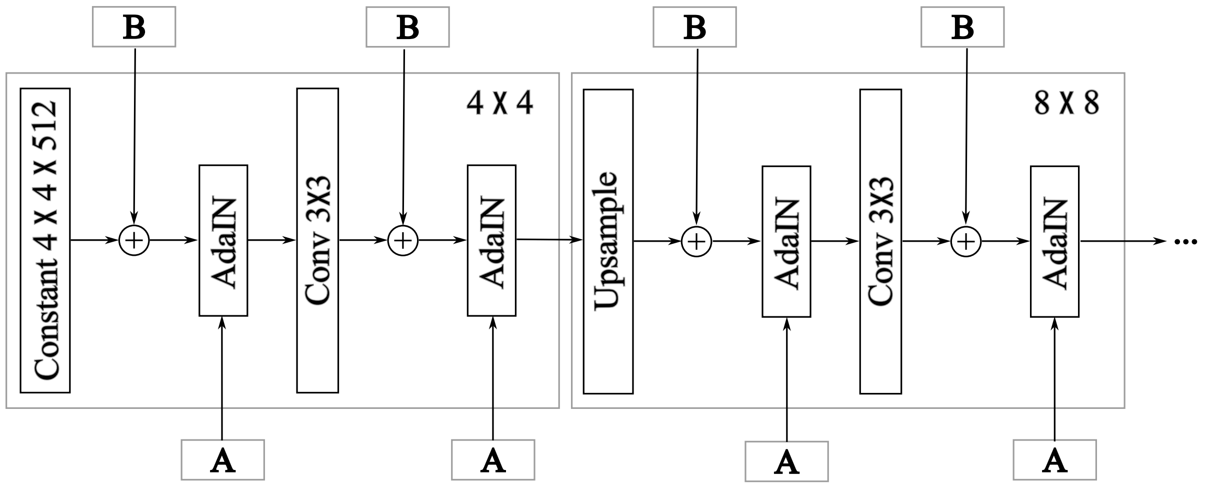


Figure 5.2: StyleGAN synthesis block. [A] represents a learned affine transform produced from the style vector, and [B] represents a broadcasting operation that is used to inject noise responsible for creating localized stochasticity in the generated image. As will be found later, (Fig. 5.3), we modify this noise broadcasting operation to inject the conditioning signal into the generator.

a StyleGAN has two main parts, i) mapping network (f), ii) synthesis network (S). The whole generator network G, therefore, can be described as $G(z) := S(f(z)) : \mathcal{R}^{512} \to \mathcal{R}^{1024\times1024}$.

The job of the mapping network is to map the latent code Z to an intermediate space, simply referred to as the W-space. Usually, the latent variable Z is assumed to be normally distributed in a 512 dimensional real vector space $\mathcal{R}^{512}$. The intermediate W-space is the range space of the mapping network (f). f consists of 8 fully connected feedforward layers with 512 neurons each, interleaved with relu activations. This choice makes the W-space to be also a 512 dimensional real valued vector space $\mathcal{R}^{512}$.

The synthesis network is slightly more complicated and best visualized as in Fig. 5.2. Interestingly, unlike traditional GANs it starts with a constant vector and the latent code progressively stylizes it into different images. Here, in Fig. 5.2, we visualize this network up to a special resolution of 8×8 pixels. In reality, we can repeat the resolution doubling block a few more times until we reach a desired output image size. In the interest of brevity and readability, we will skip over the details of adaptive normalization (AdaIN) operation and the noise broadcast operation. We refer the reader looking to reproduce our results to cite [Karras et al., 2019b] for such details.

FLAME

FLAME is a publicly available 3D head model [Li et al., 2017], $M(\beta,\theta,\psi) : \mathbb{R}^{|\beta|\times|\theta|\times|\psi|} \to \mathbb{R}^{N\times3}$, which given parameters for facial shape $\beta \in \mathbb{R}^{300}$, pose $\theta \in \mathbb{R}^{15}$ (*i.e.*,axis-angle rotations for global rotation and rotations around joints for neck, jaw, and eyeballs), and facial expression $\psi \in \mathbb{R}^{100}$, outputs a mesh with ($N = 5023$) vertices. We further transfer the appearance space of Basel Face Model [Paysan et al., 2009], parametrized by $\alpha \in \mathbb{R}^{|\alpha|}$, into FLAME's UV layout to augment it with a texture space. We use the same subset of FLAME parameters as RingNet [Sanyal et al., 2019], namely 6 pose coefficients for global rotation

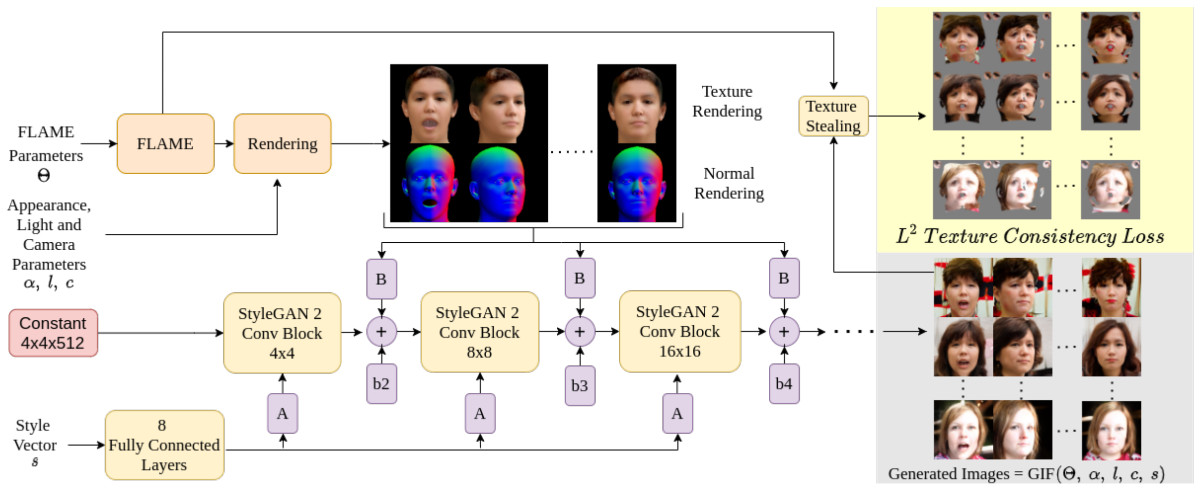

FLAME Parameters Θ
Appearance, Light and Camera Parameters α, l, c
FLAME
Rendering
Texture Rendering
Normal Rendering
Texture Stealing
L^2 Texture Consistency Loss
Constant 4x4x512
StyleGAN 2 Conv Block 4x4
StyleGAN 2 Conv Block 8x8
StyleGAN 2 Conv Block 16x16
B
A
b2
b3
b4
Style Vector s
8 Fully Connected Layers
Generated Images = GIF(Θ, α, l, c, s)

Figure 5.3: Our generator architecture is based on StyleGAN2 [Karras et al., 2019b], A is a learned affine transform, and B stands for per-channel scaling. We make several key changes, such as introducing 3D model generated condition through the noise injection channels and introduce texture consistency loss. We refer to the process of projecting the generated image onto the FLAME mesh to obtain an incomplete texture map as *texture stealing*.

and jaw rotation, 100 shape[1], and 50 [2] expression parameters, and we use 50 parameters for appearance. We use rendered FLAME meshes as the conditioning signal in GIF.

5.3.4 Method

Goal: Our goal is to learn a generative 2D face model controlled by a parametric 3D face model. Specifically, we seek a mapping $\text{GIF}(\Theta, \alpha, l, c, s) : \mathbb{R}^{156+50+27+3+512} \rightarrow \mathbb{R}^{P\times P\times 3}$, that given FLAME parameters $\Theta = \{\beta, \theta, \psi\} \in \mathbb{R}^{156}$, parameters for appearance $\alpha \in \mathbb{R}^{50}$, spherical harmonics lighting $l \in \mathbb{R}^{27}$, camera $c \in \mathbb{R}^3$ (2D translation and isotropic scale of a weak-perspective camera), and style $s \in \mathbb{R}^{512}$, generates an image of resolution $P \times P$.

Here, the FLAME parameters control all aspects related to the geometry. Appearance and lighting parameters control the face color (*i.e.*,skin tone, shadow lines, *etc*). Finally, the style vector s controls all factors that are not described either by the FLAME geometry or the appearance parameters (*e.g.*, hairstyle, background, etc.).

Training data

Our data set consists of Flickr images (FFHQ) introduced in StyleGAN [Karras et al., 2019a] and their corresponding FLAME parameters, appearance parameters, lighting parameters, and camera parameters. We obtain these using DECA [Feng et al., 2021], a publicly available monocular 3D face regressor. In total, we use about 65,500 FFHQ images, paired with the corresponding parameters. This is about 500 fewer than the original FFHQ dataset. We are forced to discard these images as we manually found out that DECA [Feng et al., 2021] fails to

[1]Simply the first 100 coefficients of total 300
[2]Simply the first 50 of total 100

provide good control parameters for these images. The obtained FFHQ training parameters are available for research purposes[3].

Condition representation

At this point, we have a dataset of the form $(\mathcal{X}, \mathcal{Y}) := \{(x_1, y_1), (x_1, y_1), ...(x_n, y_n)\}$, where x_i is a face image and y_i is its corresponding conditioning parameters (flame parameters, appearance, lighting, and camera parameters). Therefore, our dataset can be used to learn a conditional GAN formulation described in Sec. 5.2. However, we empirically found out that different representation format of the conditioning signal y_i influences the final result strongly. Next we shall study this phenomenon in detail and make a few important design choices.
Condition cross-talk: The vanilla conditional GAN formulation, as described in Sec. 5.2 does not encode any semantic factorization of the conditional probability distributions that might exist in the nature of the problem. Consider a situation where the true data depends upon two independent generating factors C_1 and C_2, *i.e.*,the true generation process of our data is $x \sim P(X|C_1, C_2)$ where $P(X, C_1, C_2) = P(X|C_1, C_2) \cdot P(C_1) \cdot P(C_2)$. Ideally, given complete data (often infinite) and a perfect modeling paradigm, this factorization should emerge automatically. However, in practice, neither of these can be assumed. Hence, the representation of the condition highly influences the way it gets associated with the output. We refer to this phenomenon of independent conditions influencing each other as – *condition cross-talk*. Inductive bias introduced by condition representation in the context of conditional cross-talk is empirically evaluated in Sec. 5.3.5.
Pixel-aligned conditioning: Learning the rules of graphical projection (orthographic or perspective) and the notion of occlusion as part of the generator is wasteful if explicit 3D geometry information is present, as it approximates classical rendering operations, which can be done learning-free and are already part of several software packages (*e.g.*, [Loper and Black, 2014, Ravi et al., 2020]). Hence, we provide the generator with explicit knowledge of the 3D geometry by conditioning it with renderings from a classical renderer (as seen in the left half of Fig. 5.3). This makes pixel-localized association between the FLAME conditioning signal and the corresponding generated image (by the generator) as well as the corresponding real image (by the discriminator). We find that, although a vanilla conditional GAN achieves comparable image quality, GIF learns to better obey the given condition.

We condition GIF on two renderings as shown in Fig. 5.3, one provides pixel-wise color information (referred to as texture rendering), and the other provides information about geometry (referred to as normal rendering). Normal renderings are obtained by rendering the mesh with a color-coded map of the surface normals $\mathbf{n} = \mathbf{N}(M(\beta, \theta, \psi))$.

Both renderings use a scaled orthographic projection with the camera parameters provided with each training image. For the color rendering, we use the provided inferred lighting and appearance parameter. The choice of the representation parametrization is dictated by the interface mechanism we chose to follow [Feng et al., 2021]. This inference mechanism is out of the scope of this thesis, therefore, for details, we refer the reader to DECA [Feng et al., 2021]. The texture and the normal renderings are concatenated along the color channel to form a pixel aligned six channel conditioning images. This conditioning image is next injected into the generator via the noise broadcasting operation [B] (see Fig. 5.2 and Fig. 5.3) of StyleGAN.

[3] `https://gif.is.tue.mpg.de/`

One technical difficulty associated with this approach however is the dimensionality mismatch between the conditioning image and the intermediate activations. To match the spacial dimensions we simply bilinearly interpolate the conditioning image and to match the channel dimensions we introduce a convolution layer with appropriate kernel size. As shown in the experiment section, Sec. 5.3.5 this way to condition the generator performs better than naive vector conditioning. As demonstrated in Sec. 5.3.5, this conditioning mechanism helps reduce condition cross-talk.

GIF architecture

The model architecture of GIF is based on StyleGAN2 [Karras et al., 2019b], with several key modifications. Since our conditioning signal has the same spacial size as that of the the images, we simply concatenate the six channel conditioning images with the images for the faces to form nine channel images that contain the data and its condition together in a specially aligned fashion. This makes it straightforward to modify the original StyleGAN discriminator, namely, we simply modify the input convolution layer to accommodate for nine input channels. The modification of the generator network is a little more involved. We have already discussed the modifications necessary to re-purpose the noise introduction mechanism of StyleGAN to inject the conditioning signal. The final modification to the generator is the introduce a Style-embedding. This is discussed in the follow paragraph. An overview of our generator model is shown in Figure 5.3. We step through the architecture details in this video – `https://youtu.be/-ezPAHyNH9s?&start=111&end=177`.

Style embedding: Rendering the textured FLAME mesh does not consider hair, mouth cavity (i.e., teeth or tongue), and fine-scale details like wrinkles and pores. Generating a realistic face image, however, requires these factors to be considered. We introduce a style vector $s \in \mathbb{R}^{512}$ to model these factors. Note that original StyleGAN2 has a similar input named ‘style vector’ denoted by Z, with the same dimensionality. However, instead of drawing it randomly from a standard normal $\mathcal{N}(0,I)$ distribution (as common in GANs), we assign a random but unique vector for every image. Here we follow roughly the motivations of autodecoders [Tan and Mayrovouniotis, 1995] but with the difference that we never compute reconstruction loss. This is further motivated by the key observation that in the FFHQ dataset [Karras et al., 2019a], each identity mostly occurs only once and mostly has a unique background. Hence, if we use a dedicated vector for each image using, e.g., an embedding layer, we will encode an inductive bias for this vector to capture background and appearance-specific information.

Noise channel conditioning: StyleGAN and StyleGAN2 insert random noise images at each resolution level into the generator, which mainly contributes to local texture changes *i.e.*,the changes that are confined specially in the input noise stays confined specially in the output images (see Sec 3.2 of [Karras et al., 2019a]). As described in the previous section, we replace this random noise by the concatenated textured and normal renderings from the FLAME model, and insert scaled versions of these renderings at different resolutions into the generator. This is motivated by the observation that varying FLAME parameters, and therefore varying FLAME renderings, should have direct, pixel-aligned influence on the generated images.

Texture consistency

Since GIF's generation is based on an underlying 3D model, we can further constrain the generator by introducing a texture consistency loss optimized during training. We first generate a set of new FLAME parameters by randomly interpolating between the parameters within a mini batch. Next, we generate the corresponding images with the same style embedding s, appearance α and lighting parameters l by an additional forward pass through the model. Finally, the corresponding FLAME meshes are projected onto the generated images to get a partial texture map (also referred to as 'texture stealing'). To enforce pixel-wise consistency, we apply an L^2 loss on the difference between pairs of texture maps, considering only pixels for which the corresponding 3D point on the mesh surface is visible in both the generated images. We find that this texture consistency loss improves the parameter association (see Sec. 5.3.5).

5.3.5 Experiments

Qualitative evaluation

Condition influence: As described in Sec. 5.3.4, GIF is parametrized by FLAME parameters $\Theta = \{\beta, \theta, \psi\}$, appearance parameters α, lighting parameters l, camera parameters c and a style vector s. Figure 5.4 shows the influence of each individual set of parameters by progressively exchanging one type of parameter in each row. The top and bottom rows show GIF generated images for two sets of parameters, randomly chosen from the training data.

Exchanging style (column 2) most noticeably changes hairstyle, clothing color, and the background. Shape (column 3) is strongly associated to the person's identity, among other factors. The expression parameters (column 4) control the facial expression, the best visible around the mouth and cheeks. The change in pose parameters (column 5) affects the orientation of the head (*i.e.*,head pose) and the extent of the mouth opening (jaw pose). Finally, appearance (column 6) and lighting (column 7) change the skin color and the lighting specularity.

Random sampling: To further evaluate GIF qualitatively, we sample FLAME parameters, appearance parameters, lighting parameters, and style embeddings and generate random images, shown in Figure 5.5. For shape, expression, and appearance parameters, we sample parameters of the first three principal components from a standard normal distribution and keep all other parameters at zero. For pose, we sample from a uniform distribution in $[-\pi/8, +\pi/8]$ (head pose) for rotation around the y-axis, and $[0, +\pi/12]$ (jaw pose) around the x-axis. For lighting parameters and style embeddings, we choose random samples from the set of training parameters. Figure 5.5 shows that GIF produces photo-realistic images of different identities with a large variation in shape, pose, expression, skin color, and age. Figure 5.1 further shows rendered FLAME meshes for generated images, demonstrating that GIF generated images are well associated with the FLAME parameters.

Speech driven animation: As GIF uses FLAME's parametric control, it can directly be combined with existing FLAME-based application methods such as VOCA [Cudeiro et al., 2019], which animates a face template in FLAME mesh topology from speech. For this, we run VOCA for a speech sequence, render them, and use these parameters to drive GIF for different appearance embeddings (see Figure 5.6). For more qualitative results and the full animation sequence, please follow this link – `https://youtu.be/-ezPAHyNH9s?&start=230&end=246`.

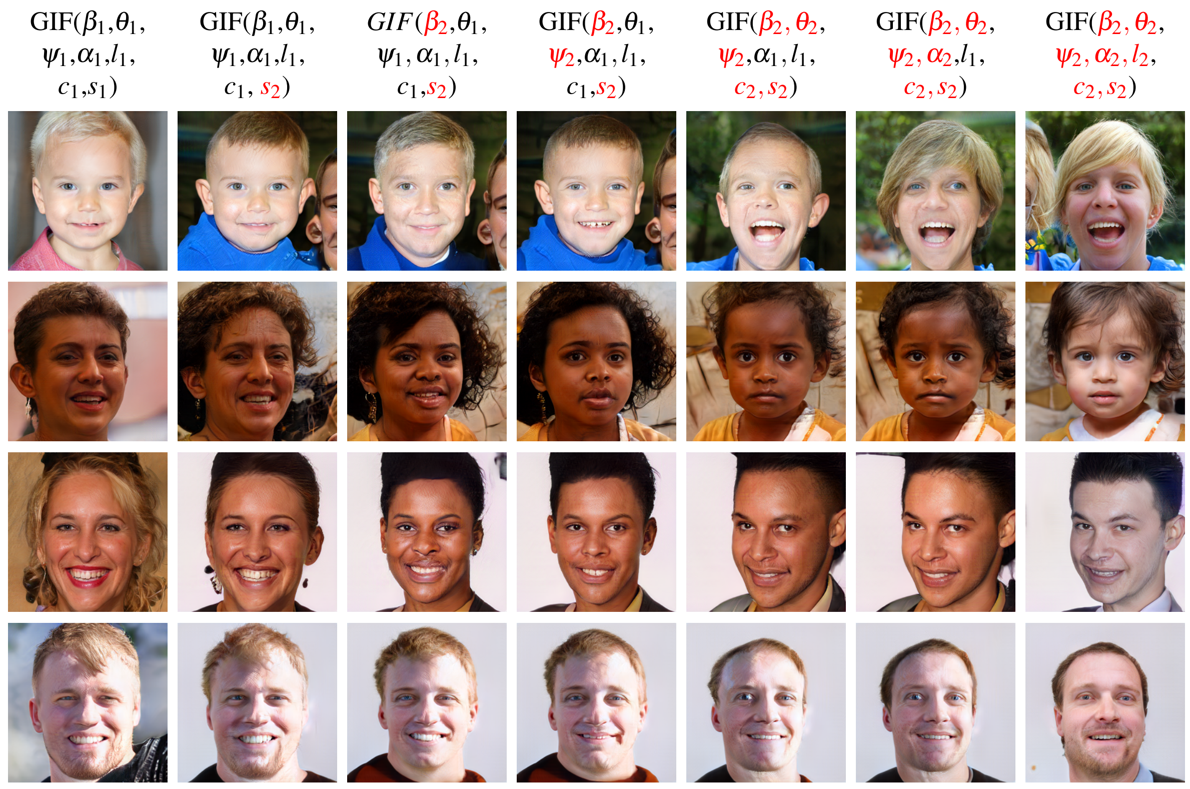

GIF(β_1,θ_1, ψ_1,α_1,l_1, c_1,s_1)
GIF(β_1,θ_1, ψ_1,α_1,l_1, c_1, s_2)
GIF(β_2,θ_1, ψ_1,α_1,l_1, c_1,s_2)
GIF(β_2,θ_1, ψ_2,α_1,l_1, c_1,s_2)
GIF(β_2, θ_2, ψ_2,α_1,l_1, c_2,s_2)
GIF(β_2, θ_2, ψ_2, α_2,l_1, c_2,s_2)
GIF(β_2, θ_2, ψ_2, α_2, l_2, c_2,s_2)

Figure 5.4: Impact of individual parameters, when being exchanged between two different generated images one at a time. From left to right we progressively exchange style, shape, expression, head and jaw pose, appearance, and lighting of the two parameter sets. We progressively change the color of the parameter symbol that is effected in every column to red.

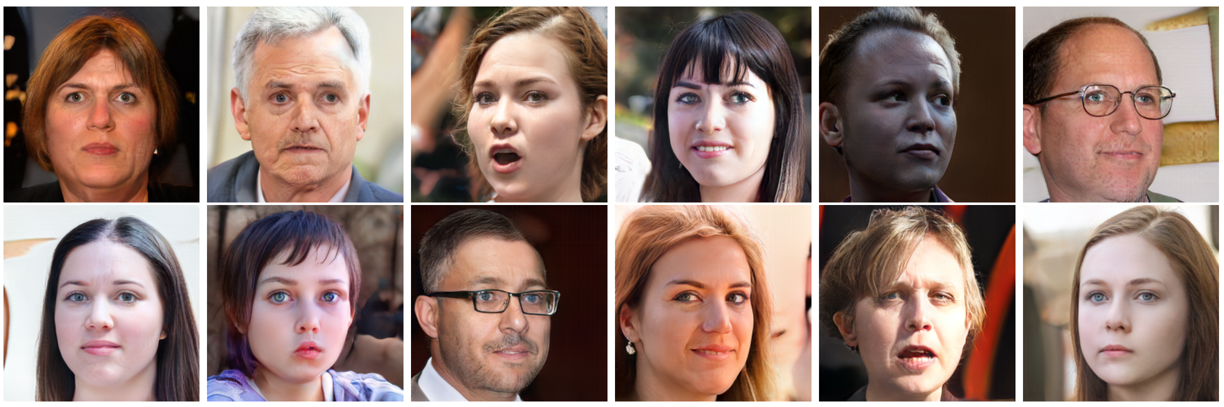


Figure 5.5: Images obtained by randomly sampling FLAME, appearance parameters, style parameters, and lighting parameters. Specifically, for shape, expression, and appearance parameters, we sample parameters of the first three principal components from a standard normal distribution and keep all other parameters at zero. We sample pose from a uniform distribution in $[-\pi/8, +\pi/8]$ (head pose) for rotation around the y-axis, and $[0, +\pi/12]$ (jaw pose) around the x-axis

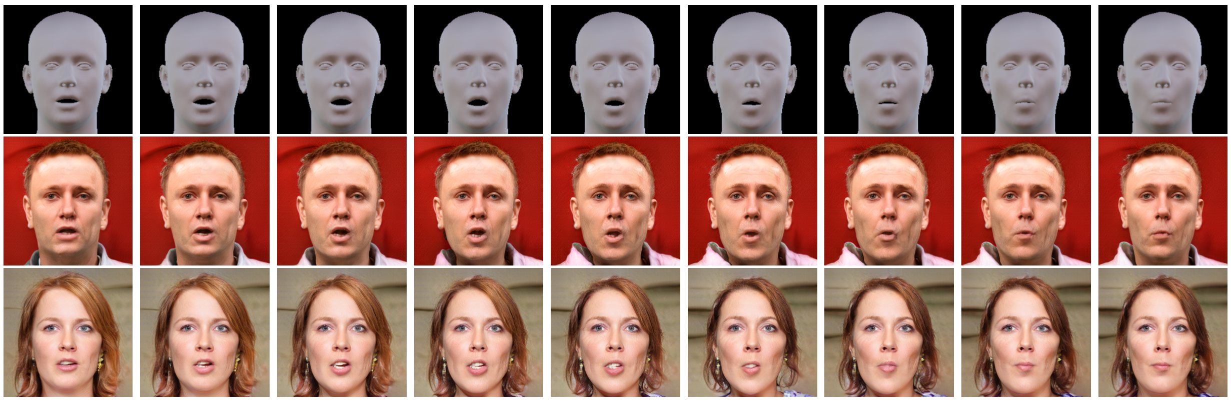
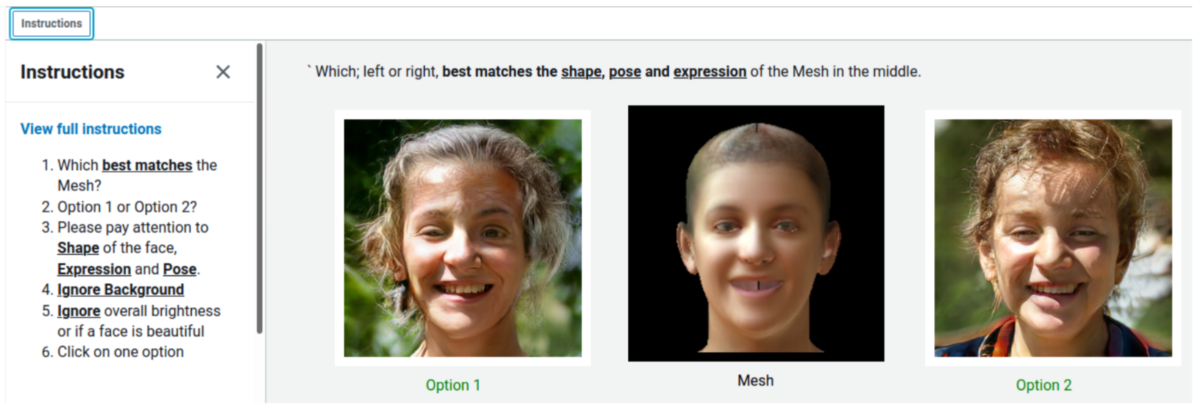


Figure 5.6: Combination of GIF and an existing speech-driven facial animation method by generating face images for FLAME parameters obtained from VOCA [Cudeiro et al., 2019]. Sample frames to highlight jaw pose variation. Please see the video for the full animation at – `https://youtu.be/-ezPAHyNH9s?&start=230&end=246`.

Quantitative evaluation

We conduct two Amazon Mechanical Turk (AMT) studies to quantitatively evaluate i) the effects of ablating individual model parts, and ii) the disentanglement of geometry and style. We compare GIF in total with 4 different ablated versions, namely *vector conditioning*, *no texture interpolation*, *normal rendering*, and *texture rendering conditioning*. For the *vector conditioning* model, we directly provide the FLAME parameters as a 236 dimensional real valued vector. In the *no texture interpolation* model, we drop the texture consistency during training. The *normal rendering* and *texture rendering conditioning* models condition the generator and discriminator only on the normal rendering or texture rendering, respectively, while removing the other rendering. For a walk through of this AMT interface, please follow this link – `https://youtu.be/-ezPAHyNH9s?&start=177&end=230`
Ablation experiment: Participants see three images, a reference image in the center, which shows a rendered FLAME mesh, and two generated images to the left and right in random order. Both images are generated from the same set of parameters, one using GIF, another an ablated GIF model. An example of such an experimental setup is shown in Fig. 5.7.

Participants then select the generated image that corresponds best with the reference image. Table 5.1a shows that with the texture consistency loss, normal rendering, and texture rendering conditioning, GIF performs slightly better than without each of them. Participants tend to select GIF generated results over a vanilla conditional StyleGAN2 (refer to Sec 5.3.4 for architecture details). Furthermore, Figure 5.1b quantitatively evaluates the image quality with FID scores, indicating that all models produce similar high-quality results.
Geometry-style disentanglement: In this experiment, we study the entanglement between the style vector and FLAME parameters. We find qualitatively that the style vector mostly controls aspects of the image that are not influenced by FLAME parameters like background, hairstyle, etc. (see Figure 5.4). To evaluate this quantitatively, we randomly pair style vectors and FLAME parameters and conduct a perceptual study with AMT (see Fig 5.8). Participants see a rendered FLAME image and GIF generated images with the same FLAME parameters but with a variety of different style vectors. Participants then rate the similarity of the generated image to shape, pose and expression of the FLAME rendering on a standard 5-Point Likert scale


Instructions

Instructions

View full instructions

1. Which **best matches** the Mesh?
2. Option 1 or Option 2?
3. Please pay attention to **Shape** of the face, **Expression** and **Pose**.
4. **Ignore Background**
5. **Ignore** overall brightness or if a face is beautiful
6. Click on one option

` Which; left or right, **best matches the shape, pose and expression** of the Mesh in the middle.

Figure 5.7: An example of the experiment window shown to the AMT participants to compare between GIF and an ablated version of itslef.

	Vec. cond.	No Tex. interp.	Norm. rend. cond.	Tex. rend. cond.
GIF	89.4%	51.1%	55.8%	51.7%

(a)

GIF	Vec. cond.	No Tex. interp.	Norm. rend. cond.	Tex. rend. cond.
8.94	10.34	11.71	9.89	11.28

(b)

Table 5.1: (5.1a) AMT ablation experiment. Preference percentage of GIF generated images over ablated models and vector conditioning model. Participants were instructed to “pay particular attention to shape, pose, and expression and ignore image quality”. (5.1b) FID scores of images generated by GIF and ablated models (lower is better). Note that this score only evaluates image quality and does not judge how well the models obey the underlying FLAME conditions.

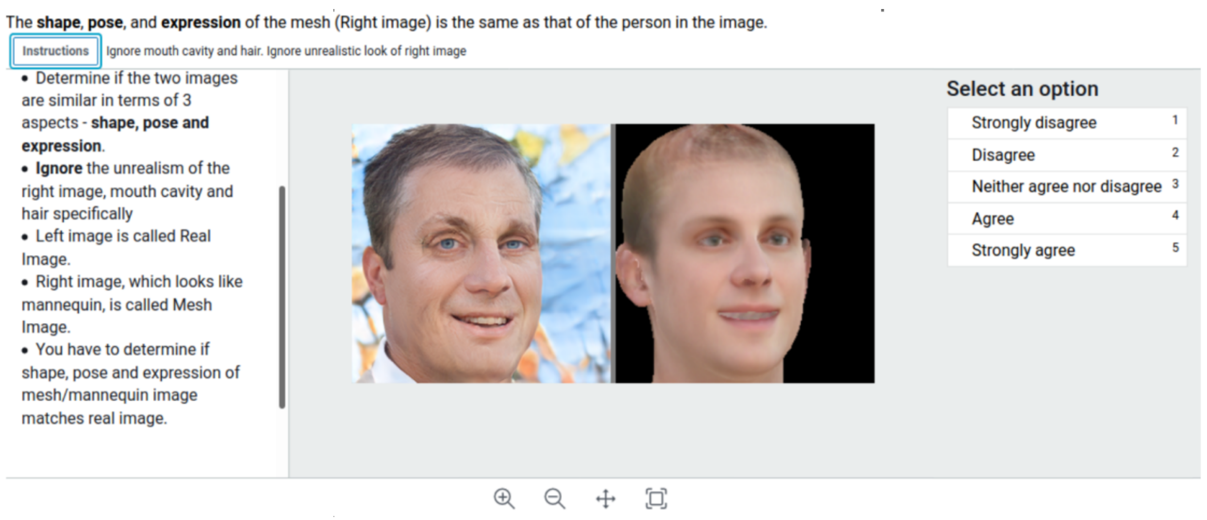
(*i.e.*, 1: Strongly Disagree, 2. Disagree, 3. Neither agree nor disagree, 4. Agree, 5. Strongly Agree). An example screenshot of this experiment page is shown in Fig 5.8

We use 10 randomized styles s and 500 random FLAME parameters totaling to 5000 images. We find that the majority of the participants agree that generated images and FLAME rendering are similar, irrespective of the style (see Figure 5.9a).

Re-inference error: In the spirit of DiscoFaceGAN [Deng et al., 2020], we run DECA [Feng et al., 2021] on the generated images and compute the Root Mean Square Error (RMSE) between the FLAME face vertices of the input parameters and the inferred parameters, as reported in Table 5.9b. We generate a population of 1000 images by randomly sampling one of the shape, pose and expression latent spaces while holding the rest of the generating factors to their neutral. Thus, we find an association error for individual factors.

5.3.6 Discussion

GIF is trained on the FFHQ data-set and hence inherits some of its limitations, *e.g.*, the images are roughly eye-centered. Although StyleGAN2 [Karras et al., 2020] has to some extent addressed this issue (among others), it has failed to do so completely. Hence, rotations of the head



The **shape**, **pose**, and **expression** of the mesh (Right image) is the same as that of the person in the image.

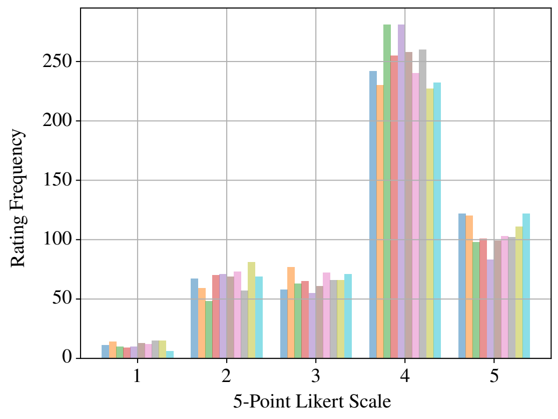
Instructions | Ignore mouth cavity and hair. Ignore unrealistic look of right image

- Determine if the two images are similar in terms of 3 aspects - **shape, pose and expression**.
- **Ignore** the unrealism of the right image, mouth cavity and hair specifically
- Left image is called Real Image.
- Right image, which looks like mannequin, is called Mesh Image.
- You have to determine if shape, pose and expression of mesh/mannequin image matches real image.

Select an option

Strongly disagree	1
Disagree	2
Neither agree nor disagree	3
Agree	4
Strongly agree	5

Figure 5.8: An example of the experiment window shown to the AMT participants to quantify Geometry-style disentanglement. It contains a random style vector and a flame parameter from the training set.



(a)

Model	Shape	Expression	Pose
Vec. cond.	3.43 mm	23.05 mm	29.69 mm
GIF	**3.02** mm	**5.00** mm	**5.61** mm

(b)

Figure 5.9: (5.9a) Preference frequency of styles on a 5-Point Likert Scale. Note that almost all style vectors, represented with different colors here, get a similar distribution of likeness ratings indicating that they do not influence the perceived FLAME conditioning. (5.9b) To evaluate shape error, we generate 1024 random faces from GIF and re-infer their shape using DECA [Feng et al., 2021]. Using FLAME, we compute face region vertices twice: once with GIF's input condition and once with re-inferred parameters. Finally, the RMSE between these face region vertices are computed. The process is repeated for expression and pose.

look like an eye-centered rotation, which involves a combination of 3D rotation and translation as opposed to a pure neck-centered rotation. Adapting GIF to use an architecture other than StyleGAN2 or training it on a different data set to improve image quality is subject to future work.

As FLAME renderings for conditioning GIF must be similarly eye-centered as the FFHQ training data, we compute a suitable camera parameter from given FLAME parameters so that the eyes are located roughly at the center of the image plane, with a fixed distance between the eyes. However, for profile view poses, this can not be met without an extreme zoomed in view. This often causes severe artifacts. Please see Appendix B for examples.

GIF requires a statistical 3D model and a way to associate its parameters to a large data set of high-quality images. While 'objects' like human bodies [Pavlakos et al., 2019] or animals [Zuffi et al., 2019] potentially fulfill these requirements, it remains unclear how to apply GIF to general object categories.

Faulty image-to-3D-model associations stemming from the parameter inference method potentially degrade GIF's generation quality. One example is the ambiguity between lighting and appearance, which causes most color variations in the training data to be described by lighting variation rather than by appearance variation. GIF inherits these errors.

Finally, as GIF is solely trained from static images without multiple images per subject, generating images with varying FLAME parameters is not temporally consistent. Several parts are not modeled by FLAME (*e.g.*, hair, mouth cavity, background, *etc.*). For these, GIF generates them either independently for every generated image or follows spurious correlations present in the dataset. When such images are put together to form am a video sequence, we get jittery results. Training or refining GIF on temporal data with additional temporal constraints during training remains subject to future work.

5.3.7 Conclusion

We present GIF, a generative 2D face model, with high realism and with explicit control from FLAME, a statistical 3D face model. Given a data set of approximately 65,500 high-quality face images with associated FLAME model parameters for shape, global pose, jaw pose, expression, and appearance parameters, GIF learns to generate realistic face images that associate with them. Our key insight is that conditioning a generator network on explicit information rendered from a 3D face model allows us to decouple shape, pose, and expression variations within the trained model. Given a set of FLAME parameters associated with an image, we render the corresponding FLAME mesh twice, once with color-coded normal details, once with an inferred texture, and insert these as condition to the generator. We further add a loss that enforces consistency in texture for the reconstruction of different FLAME parameters for the same appearance embedding. This encourages the network during training to disentangle appearance and FLAME parameters, and provides us with better temporal consistency when generating frames of FLAME sequences. Finally we devise a comparison-based perceptual study to evaluate continuous conditional generative models quantitatively.

Chapter 6

Invertible GANs (InvGAN)

In the last chapter, we have studied one of the two major ways of making GANs practically useful, namely making them a powerful graphics tool by carefully conditioning the generation process with the desired control parameters. The other key missing factor in GAN utility is an inference model. An inference model in this context is a model that, given a data point, can infer its latent representation. It would unlock semi-supervised learning, downstream learning, clustering, in-painting *etc*, using a GAN framework. To explore all of these possibilities we bake into the original formulation of GAN an inference module.

6.1 Introduction

The ability to generate photo-realistic images of objects such as human faces (see Chap. 5) or fully clothed bodies has wide applications in computer graphics and computer vision. Traditional computer graphics, based on physical simulation, often fails to produce photo-realistic images of objects with complicated geometry and material properties. In contrast, modern data-driven methods, such as deep learning based generative models, show great promise for realistic image synthesis [Karras et al., 2019a, 2020]. Among the four major categories of generative models, generative adversarial networks (GANs), variational auto-encoders (VAEs), normalizing flow networks and autoregressive models, GANs deliver images with the best visual quality. Although recent efforts in VAEs [Child, 2021, Razavi et al., 2019] have tremendously improved their generation quality, they still use larger latent space dimensions and deliver lower quality images. Autoregressive models are very slow to sample from and do not provide a latent representation of the data it operates on. Finally, flow-based methods [Kobyzev et al., 2020] do not perform dimensionality reduction and hence produce large models and latent representations. On the other hand, GANs do not provide a mechanism to embed real images into the latent space, *i.e.*, they lack an inference model. This limits them as a tool for image editing and manipulation. Specifically, while several methods exist, there is no method that trains a GAN so that it can be efficiently and effectively inverted. To that end, we propose *InvGAN*, an invertible GAN with an inference module that can embed real images into the latent space. InvGAN has a wide range of applications.

Representation Learning. GANs learn a latent representation of the training data. This representation has been shown to be well-structured [Karras et al., 2019a, Brock et al., 2018, Karras et al., 2020], allowing GANs to be employed for a variety of downstream tasks (*e.g.*, classification, regression and other supervised tasks) [Marriott et al., 2020, Ramaswamy et al., 2021]. We extend the GAN framework to include an inference model that embeds real images into the latent space. InvGAN addresses this problem and can be used to support representation

learning [Chen et al., 2020, Locatello et al., 2019], data augmentation [Brock et al., 2018, dos Santos Tanaka and Aranha, 2019] and algorithmic fairness [Balakrishnan et al., 2020, Sharmanska et al., 2020, Sattigeri et al., 2018]. Algorithmic fairness in the context of a classifier is roughly defined as the trade-off that the model strikes between overall performance and performance in the disadvantaged groups. A generative model is often used in a variety of ways to shift this balance towards a more fair point. For a specific use case of InvGAN in this context, please refer to [Zietlow et al., 2022]. Previous methods of inversion rely on computationally expensive optimization of the inversion processes. This limits their capacity to perform data augmentation for a secondary model, *i.e.*, the whole data augmentation has to happen before training of the secondary model starts. Efficient, photo-realistic, semantically consistent, and model-based inversion is the key to online and adaptive use-cases.
Conditional Image Editing. Recent work shows that even unsupervised GAN training isolates several desirable generative characteristics [Nguyen-Phuoc et al., 2019, Voynov and Babenko, 2020]. Prominent examples are correspondences between latent space directions and *e.g.*, hairstyle, skin tone and other visual characteristics. Recent works provide empirical evidence suggesting that one can find paths in the latent space (albeit non-linear) that allow for editing individual semantic aspects. GANs therefore have the potential to become a high-quality graphics editing tool [Ghosh et al., 2020a, Tewari et al., 2020]. However, without a reliable mechanism for projecting real images into the latent space of the generative model, editing of real data is impossible. InvGAN take a step towards addressing this problem.

6.2 Related work

The GAN inversion task has been addressed in two primary ways (1) using an inversion model (often a deep neural network), (2) embedding real images into the latent space of a trained generator using an iterative optimization-based method, typically initialized with a deep model.
Optimization based: iGAN [Zhu et al., 2016] optimizes for a latent code while minimizing the distance between a generated image and a source image. To ensure uniqueness of the preimage of a GAN-generated data point [Lipton and Tripathi, 2017], employ stochastic clipping. As the complexity of the GAN generators increases, an inversion process based on gradient descent and pixel space MSE is insufficient. Addressing this, Rameen et al. specifically target StyleGAN generators and optimize for perceptual loss Abdal et al. [2019, 2020]. However, they invert into the $w+$ space, the so-called extended w space of StyleGAN. This results in high dimensional latent codes and consequently prolongs inversion time. This can also produce out-of-distribution latent representations, which makes them unsuitable for downstream tasks. Contrary to these drawbacks, InvGAN offers fast inference embedding in the non-extended latent space.
Model based: BiGAN [Donahue et al., 2017] and ALI [Dumoulin et al., 2017] invert the generator of a GAN during the training process by learning the joint distribution of the latent vector and the data in a completely adversarial setting. However, the quality is limited, partially because of the choice of DCGAN [Radford et al., 2016] and partially because of the significant dimensionality and distribution diversity between the latent variable and the data domain Donahue and Simonyan [2019]. More recent models target the StyleGAN architecture Richardson et al. [2021], Wei et al. [2021], Zhu et al. [2019] and achieve impressive results. Most leverage StyleGAN peculiarities, *i.e.*, they invert in the $W+$ space – so adaptation to other GAN back-

bones is non-trivial. Adversarial latent auto-encoders Pidhorskyi et al. [2020], are closest to our current work. Our model and adversarial autoencoders can be made equivalent with a few alterations to the architecture and to the optimization objective. We discuss this more in detail in Sec. 6.3.2. Our method, in contrast to previous works discussed in this section, neither uses any data set specific loss nor does it depend upon any specific network architecture.
Hybrid optimization and regression based: Guan et al. [2020] train a regressor that is used to initialize an optimization-based method to refine the regressor's guess. However, to achieve good results, this method uses an identity loss to guide the refinement procedure, making it specific to human face datasets. Zhu et al. [2020] modify the general hybrid approach with an additional criterion, such that the recovered latent code must belong to the semantically meaningful region learned by the generator. It is thereby assumed that the real image can be reconstructed more faithfully in the immediate neighborhood of this initial guess. Alaluf et al. [2021] replace gradient-based optimization with an iterative encoder that encodes the current generation and target image to the estimated latent code difference. They empirically show that this iterative process converges and the recovered image improves over iterations. However, this method requires multiple forward passes in order to achieve a suitable latent code. In contrast to the work above, the inference module obtained by our method infers the latent code in one shot. Hence, it is much faster and does not run the risk of finding a non-meaningful latent code.

The inversion mechanisms presented so far do not directly influence the generative process. In most of the cases, they are conducted on a pre-trained frozen generator. Although in the case of ALI [Dumoulin et al., 2017] and BiGAN [Donahue et al., 2017], the inference model loosely interacts with the generative model at training time. However, the interaction is only indirect; *i.e.*,through the discriminator. In our work, we tightly couple the inference module with the generative module, resulting in better reconstruction quality.
Joint training of generator and inference model: We postulate that jointly training an inference module will help regularize GAN generators towards invertability. This is inspired by the difficulty of inverting a pre-trained high-performance GAN. For instance, Bau et al. [2019] invert PGAN [Karras et al., 2018], but for best results a two-stage mechanism is needed. Similarly, Image2StyleGAN [Abdal et al., 2019] projects real images into the extended w^+ space of StyleGAN, whereas, arguably, all the generated images can be generated from the more compact z or w space. This is further evident from Wulff and Torralba [2020] who find an intermediate latent space in StyleGAN that is more Gaussian than the assumed prior. However, they too use an optimization-based method and, hence, it is computationally expensive and at the same time specific to both the StyleGAN backend and the specific data set. Finally, we refer the readers to 'GAN Inversion: A Survey' [Xia et al., 2021] for a comprehensive review of related work.

6.3 Method

Goal: Our goal is to learn an inversion module alongside the generator during GAN training. Specifically, we find a generator $G : \mathbb{W} \rightarrow \mathbb{X}$ and an inference model $D : \mathbb{X} \rightarrow \mathbb{W}$ such that $x \approx G(D(x \sim \mathbb{X}))$. Where $\mathbb{X}$ denotes the data domain and $\mathbb{W}$ denotes the latent space and $\approx$ denotes the approximately equal operator. By doing so, we (1) unlock semantic editability of real images, (2) allow semi-supervised learning, (3) encourage latent space smoothness within the GAN framework. It is worth noting here that having access to the latent embeddings corresponding to the real images is the key factor that unlocks these functionalities. Since the GAN latent space is somewhat semantically meaningful (often also called disentangled), given an inference mechanism we can infer its latent representation and edit it, and decode back to the image domain to obtain meaningful changes in the pixel space. Furthermore, it enables semi supervised learning as follows. Given a dataset $\mathcal{X} := \{x_1, x_2 ... x_n\}$ of an independent variable and $\mathcal{Y} := \{y_1, y_2 ... y_m\}$, (with $m < n$) a set of corresponding dependent variable y_j, we can simply follow the usual semi-supervised learning scheme. That is, for an index $0 < j < m < n$ we train using the reconstruction and the supervision criterion while for $m < j < n$, we simply use the reconstruction criterion. Finally, a desirable characteristic of a generator as motivated in Sec. 3.2 in [Karras et al., 2020] is that a fixed length change in the latent representation should result into a fixed length change in the data domain. It is well-known, however, that the space of all images is not a Euclidean one and L_p norms are not good measures of distance. Therefore, it is hard to enforce such constraints in reality. With the inversion module (also referred to here as the inference module) available, we have access to different levels of features of the generated images. This helps us enforce such regularization.

6.3.1 Architecture

We demonstrate InvGAN using DC-GAN, BigGAN and StyleGAN as the underlying architectures. This shows that the benefits obtained by our method is not architecture dependent and is likely to return similar benefits in the future GAN models. Figure 6.1 represents the schematic of our model. We follow the traditional alternative generator-discriminator training mechanism. The generative part consists of three steps $z \sim \mathcal{N}(0, I); w = M(z); x = G(w)$, where M is a mapping network, G is the generator, D is the discriminator, and $\mathcal{N}(0, I)$ is the standard normal distribution. In the generator, we use the standard 8 fully connected layers (512 units in each layer) interleaved with Relu nonlinearities as the mapping network with StyleGAN and add a 2-layer (fully connected with Relu and 512 neurons each) mapping network to BigGAN and DC-GAN. The discriminator, besides outputting a real/fake score, also outputs an inferred w parameter. From here on, we use $\tilde{w}, c = D(x)$ to denote the inferred latent code ($\tilde{w}$) using the discriminator D and c to denote the real-fake classification decision for the sample $x \in \mathbb{X}$. We achieve this by adding an extra output head (a fully connected layer with appropriate number of neurons) to the penultimate layer, *i.e.*, the real/fake decision and the inferred latent code $(c, \tilde{w})$ are computed by two different fully connected layers that take as input the same intermediate tensor. Wherever obvious, we simply use $D(x)$ to refer to c, the discrimination decision only.

6.3.2 Objective

GAN Objective: Recall from Sec. 2.2.2 the objective function of vanilla GAN training is as follows.

$$\min_{G,M} \max_{D} \mathcal{L}_{\text{GAN}} = \min_{G,M} \max_{D} \left[\mathbb{E}_{x \in \mathbb{X}}[\log D(x)] + \mathbb{E}_{z \in \mathbb{Z}}[\log(1 - D(G(M(z))))]\right]. \tag{6.1}$$

A naive attempt at an approximately invertible GAN would perform $\min_G \max_D L_{\text{GAN}} + \min_D ||w - \tilde{w}||_p$, where $||\bullet||_p$ denotes an L_p norm. This loss function can be interpreted as optimal transport cost. We discuss this in more detail at the end of this section. However, this arrangement, coined the "naive model", does not yield satisfactory results, see Sec. 6.4.4. This can be attributed to the following factors: (1) w corresponding to real images are never seen by the generator; (2) no training signal is provided to the discriminator for inferring the latent code corresponding to real images (w_R); (3) the distribution of w_R might differ from prior distribution of w. We address each of these concerns with a specific design choice. Our naive model corresponds to adversarial autoencoders [Pidhorskyi et al., 2020] if the real-fake decision is derived from a common latent representation. However, this forces the encoding of real and generated images to be linearly separable and contributes to degraded inference performance.

Minimizing latent space extrapolation: Since, in the naive version, neither the generator nor the discriminator gets trained with w_R, it relies completely upon its extrapolation characteristics. In order to reduce the distribution mismatch for the generator, we draw half the mini batch of latent codes from the prior and the other half consists of w_R; *i.e.*,$w_{\text{total}} = w \mathbin{+\!\!+} w_R,\ w \sim P(W)$ where $\mathbin{+\!\!+}$ denotes a batch concatenation operation. By $w \sim P(W)$ we denote the two stage process given by the following $w = M(z \sim P(Z))$.

Pixel space reconstruction loss: Since latent codes for real images are not given, the discriminator cannot be trained directly. However, we recover a self-supervised training signal by allowing the gradients from the generator to flow into the discriminator. Intuitively, the discriminator tries to infer latent codes from real images that help the generator reproduce that image. Speaking from a practitioner's point of view, implementing this requires one to follow the following steps. i) Sample real images at random from the dataset $I_R \sim \mathcal{X}$, ii) Infer their latent representation by applying the discriminator model on them as $w_R, c = D(x)$, iii) Decode the inferred latent representations back to the image domain using the generator as $I_F = G(w_R)$, iv) Compare I_F and I_R using some loss function $\mathcal{L}(I_R, I_F)$. The choice of the loss function $\mathcal{L}$ impacts performance, and we will discuss this in detail in the following paragraph. As shown in Sec. 6.4.4, this helps improve real image inversion tremendously. As introduced to enforce consistency, one needs to imposing a reconstruction loss in the image domain between input and reconstructed real images. However, designing a meaningful loss function (distance function) between images is a non-trivial task. Ideally, we would like a feature extractor function f that extracts low- and high-level features from the image such that two images can be compared meaningfully. Given such a function, a reconstruction loss can be constructed as

$$\mathcal{L}_{\text{fm}} = \|\mathbb{E}_{x\in\mathbb{X}}\left(f(x) - f(G(w \sim P(W|x)))\right)\|_p \tag{6.2}$$

A common practice in the literature is to use a pre-trained VGG [Johnson et al., 2016, Zhang et al., 2018] network as a feature extractor f. However, it is well known that deep neural networks are susceptible to adversarial perturbations. Given this weakness, optimizing for perceptual loss is error-prone. Hence, a combination of a pixel-domain L_p and feature-space loss is typically used. This often results in degraded quality. Consequently, we take the discriminator itself as the feature extractor function f. Due to the min-max setting of GAN training, we are guaranteed to avoid the perils of adversarial and fooling samples. The feature loss is shown in the second half of Figure 6.1. Although this resembles the feature matching described by Salimans et al. [2016], it has a crucial difference. As seen in Eq. (6.2) the latent code fed into the generator is drawn from the conditional distribution $P(W|x) := \delta_{D(x)}(w)$ rather than the prior $P(W)$, where $\delta(x)$ represents the Dirac delta function located at x. This forces the distribution of the features to match more precisely, as compared to the simple first-moment matching proposed by Salimans et al. [2016].

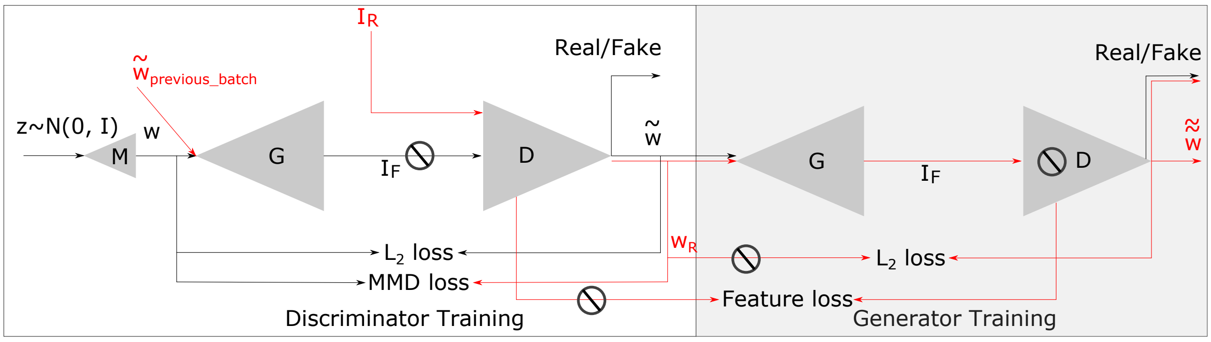


Figure 6.1: We train InvGAN following a regular GAN. We use a second output head in the discriminator besides the real fake decision head, to infer the latent-code z of a given image. Here ⃠ denotes no gradient propagation during the back propagation step. It also denotes 'no training' when it is placed on a model. We use red to show data flow corresponding to real images.

Addressing mismatch between prior and posterior: Finally, we address the possibility of mismatch between inferred and prior latent distributions (point (3) described above), by imposing a maximum mean discrepancy (MMD) loss between the sets of samples of the said two distributions. We use an RBF kernel to compute this loss. This loss improves the random sampling quality by providing a direct learning signal to the mapping network.

Finally, we summarize the objective of our complete model in Eq. (6.3). Here and in the rest of the chapter we use a plus operator $+$, between two optimization process, to indicate that both of them are performed simultaneously.

$$\min_{G,M}\left[\max_{D}\mathcal{L}_{\text{GAN}}+\min_{D}\left[\mathbb{E}_{w+w_R}\left[\|M(z)-\tilde{w}\|_2^2+\mathcal{L}_{\text{fm}}+\left\|\tilde{w}-\tilde{\tilde{w}}\right\|_2^2+\text{MMD}\{w,w_R\}\right]\right]\right] \tag{6.3}$$

We summarize the training process in the following pseudocode Alg. 1

An optimal transport based interpretation: Neglecting the last three terms in Eq. (6.3), our method can be interpreted as a Wasserstein autoencoder-GAN (WAE-GAN) [Tolstikhin et al., 2018]. Considering a WAE with its data domain set to our latent space and its latent space assigned to our image domain, if the encoder and the discriminator share weights the analogy is complete. Our model can, hence, be thought of as learning the latent variable model $P(W)$ by randomly sampling a data point $x \sim \mathbb{X}$ from the training set and mapping it to a latent code w via a deterministic transformation. In terms of density, it can be written as in Eq. (6.4).

$$P(W) := \int_{x\in\mathbb{X}} P(w|x)P(x)\mathrm{d}x. \tag{6.4}$$

As proven by Bousquet et al. [2017], under this model the optimal transport problem $W_c(P(W),P_D(W)) := \inf_{\Gamma\in P(w_1\sim P(W),w_2\sim P_D(W))}\left[\mathbb{E}_{w_1,w_2\sim\Gamma}\left[c(w_1,w_2)\right]\right]$ can be solved by finding a generative model $G(X|W)$ such that its X marginal, $P_G(X) = \mathbb{E}_{w\sim P(W)}G(X|w)$ matches the image distribution $P(X)$. We ensure this by considering the Jensen–Shannon divergence $D_{\text{JS}}(P_G(X),P(X))$ using a GAN framework. Now, if we choose the ground cost function $c(w_1,w_2)$ to be squared L_2 norm the cost function given in Eq. (6.5).

```
Data: A set of images $\mathcal{X}$ (e.g., FFHQ, CELEBA etc.)
Result: A trained invertible generator using GAN framework
initialize $M, G, D$;
while Reconstructed, interpolated and generated image FID is not satisfactory do
    Sample noise $Z \sim \mathcal{N}(0, I)$ and real images $I_R \sim \mathcal{X}$;
    $w = M(z)$; $w_{total} = Concat(w, \tilde{w}_{previous\_batch}, axis = 0)$;
    Generate fake images $I_F = G(w_{total})$;
    $\tilde{w}, c = D(I_F)$; $w_R, c_R = D(I_R)$;;
    Compute $||w - \tilde{w}||_2$; $MMD(w_R, w)$; $\mathcal{L}(c, I(Generator))$; $\mathcal{L}(c_R, 1)$;
    Recosntruct real images $I_{FR} = G(w_R)$;
    Compute reconstruction loss $\mathcal{L}_{rec}(I_R, I_{FR})$;
    Re-infer $w_R$ as $\tilde{\tilde{w}}$; $\tilde{\tilde{c}} = D(I_{FR})$;
    Compute $||w_R - \tilde{\tilde{w}}||$; $\mathcal{L}(\tilde{\tilde{c}}, I(Generator))$;
    Back propagate all losses appropriately
end
```

Algorithm 1: Pseudocode for InvGAN. This algorithm outlines steps required to train InvGAN to reproduce results shown here. Where $I(Generator)$ is an indicator function that returns 1 if the loss function $\mathcal{L}$ (usually BCE, but we use the non saturating version of it, as is common in GAN training) is computed to update the generator; 0 otherwise.

$$\min_{G,M}\max_{D}\mathcal{L}_{\text{GAN}} + \min_{G,M}\min_{D}||w - \tilde{w}||_2^2 \tag{6.5}$$

Finally, we find that by running the encoding/decoding cycle one more time, we can impose several constraints that improve the quality of the encoder and the decoder network in practice. This leads to our full optimization criterion, as described in Eq. (6.3).

6.3.3 Dealing with resolutions higher than training resolution

Although StyleGAN [Karras et al., 2020] and BigGAN [Brock et al., 2018] have shown that it is possible to generate relatively high-resolution images, in the range of 1024×1024 and 512×512, their training is resource intense and the models are difficult to tune for new data sets. Equipped with invertibility, we explore a tiling strategy to improve output resolution. First, we train an invertible GAN at a lower resolution ($m \times m$) and simply tile them $n \times n$ times with n^2 latent codes to obtain a higher resolution ($mn \times mn$) final output image. The representation thus obtained can be used to accomplish various tasks as outlined in Sec. 6.4.3. The reconstruction results are visualized in Figure 6.5. This process correlates in spirit somewhat to COCO-GAN [Lin et al., 2019]. The main difference, however, is that our model at no point learns to assemble neighboring patches. Indeed, the seams are visible if one squints at the generated images, *e.g.*, in figure 6.5. However, a detailed study of tiling for generation of higher resolution images than the input domain is beyond the scope of this thesis. We simply explore some naive settings and their applications in sec. 6.4.3.

6.4 Experiments

We test InvGAN on several diverse datasets (MNIST [LeCun et al., 1998b], ImageNet [Deng et al., 2009], CelebA [Liu et al., 2015], FFHQ [Karras et al., 2019a]) and multiple backbone architectures (DC-GAN, BigGAN, StyleGAN). Our method is evaluated both qualitatively (via style mixing, image inpainting etc.) and quantitatively (via the FID score and the suitability for data augmentation for discriminative tasks such as classification). Figure 6.2 shows reconstruction of CelebA faces.

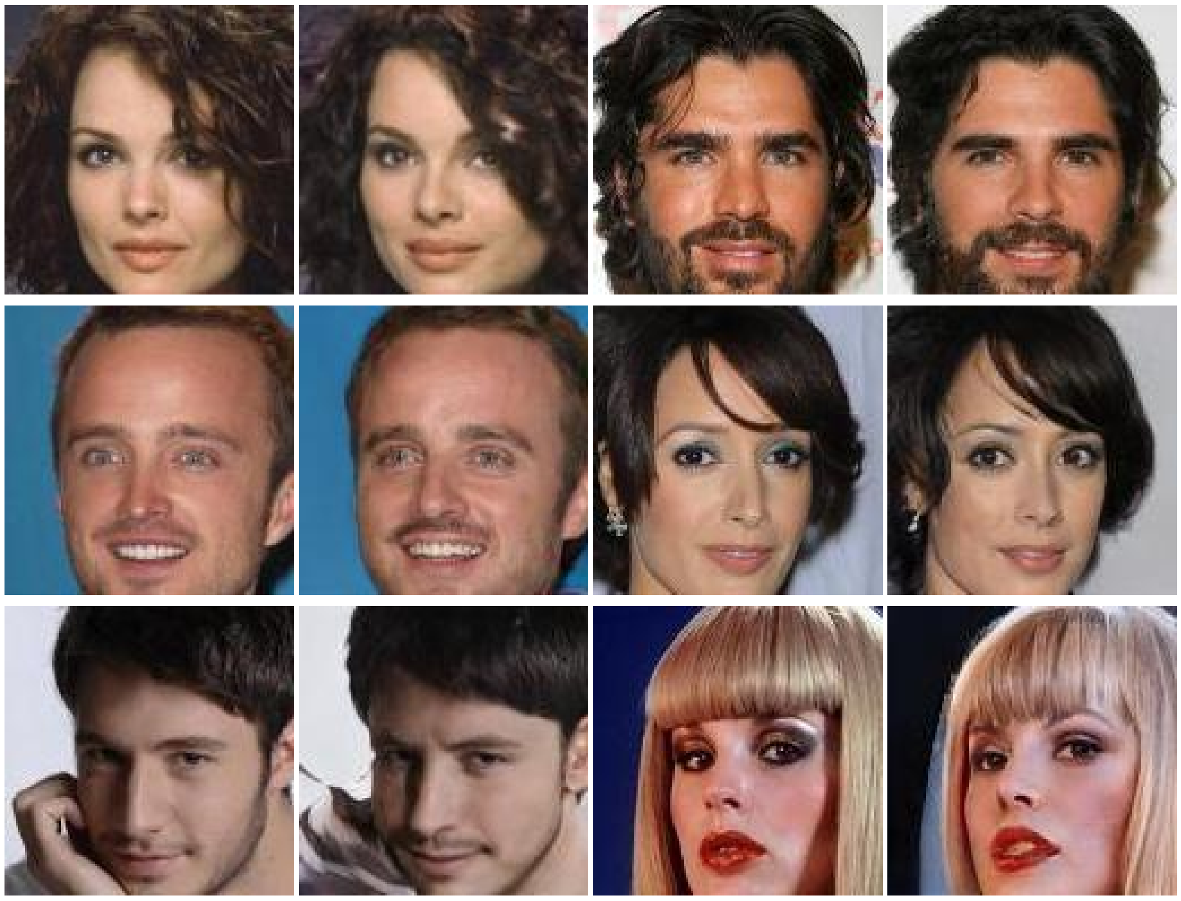


Figure 6.2: InvGAN reconstructions of CelebA at 128 × 128 resolution. Alternating (from left to right) original and reconstructed images. We show more such examples in Appendix C

Table 6.1 shows random sample FIDs, middle point linear interpolation FIDs and test set reconstruction mean absolute errors (MAEs) of our generative model. We intend to provide a definition, baseline, and understanding of inversion of a high-quality generator. Specifically, we highlight model-based inversion, joint training of generative and inference model and its usability in downstream tasks. We demonstrate that our method generalizes across architectures, datasets, and types of downstream task.
Training data and tasks: We start with a StyleGAN-based architecture on FFHQ and CelebA for image editing. Then we train a BigGAN-based architecture on ImageNet, and show super resolution and video key-framing by tiling in the latent domain to work with images and videos that have higher resolution than the training data. We also show ablation studies with a DC-GAN-based architecture on MNIST. This variety of architectures, datasets, and tasks provide

Models	RandFID	RandRecFID	TsRecFID	IntTsFID	MAE±1	Run Time
FFHQ [Xu et al., 2021]	49.65/14.59	56.71/23.93	-/13.73	68.45/38.01	0.129	0.045
FFHQ Enc. [Zhu et al., 2020]	46.82/14.38	-/-	88.48 / -	-/-	0.460	
FFHQ MSE opt. [Zhu et al., 2020]	46.82/14.38	-/-	58.04 / -	-/-	0.106	
FFHQ In-D. Inv. [Zhu et al., 2020]	46.82/14.38	52.02/-	42.64 / -	71.83/-	0.115	99.76
DCGAN, MNIST	17.44/6.10	16.76/4.25	17.77/4.70	26.04/11.44	0.070	$3.3 \cdot 10^{-5}$
StyleGAN, CelebA	26.63/4.81	24.35/3.51	24.37/4.14	32.37/15.60	0.150	$1.0 \cdot 10^{-3}$
StyleGAN, FFHQ	49.14/12.12	44.42/8.85	41.14/7.15	49.52/14.36	0.255	$2.0 \cdot 10^{-3}$

Table 6.1: Here we report random sample FID (RandFID), FID of reconstructed random samples (RandRecFID), FID of reconstructed test set samples (TsRecFID), FID of the linear middle interpolation of test set images (IntTsFID) and reconstruction per pixel per color channel mean absolute error when images are normalized between ±1, also from test set. The last three rows are our models. All FID scores are here evaluated against the training set using 500 and, 50000 samples. They are separated by '/'. For the traditional MSE optimization based and In-Domain GAN inversion, the MSE errors are converted to MAE by taking square root and averaging over the color channels and accounting for the re-normalization of pixel values between ±1. Runtime is given in seconds per image. We ran them on a V100 32GB GPU and measured wall clock time.

insights into the method and its generality. In the following sections we evaluate qualitatively by visualizing semantic editing of real images and quantitatively on various downstream tasks including classification fairness, image super resolution, image mixing, etc.

6.4.1 Semantically consistent inversion using InvGAN

GANs can be used to augment training data and substantially improve downstream task learning. Improving fairness of classifiers on human face images is a prominent example [Sharmanska et al., 2020, Sattigeri et al., 2018, Balakrishnan et al., 2020, Ramaswamy et al., 2021]. There is an important shortcoming in using existing GAN approaches for such tasks: the labeling of augmented data relies on methods that are trained independently on the original data set, using human annotators or compute-expensive optimization-based inversion. This is due to the fact that most generative models used are unconditional and so generate unlabeled synthetic data. A typical example is data-set de-biasing by Ramaswamy et al. [2021]. For each training image, an altered example that differs in some attribute (*e.g.*, age, hair color, *etc*) has to be generated. This has previously been done in one of two ways, *e.g.*, by finding the latent representation of the ground truth image via optimization or by labeling random samples using pre-trained classifiers on the biased data set. Optimization-based methods are slow and not a viable option for on-demand/adaptive data augmentation. Methods using pre-trained classifiers inherit their flaws, like correlation induced dependencies.

Here we focus on the subproblem of reliably encoding face images to the latent space in a semantically consistent manner using InvGAN. For this, we train ResNet50 attribute classifiers on the CelebA dataset. We validate that the encoding and decoding of InvGAN results in a semantically consistent reconstruction by training a classifier network only on reconstructions of the full training set. As a baseline, we use the same classifier trained on the original CelebA training set. We produce two reconstructed training sets 1. by using the tiling-based inversion

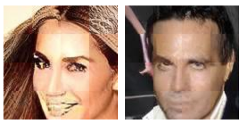
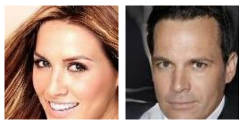
Train on → **Eval. on ↓**	**Original**	**Tile Recon.**	**Full Recon.**
Original	0.81 ± 0.15	0.77 ± 0.16	0.79 ± 0.15
Tile recon.	0.79 ± 0.16	0.80 ± 0.15	0.78 ± 0.16
Full recon.	0.81 ± 0.15	0.78 ± 0.16	0.81 ± 0.14
Recon. Vis.			

Table 6.2: Mean average precision for a ResNet50 attribute classifier on CelebA, averaged over 20 attributes. We report the performance for training on the original dataset, the reconstructed dataset using the tiling-based method pre-trained on ImageNet and the reconstruction on InvGAN trained on the CelebA training set directly.

(trained on ImageNet) and 2. by training InvGAN on CelebA (without tiling). We elaborate on the tiling-based inversion mechanism in Sec. 6.4.3. For each attribute, a separate classifier has been trained for 20 epochs. The resulting mean average precisions are reported in Table 6.2. We see that training on the reconstructions allows for very good domain transfer to real images, indicating that the reconstruction process maintained the semantics of the images.

6.4.2 Suitability for image editing

GAN inversion methods have been proposed for machine supported photo editing tasks [Zhu et al., 2020, Cheng et al., 2020, Perarnau et al., 2016]. Although there is hardly any quantitative evaluation for the suitability of a specific inversion algorithm or model, a variety of representative operations have been reported [Abdal et al., 2020, 2019, Zhu et al., 2020]. Among those are in-painting cut out regions of an image, image-merging and improving on amateurish manual photo editing. Figures 6.3 visualizes those operations performed on FFHQ and CelebaA images, respectively. We present more qualitative examples in C.3 in the appendix. We demonstrate in-painting by zeroing out a randomly positioned square patch and then simply reconstructing the image. This can be interpreted as an image-repair operation or correcting imperfections in unseen data. The image-merging is performed by reconstructing an image which is composed out of two images by simply placing them together. By reconstructing an image that has undergone manual photo editing, higher degrees of photo-realism are achieved. Quantitative metrics for such tasks are hard to define and hence are scarcely found in prior art, since they depend upon visual quality of the results. We report reconstruction and interpolation FIDs in Table 6.1, in an effort to establish a baseline for future research. However, we do acknowledge that a boost in pixel fidelity in our reconstruction will greatly boost the performance of InvGAN on photo editing tasks. The experiments clearly show the general suitability of the learned representations to project out of distribution images to the learned posterior manifold via reconstruction.

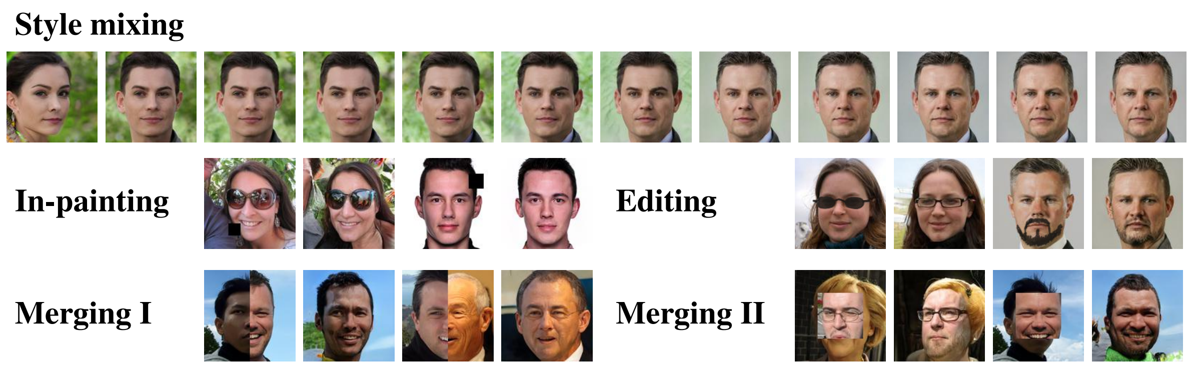


Figure 6.3: Benchmark image editing tasks on FFHQ (128 px). Style mixing: We transfer the first $0, 1, 2, \dots, 11$ style vectors from one image to another. For the other image editing tasks, pairs of images are input image (left) and reconstruction (right).

6.4.3 Tiling to boost resolution

Memory limitations and instability of high resolution GANs are prominent obstacles in generative model training. One way to bypass such difficulties is to generate the image in parts. Here we train our invertible generative model, a BigGAN architecture, on 32×32 random patches from ImageNet. Once the inversion mechanism and the generator are trained to satisfactory quality, we reconstruct both FFHQ and ImageNet images. We use 256×256 resolution and tiling 64 patches in an 8×8 grid for FFHQ images, and 128×128 resolution and tiling 16 patches in a 4×4 grid for ImageNet images. The reconstruction results are shown in Figure 6.5. Given the successful reconstruction process, we explore the tiled latent space for tasks such as image deblurring and time interpolation of video frames.
Image de-blurring: Here we take a low-resolution image, scale it to the intended resolution using bicubic interpolation, invert it patch by patch, Gaussian blur it, invert it again and linearly extrapolate it in the deblurring direction. The deblurring direction is simply obtained by subtracting the latent code of the given low resolution but bicubic up sampled image from the latent code of the blurred version of it at the same resolution. The exact amount of extrapolation desired is left up to the user. In Figure 6.4 we show the effect of 3 different levels of extrapolation. Although our method is not trained for the task of super resolution, by virtue of a meaningful latent space we can enhance image quality.

6.4.4 Ablation studies

Recall that the naive model defined in Sec. 6.3.2 uses the following optimization $\min_{G,M}\left[\max_D L_{\text{GAN}} + \min_D \mathbb{E}_{z\sim P(Z)} \|M(z) - \tilde{w}\|_p\right]$ to train (also given in Eq. (6.5)), (results in Figure 6.6a). Here we progressively show how our three main components influence the naive model. As is apparent from Sec. 6.3, the first major improvement comes from exposing the generator to the latent code inferred from real images. This is primarily due to the difference in the prior and the induced posterior distribution. This is especially true during early training, which imparts a lasting impact. The corresponding optimization is $\min_{G,M}\left[\max_D L_{\text{GAN}} + \min_D \mathbb{E}_{w=M(z\sim P(Z))+\!\!\!+ w_R} \|w - \tilde{w}\|_p\right]$. Simply reducing the distribution mismatch between prior

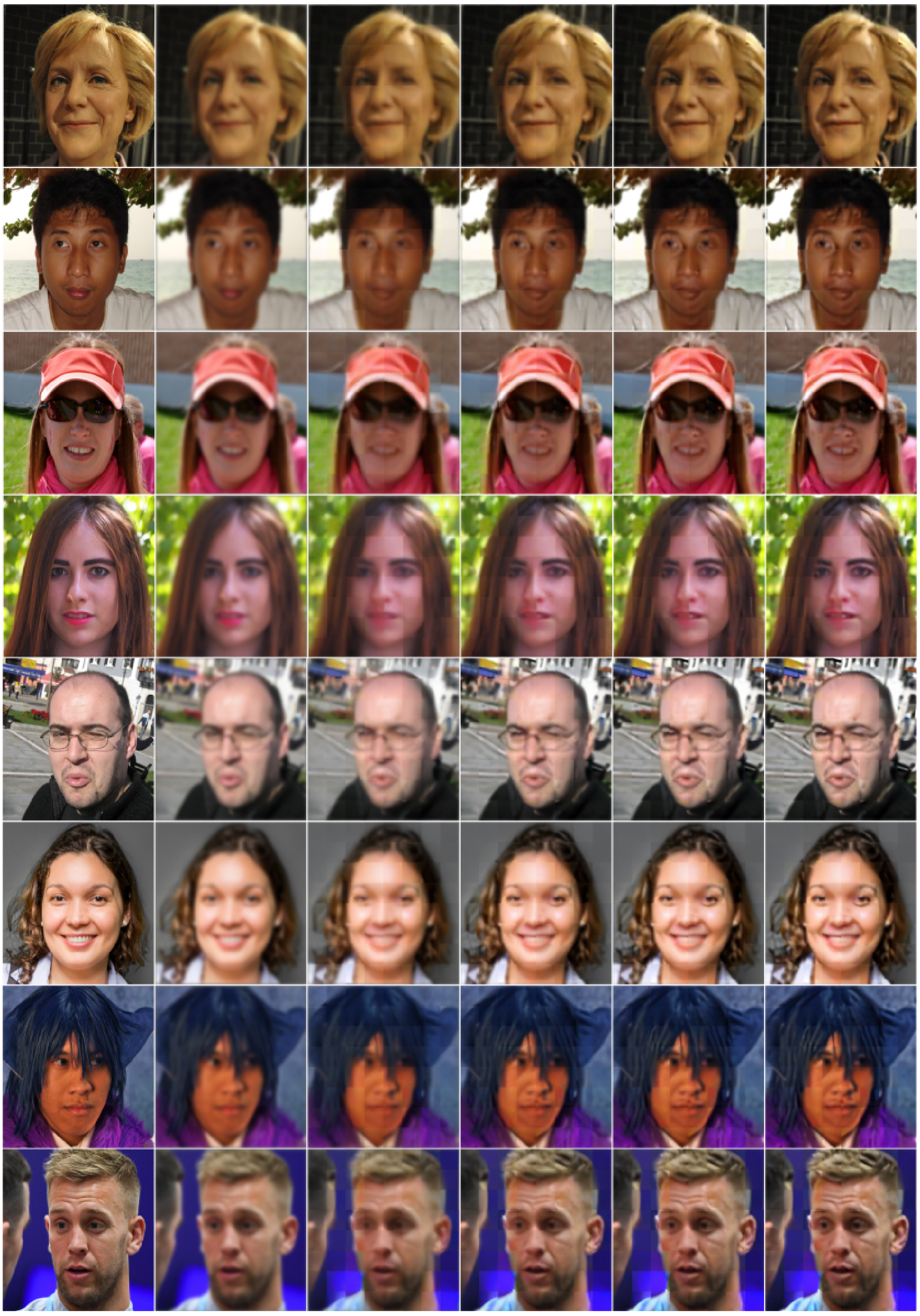
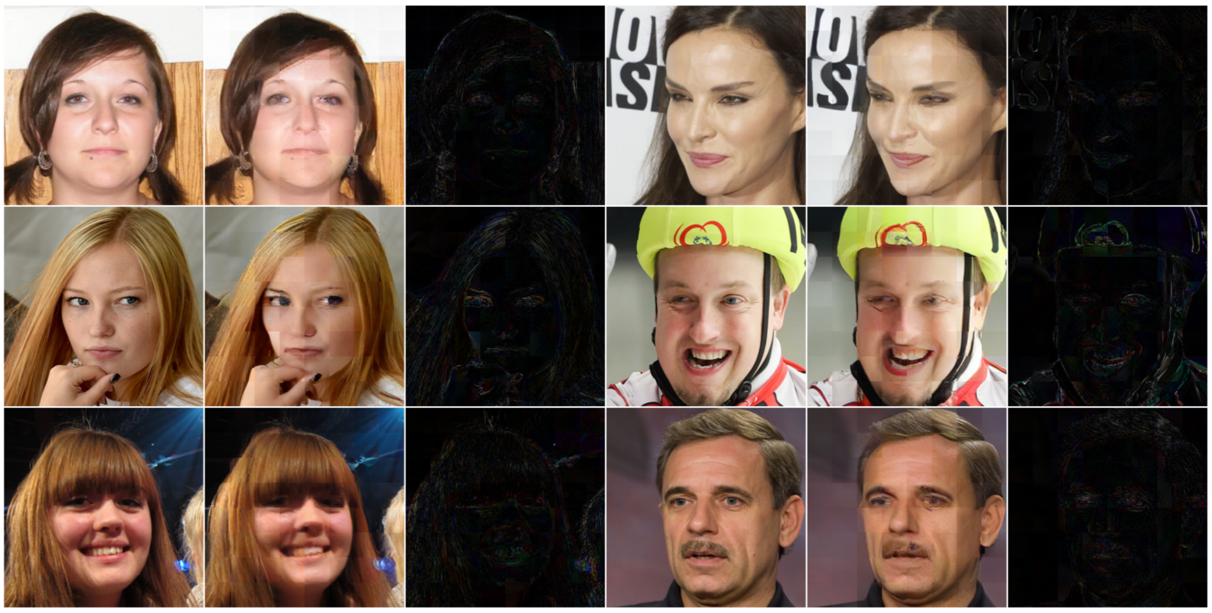
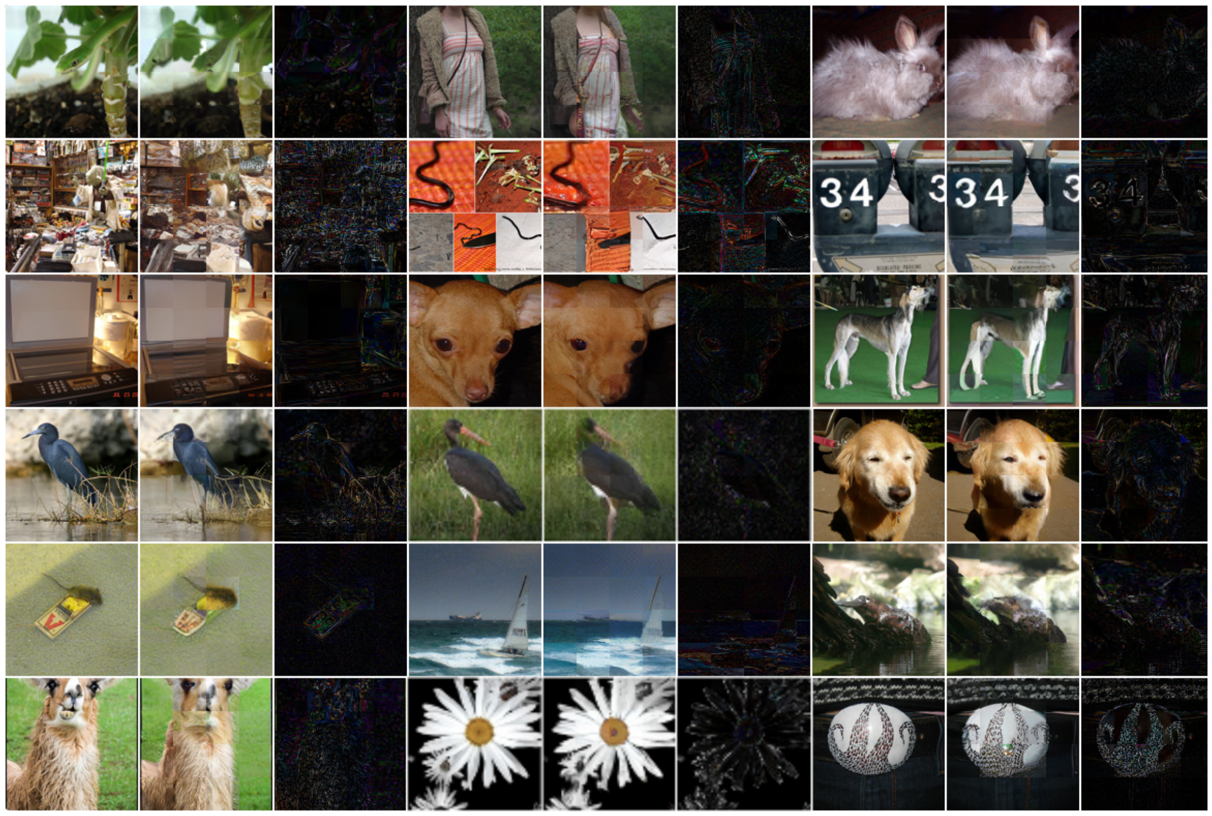


Figure 6.4: Super resolution using extrapolation in the tiled latent space. From left, we visualize the original image, the low-resolution version of it, the reconstruction of the low-resolution version, and progressive extrapolation to achieve deblurring.



(a)



(b)

Figure 6.5: Tiled reconstruction of random (a) FFHQ Images and (b) ImageNet images. The left column shows the real images, the second shows the patch by patch reconstructions, and the third shows the absolute pixel-wise differences. Note that interestingly, though the patches are reconstructed independently of each other, the errors lie mostly on the edges of the objects in the images, arguably the most information dense region of the images.

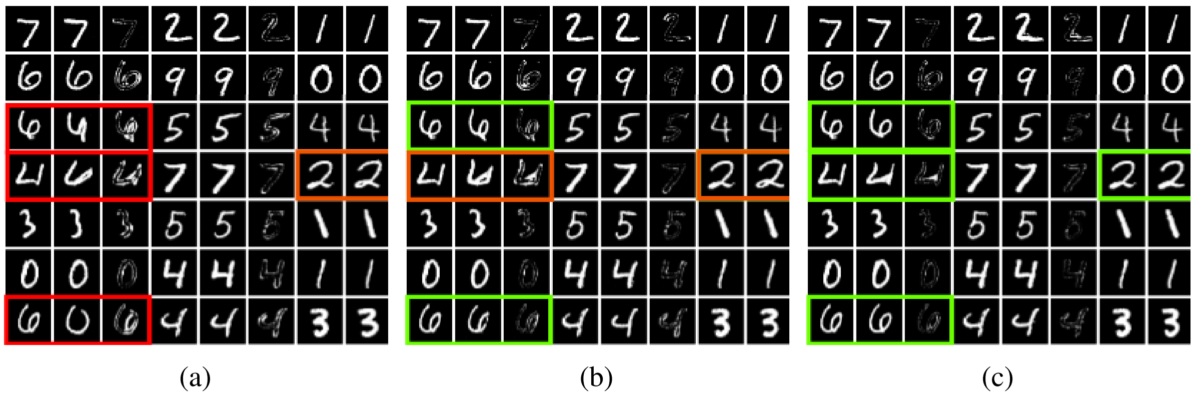


Figure 6.6: Inversion of held out test samples. Columns are in groups of three: the first column holds real images, second their reconstruction and third the absolute pixel-wise difference. (a) Inversion using naive model, *i.e.*,only z–reconstruction loss is used, (b) inversion using model that uses latent codes from real samples, *i.e.*,the augmented naive model. (c) our full model. Notice how the imperfections in the reconstructions highlighted with red boxes gradually turns green as the model improves.

and posterior by injecting inferred latent codes improves inversion quality. This is visualized in Figure 6.6b. We shall call this model the augmented naive model. However, this modification unlocks the possibility to enforce back propagation of the generator loss gradients to the discriminator. Moreover, one can exploit the real-generated image pairing that arises (see Sec. 6.3.2). This leads to our full model and the results are visualized in Figure 6.6c

6.4.5 Discussion and future work

While InvGAN can reliably invert the generator of a GAN, it still can benefit from an improved reconstruction fidelity for tasks such as, image compression, image segmentation, etc. We observe that the reconstruction of rare features, such as hands, microphones, hats, hair strands or background features, tend to have lower reconstruction fidelity, as seen in the bottom row first and second column in Figure 6.2 (more examples in bottom row 3rd and 4th column of Fig. C.2, in Appendix C). This, combined with the fact that the reconstruction loss during training tends to saturate even when the weights are sufficiently high, indicates that even well-engineered architectures such as StyleGAN and BigGAN lack the representative power to provide sufficient data coverage.

Strong inductive biases in the generative model have the potential to improve the quality of the inference module. For instance, GIF [Chap. 5.3], and hologan [Nguyen-Phuoc et al., 2019] among others introduce strong inductive bias from the underlying 3D geometry and lighting properties of a 2D image. Hence, an inverse module of these generative mechanism has the potential to produce a high-quality inference model.

As is shown by the success of RAEs [Chap. 3], there is often a mismatch between the induced posterior and the prior of generative models, that can be removed by an ex-post density estimator. InvGAN is also amenable to ex-post density estimation. When applied to the tiled latent codes, it estimates a joint density of the tiles for unseen data. This would recover a generative model without going through the unstable GAN training.

We have shown that our method scales to large datasets such as ImageNet, CelebA, and FFHQ. A future work that is able to improve upon reconstruction fidelity, would be able to explore adversarial robustness by extending [Ghosh et al., 2019] to larger datasets.

6.4.6 Conclusion

We have presented InvGAN, a model-based inference framework for the latent parameters used by a GAN generator. InvGAN enjoys several advantages compared to state-of-the-art inversion mechanisms. The inversion mechanism is integrated into the training phase of the generator. This discourages mode collapse. However, it was observed that infrequent features suffer worse reconstruction fidelity as compared to more frequent features. This leads us to hypothesize that even the carefully designed architectures, like StyleGAN and BigGAN, lack representation power. Beyond the computational advantage of model-based inversion, our mechanism can reconstruct images that are larger than the training images by tiling with no additional post-processing step such as merging. We further have demonstrated that the inferred latent code for a given image is semantically meaningful, *i.e.*, it falls inside the structured part of the latent space learned by the generator similar to in-domain GAN inversion [Zhu et al., 2020].

Chapter 7

Conclusion

The main theme running through this thesis is the problem of identifying and gaining task-specific control over deep generative models. Unsupervised learning has been instrumental to many developments in machine learning. The latest edition of this is generative models. The key hypothesis here is that if a model can generate new data that belongs to the population of its training data, it 'understands' what it means to be part of this (training data) population. Recently, the field has been rather successful at this. We have created models that can generate excellent new members that belong to its training distribution. However, the 'understanding' of the model about its training data still can not be trivially utilized, *e.g.*, there is no obvious way of using the GAN latent space for downstream tasks (*e.g.*, regression, classification, clustering *etc*), except with using an inference mechanism. Besides, there are many applications that require precision control in different aspects of the generation process *e.g.*, in computer graphics related applications it is necessary to have control over different semantic aspects of the generated images. This remains a hard task for modern generative models. These observations have motivated us to study VAEs and GANs and gain control over their latent structure, conditional fidelity and build an inference model.

7.1 Contributions

7.1.1 RAE

- We introduce the RAE framework for generative modelling as a drop-in replacement for any VAE architecture;
- We propose an ex-post density estimation scheme that greatly improves sample quality for VAEs, WAEs and RAEs without the need to retrain the models;
- We conduct a rigorous empirical evaluation to compare RAEs with VAEs and several baselines on standard image datasets and on more challenging structured domains such as molecule generation [Kusner et al., 2017, Gómez-Bombarelli et al., 2018].

7.1.2 GMM-VAE

- We show how VAE's can be trained with labelled data (class labels), using a Gaussian mixture a prior distribution. This naturally gives us a classifier.

- We perform selective classification using this framework, thereby rejecting adversarial and fooling samples.
- We propose a method to learn a classifier in a semi-supervised scenario using the same framework, and show that this classifier is also resistant to adversarial attacks.
- We also show how the detected adversarial samples can be reclassified into the correct class by iterative optimization.
- We verify our claims through experimentation on three publicly available data-sets: MNIST [LeCun et al., 1998a], SVHN [Netzer et al., 2011] and COIL-100 [Nayar et al., 1996].

7.1.3 GIF

- We provide a 2D generative model with predetermined control. Specifically GIF is a generative 2D face model with FLAME [Li et al., 2017] control,
- By conditioning on images rendered from FLAME, GIF incorporates precise information about 3D face structure,
- By leveraging texture consistency constraints, GIF disentangles between different parameters and unconditioned factors, *e.g.*, lighting, and geometry.
- We verify GIF's effectiveness using a large-scale perceptual study and using state-of-the-art automated metics.

7.1.4 InvGAN

- We extend the GAN framework to include an inference model that embeds real images into the latent space.
- InvGAN can be used in a variety of computer vision applications *e.g.*, data augmentation, image inpainting, spatio-temporal super resolution, image harmonization *etc*.
- By including an inversion mechanism into GAN training, we can interpolate between real images.
- Using a tiling mechanism, we can effectively work with larger image size than the images from the training set.

7.2 Open Questions

This thesis has advanced the state of the art of generative models by improving access to the 'knowledge' extracted by such models and by improved control over their learning process, however a number of future research directions open up that deserve further study.

7.2.1 Likelihood-based vs adversarial models

VAEs, Flow-based, and Diffusion-based methods are the main likelihood based generative modelling methods. While Flow-based and diffusion-based methods achieve excellent data fidelity, *e.g.*, achieve excellent generated image sharpness when trained on image data, they do not compress their training data. This is problematic, both conceptually and in practice. Conceptually, we expect images to lie in a much lower dimensional space as compared to their pixel representation. Therefore, it is unsatisfactory that these models do not attempt to discover this underlying manifold. Since images, especially high-resolution ones, are rather large and unwieldy, practically, flow- and diffusion-based models tend to be large and hard to use. It is also problematic to use their representations for downstream tasks. This leaves us with VAEs from the family of likelihood-based methods for further examination. Even though modern VAE-based methods such as VQ-VAE by van den Oord et al. [2017] or VD-VAE by Child [2021] claim to have achieved high-quality generation, in practice these methods require large latent space dimensionality, offering only modest compression. Adversarial methods (GANs), however, do not suffer from any of the above discussed issues, proving that there exists room for improvement for the likelihood-based methods. In this thesis, we have dedicated Chapter 3 to address the mismatch between posterior and prior distributions in VAEs. We found that the contribution of the latent prior towards generation quality is limited. This provides further evidence that the likelihood model of a VAE is the main contributor of this drawback.

7.2.2 Quantifying generative capacity

One of the critical questions that we can not answer yet is whether a GAN truly can cover the whole span of its training distribution, or if it only covers part of it such that its generation 'looks' similar to the training population but in fact lacks diversity. This is partially addressed by FID scores [Heusel et al., 2017a] and precision-recall score [Sajjadi et al., 2018], but they rely on an arbitrary feature extractor, clustering mechanisms, and estimates of high dimensional distribution divergence. Our InvGAN framework partially sheds light on this direction, in the sense that if one is able to embed unseen real samples meaningfully into the latent space, then we can be sure that the GAN framework has found the true manifold to data mapping. However, if this method fails or does not perform satisfactorily, then we do not gain any insight. Hence, characterization of the generative capacity of GANs is still an open problem.

7.2.3 Training Instability of GANs

Although recently much progress has been made in the direction of stabilizing GAN training, any practitioner knows that it is a much harder job as compared to training a classifier for example. The min-max training process at the heart of a GAN is widely regarded to be the problem. However, a comprehensive fix is still an open problem.

7.3 Condition cross-talk

We introduced this term in Sec. 5.3.4. This, in short, is when a conditional generative model with multiple conditioning factors finds spurious correlation, *e.g.*, because of incomplete data, and generates features belonging to one condition depending upon the other. So far, there is no generalized study in this direction, although we address this in a problem-specific scenario, *i.e.*,in the context of human face image generation. A more in-depth general case study is necessary.

7.4 Modular Generative models

Although it has been shown that scale-specific aspects of training data are captured by recent generative frameworks [Karras et al., 2019a], for practical use we have to find a mechanism that yields a more precise and predictable hierarchical structure. For example, given a data set of images where n objects plus a background occur at random, we should be able to learn n separate generators for each object in addition to a background generator. Furthermore, these object generators should be conditional, *i.e.*, given a random image these generators should be able to place its object in the context of the given image. We discuss this in more detail in Sec. 7.5.2

7.5 Future Directions

7.5.1 GAN as an Energy-Based Model

Since GANs were first developed from a game-theoretic perspective, they seem to be at odds with likelihood-based density estimators and generators. However, it has recently been claimed that GANs do have an energy-based interpretation [Che et al., 2020]. How to tackle training instability and how to avoid the mode drop problem of GANs, as highlighted in Sec. 7.2.3, still remains an open problem. Here, we showcase an energy-based interpretation of GANs, that can address both. Additionally, it predicts gradient penalty regularization of the discriminator, a widely practiced regularizer found empirically. Given a data set $x_i \in \mathcal{X} : i \in \{1...N\}$, we wish to find a function $D_\phi : \mathcal{X} \to \mathbb{R}^+$ such that $D_\phi(x_i)$ gives the probability density corresponding to the sample x_i. Further, let us assume that we wish to optimize for the parameters ϕ of D_ϕ by maximizing the log-likelihood of the data set. Concretely, we follow Eq. (7.1).

$$\text{argmax} \sum_{i=0}^{N} \log D_\phi(x_i) \text{ ; s.t. } D_\phi(x_i) \geq 0 \text{ and } \int_X D_\phi(x)dx = 1 \tag{7.1}$$

Here X denotes the vector space that x_i-s belong to and the integral is w.r.t. the Lebesgue measure of that space. The constraints simply follow from the requirement of density functions. Let us further assume, D_ϕ is represented by a neural network. Now the first criterion, *i.e.*, the density of a point cannot be negative, can easily be enforced in practice with a neural network by considering the last activation function to be an exponential function. However, the second constraint that the integral of the hypothesized density over the domain of it must be 1 is much harder to enforce. Since no analytical expression for it exists, once commonly uses a numerical approximation. Specifically, we intend to use the Riemann sum $\int_X D_\phi(x)dx \approx \sum_{j=0}^{m-1} D_\phi(x^j)\mu(x^{j+1}, x^J)$, where $x^0...x^m$ is a partition of the domain X and $\mu(x^{j+1}, x^j)$ is the Lebesgue measure of the region in the partition defined between x^{j+1} and x^j.

Note that x^j denotes simply the j-th partition. Loosely speaking, the partitions can be thought of as hyper cubes and the partition point $x^j \forall j \in \{0,1,...,m\}$ can be thought of as the left-top and right-bottom corner points. Finally, then $\mu(x^{j+1},x^i)$ would be the hyper volume of a cube defined by the corner points x^{j+1} and x^j. Given this, Eq. (7.1) can now be simplified to Eq. (7.2)

$$\begin{aligned}
&\operatorname{argmax}\sum_{i=0}^{N}\log\frac{D_\phi(x_i)}{\int_X D_\phi(x)dx} \approx \operatorname{argmax}\sum_{i=0}^{N}\log\frac{D_\phi(x_i)}{\sum_{j=0}^{m-1} D_\phi(x^j)\mu(x^{j+1},x^j)} \\
&= \operatorname{argmax}\sum_{i=0}^{N}\left\{\log D_\phi(x_i) - \log\sum_{j=0}^{m-1} D_\phi(x^j)c^j\right\}
\end{aligned} \tag{7.2}$$

Here, $c^j = \mu(x^{j+1},x^j)$ and if we further assume $c^j = c\ ;\ \forall j$, then we can drop the constant hyper volume from the maximization operation. One practical problem with the objective given by Eq. (7.2) is that we need to come up with a partition $x^0...x^m$. If X is high dimensional, we need large m to approximate the integral. Furthermore, real images come from a manifold embedded in the space they are represented. Therefore, $D_\phi(x)$ is expected to be zero almost everywhere. This prompts for sophisticated choice for x^j-s. Let us imagine $x^j = G_\theta(z^j) : \mathbb{R}^d \to \mathbb{R}^D\ ;\ z^j \in \mathcal{N}(0,I)$, is a function that gives us x-s that are 'close' to real images, when fed with samples from a normally distributed random variable Z. Here d is the latent dimension and D is the data dimension. Let us also call G_θ, the generator. Further, assuming that $D_\phi(x) = 0$ for all data points that are outside the range of $G_\theta(z)$. The integral sum of Eq. (7.2) can now be expressed much more efficiently as $\sum_{j=0}^{m-1} D_\phi(x^j) = \sum_{j=0}^{m-1} D_\phi(G_\theta(z^j))$. Plugging this into Eq. (7.2), we can retrieve the objective for the discriminator of a GAN, if we set $m = 1$. This is given by Eq. (7.3).

$$\operatorname{argmax}\sum_{i=0}^{N}\left\{\log D_\phi(x_i) - \log\sum_{j=0}^{m-1} D_\phi(G_\theta(z_j \sim \mathcal{N}(0,I)))\right\} \tag{7.3}$$

A desirable characteristic of this approach is that it decouples the objective for our generator and discriminator. Hence, the famous or rather infamous min-max training instability can be avoided. If we compare this to the original GAN objective as in Eq. (6.1), we see a slight difference. Namely, instead of maximizing $1 - log\left(D_\phi\left(G_\theta(Z)\right)\right)$, we are minimizing $log\left(D_\phi\left(G_\theta(Z)\right)\right)$. When Goodfellow et al. [2014] introduced GANs, the authors motivated this shift from their theory using the point of view of saturating loss and vanishing gradients. Here, Eq. (7.3) explains this ad hoc deviation from a likelihood maximization perspective. Furthermore, Gulrajani et al. [2017] introduced a gradient penalty to stabilize the training process of GANs. Our Eq. (7.3) predicts this directly. It is clear that the integral sum becomes more and more sample efficient as we reduce the gradient of our discriminator function. Finally, we are left with the task of choosing an objective function for our generator function. Note that there are multiple ways of choosing this given different goals that we might wish to accomplish. If we wish to match the original GAN objective, we can simply train G_θ such that it produces samples from places where D_ϕ produces high responses, since that will result in sample efficiency. This leads to the generative objective as in Eq. (7.4)

$$\underset{\theta}{\operatorname{argmax}} \log \sum_{j=0}^{m-1} D_\phi(G_\theta(z_j \sim \mathcal{N}(0,I))) \tag{7.4}$$

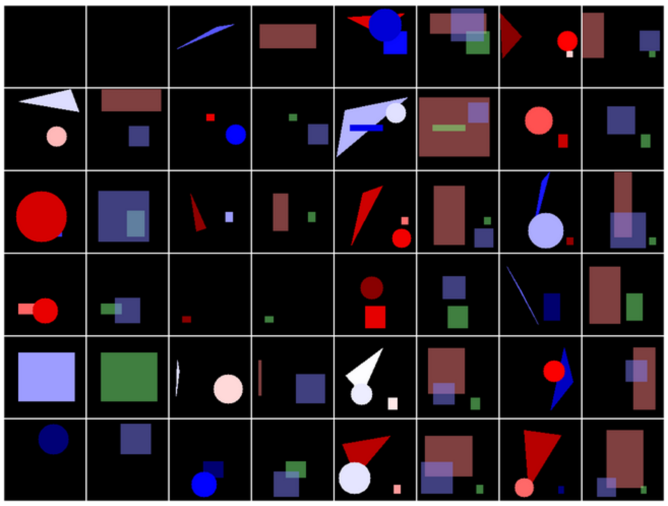
Putting Eq. (7.3) and (7.4) together will recover the practical GAN objective we regularly use including a gradient penalty as regularizer. However, one can notice that there is nothing that restricts G_θ from generating a small diversity in its output. Indeed, it is one of the infamous problems GANs have, mode dropping. It can be attributed to one constraint we simply disregarded while writing Eq. (7.4). Recall, from Eq. (7.2), that x^j-s must be a partition of the data space. However, nowhere in Eq. (7.4), is that enforced. It is unclear how to force this constraint efficiently, but, if done correctly, it should eliminate the mode drop problem. This is left as future work.

7.5.2 Assemble GAN

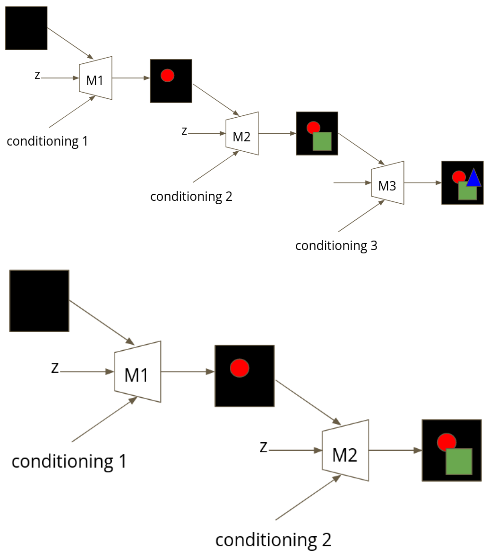
One of the open questions also posed in Sec. 7.4 is–how to obtain n 'object specific' generative models given an unlabelled data set of images, each member of which contains a random subset of all these objects. We envision that this can be achieved by building a GAN generator by 'assembling' many object specific generators. To make this idea concrete, let us imagine we get a data-set of images that consist of a black background, and the foreground consists of three 2D objects, namely circles, triangles, and squares. Now, in any particular image from our data-set, all or none of these objects can appear. Furthermore, let us introduce a dependency among the foreground objects, *i.e.*, say at least one of the vertices of the triangle always appears above (lower y coordinate value in the picture space) all the vertices of the square. Let us call it the simple shapes data set. We visualize a few images from such a data set in Figure 7.1a. Now, the objective is to learn a specific model for each foreground object, *i.e.*, here one model for triangles, one for circles and another for squares. The generative process is as shown in Figure 7.1b. Specifically, we sequentially run one generator after the other while conditioning a generator at the $m-$th stage with the image 'assembled' so far but the generators up to the $(m-1)-$th stage and with conditioning attributes of the object we intend to insert. Finally, the discriminator will be realized as a standard conditional neural network. Beyond this toy case, when run on, say images of eye-glasses and human faces, we expect to obtain one generator for glasses and another for faces, both of which can condition on the other, *i.e.*, given a picture of an eyeglass the face generator can place the face in the context or given a human face the glass generator can put on glasses that fit the face image. This can find applications in virtual try on. Assemble GANs can further be extended to more diverse data sets, *e.g.*, MS-COCO [Lin et al., 2014] to obtain models for many objects. This generative mechanism can then be used to generate data sets that are hard to capture in real life, *e.g.*, generation of rare events in the context of autonomous driving.

7.6 Discussions

To act in the real-world, one must have a model for it. Intuitively, it seems to be in line with how we humans make up our mind, namely by imagining possible outcomes of our actions.



(a)



(b)

Figure 7.1: (5.9a) Simple shapes data set. From left in alternative columns, the foreground objects and their bounding box. The bounding boxes are used as conditioning for the generator. 7.1b Two instances of the generative process of Assemble GAN.

Inherently, it is a many-to-many mapping. Generative models, especially conditional models, provide a mechanism to generate the possible outcomes of one's actions. One aspect of human learning that we seem not to have found a parallel of in machine learning is efficient lifelong learning. Although reinforcement learning gets close, a crucial missing piece there is the requirement of a world model with sufficient accuracy. Additionally, an agent's behavior must be judged. It is usually achieved through a reward function. A generative model seems like a way to represent a learnable world model. Given that, setting up an agent in the same spirit of classical 2-player game would be interesting. More concretely, a choice of an action can be made by generating the possible world states, *i.e.*, 'thinking about' the 'intuitive' actions and choosing the best one driven by some objective. From this perspective, a conditional generative model seems well suited. However, more research is necessary to extend our understanding of conditional generative models before they can be used as world model used for planing.

Appendix A

Regularized autoencoders

A.1.1 A Probabilistic Derivation of Regularization

In this section, we provide a probabilistic treatment to the RAE objective (Eq. (3.11) in Sec. 3.5) Here, we propose an alternative view on enforcing smoothness on the output of D_θ by augmenting the ELBO optimization problem for VAEs with an explicit constraint. While we keep the Gaussianity assumptions over a stochastic D_θ and $P(Z)$ for convenience, we, however, are not fixing a parametric form for $Q_\phi(Z|X)$ yet. We discuss next how some parametric restrictions over $Q_\phi(Z|X)$ lead to a variation of the RAE framework in Eq. (3.11), specifically the introduction of $\mathcal{L}_{\mathsf{GP}}$ as a regularizer of a deterministic version of the CV-VAE. To start, we augment Eq. (3.5) as:

$$\begin{aligned}&\underset{\phi,\theta}{\operatorname{argmin}}\ \mathbb{E}_{x\sim P_{\mathsf{data}}(X)}\ \mathcal{L}_{\mathsf{REC}}+\mathcal{L}_{\mathsf{KL}} \\ &\text{s.t. } ||D_\theta(z_1)-D_\theta(z_2)||_p<\varepsilon \quad \forall\, z_1,z_2\sim Q_\phi(Z|X) \quad \forall x\sim P_{\mathsf{data}}\end{aligned} \tag{A.1}$$

where $D_\theta(z)=\mu_\theta(E_\phi(X))$ and the constraint on the decoder encodes that the output has to vary, in the sense of an L_p norm, only by a small amount ε for any two possible draws from the encoding of x. Let $D_\theta(z):\mathbb{R}^{\dim(z)}\to\mathbb{R}^{\dim(x)}$ be given by a set of $\dim(x)$ elements. Specifically, it is given by $\{d_i(z):\mathbb{R}^{\dim(z)}\to\mathbb{R}^1\}$. Now we can upper bound the quantity $||D_\theta(z_1)-D_\theta(z_2)||_p$ by $\dim(x)*sup_i\{||d_i(z_1)-d_i(z_2)||_p\}$. Using mean value theorem $||d_i(z_1)-d_i(z_2)||_p\leq||\nabla_t d_i((1-t)z_1+t*z_2)||_p*||z_1-z_2||_p$. Hence $sup_i\{||d_i(z_1)-d_i(z_2)||_p\}\leq sup_i\{||\nabla_t d_i((1-t)z_1+tz_2)||_p*||z_1-z_2||_p\}$. Now, if we choose the domain of $Q_\phi(Z|X)$ to be isotopic, the contribution of $||z_2-z_1||_p$ to the aforementioned quantity becomes a constant factor. Loosely speaking, it is the diameter of the bounding ball of domain of $Q_\phi(Z|X)$. Hence, the above term simplifies to $sup_i\{||\nabla_t d_i((1-t)z_1+t*z_2)||_p\}$. Recognizing that here z_1 and z_2 are arbitrary, lets us simplify this further to $sup_i\{||\nabla_z d_i(z)||_p\}$

From this form of the smoothness constraint, it is apparent why the choice of a parametric form for $Q_\phi(Z|X)$ can be impactful during training. For a compactly supported isotropic PDF $Q_\phi(Z|X)$, the extension of the support $sup\{||z_1-z_2||_p\}$ would depend on its entropy $\mathbb{H}(Q_\phi(Z|X))$. Through some functional r. For instance, a uniform posterior over a hypersphere in Z would ascertain $r(\mathbb{H}(Q_\phi(Z|X)))\cong e^{\mathbb{H}(Q_\phi(Z|X))/n}$ where n is the dimensionality of the latent space.

Intuitively, one would look for parametric distributions that do not favor overfitting, *e.g.*, degenerating in Dirac-deltas (minimal entropy and support) along any dimensions. To this end, an isotropic nature of $Q_\phi(Z|X)$ would favor such a robustness against decoder over-fitting. We can now rewrite the constraint as

$$r(\mathbb{H}(Q_\phi(Z|X))) \cdot sup\{||\nabla D_\theta(Z)|||_p\} < \varepsilon \tag{A.2}$$

The $\mathcal{L}_{\mathsf{KL}}$ term can be expressed in terms of $\mathbb{H}(Q_\phi(Z|X))$, by decomposing it as $\mathcal{L}_{\mathsf{KL}} = \mathcal{L}_{\mathsf{CE}} - \mathcal{L}_{\mathsf{H}}$, where $\mathcal{L}_{\mathsf{H}} = \mathbb{H}(Q_\phi(Z|X))$ and $\mathcal{L}_{\mathsf{CE}} = \mathbb{H}(Q_\phi(Z|X), P(Z))$ represents a cross-entropy term. Therefore, the constrained problem in Eq. (A.1) can be written in a Lagrangian formulation by including Eq. (A.2):

$$\underset{\phi,\theta}{\text{argmin}}\ \mathbb{E}_{x\sim P_{\mathsf{data}}}\ \mathcal{L}_{\mathsf{REC}} + \mathcal{L}_{\mathsf{CE}} - \mathcal{L}_{\mathsf{H}} + \lambda\mathcal{L}_{\mathsf{LANG}} \tag{A.3}$$

where $\mathcal{L}_{\mathsf{LANG}} = r(\mathbb{H}(Q_\phi(Z|X))) * ||\nabla D_\theta(Z)||_p$. We argue that a reasonable simplifying assumption for $Q_\phi(Z|X)$ is to fix $\mathbb{H}(Q_\phi(Z|X))$ to a single constant for all realization of the random variable X. Intuitively, this can be understood as fixing the variance of, $Q_\phi(Z|X)$ as we did for the CV-VAE in Sec. 3.4. With this simplification, Eq. (A.3) further reduces to

$$\underset{\phi,\theta}{\text{argmin}}\ \mathbb{E}_{x\sim P_{\mathsf{data}}(X)}\ \mathcal{L}_{\mathsf{REC}} + \mathcal{L}_{\mathsf{CE}} + \lambda||\nabla D_\theta(Z)||_p \tag{A.4}$$

We can readily recognize $||\nabla D_\theta(z)||_p$ to be the gradient penalty $\mathcal{L}_{\mathsf{GP}}$ term and $\mathcal{L}_{\mathsf{CE}} = ||z||_2^2$ corresponds to $\mathcal{L}^{\mathsf{RAE}}_{\mathsf{KL}}$. This concludes our probabilistic take on the RAE objective function as presented in Eq. (3.11).

A.1.2 Network architecture, Training Details and Evaluation

We follow the models adopted by Tolstikhin et al. [2017] with the difference that we consistently apply batch normalization [Ioffe and Szegedy, 2015]. The latent space dimension is 16 for MNIST [LeCun et al., 1998b], 128 for CIFAR-10 [Krizhevsky and Hinton, 2009] and 64 for CelebA [Liu et al., 2015].

For all experiments, we use the Adam optimizer with a starting learning rate of 10^{-3}, which is cut in half every time the validation loss plateaus. All models are trained for a maximum of 100 epochs on MNIST and CIFAR and 70 epochs on CelebA. We use a mini-batch size of 100 and pad MNIST digits with zeros to make the size 32×32.

We use the official train, validation and test splits of CelebA. For MNIST and CIFAR, we set aside 10k train samples for validation. For random sample evaluation, we draw samples from $\mathcal{N}(0,I)$ for VAE and WAE-MMD and for all remaining models, samples are drawn from a multivariate Gaussian whose parameters (mean and covariance) are estimated using training set embeddings. For the GMM density estimation, we also utilize the training set embeddings for fitting and validation set embeddings to verify that GMM models are not over fitting to training embeddings. However, due to the very low number of mixture components (10), we did not encounter overfitting at this step. The GMM parameters are estimated by running EM for at most 100 iterations.

Conv$_n$ represents a convolutional layer with n filters. All convolutions Conv$_n$ and transposed convolutions ConvT$_n$ have a filter size of 4×4 for MNIST and CIFAR-10 and 5×5 for CELEBA. They all have a stride of size 2 except for the last convolutional layer in the decoder. Finally, $M = 1$ for all models except for the VAE which has $M = 2$ as the encoder has to produce

	MNIST	CIFAR_10	CELEBA
Encoder:	$x \in \mathcal{R}^{32\times 32}$	$x \in \mathcal{R}^{32\times 32}$	$x \in \mathcal{R}^{64\times 64}$
	$\rightarrow$ Conv$_{128}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{128}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{128}$ $\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ Conv$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ Conv$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ Conv$_{1024}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{1024}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ Conv$_{1024}$ $\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ Flatten $\rightarrow$ FC$_{16\times M}$	$\rightarrow$ Flatten $\rightarrow$ FC$_{128\times M}$	$\rightarrow$ Flatten $\rightarrow$ FC$_{64\times M}$
Decoder:	$z \in \mathcal{R}^{16}$ $\rightarrow$ FC$_{8\times 8\times 1024}$	$z \in \mathcal{R}^{128}$ $\rightarrow$ FC$_{8\times 8\times 1024}$	$z \in \mathcal{R}^{64}$ $\rightarrow$ FC$_{8\times 8\times 1024}$
	$\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ ConvT$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ ConvT$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ ConvT$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ ConvT$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ ConvT$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU	$\rightarrow$ ConvT$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU
	$\rightarrow$ ConvT$_{1}$	$\rightarrow$ ConvT$_{1}$	$\rightarrow$ ConvT$_{128}$ $\rightarrow$ BN $\rightarrow$ ReLU
			$\rightarrow$ ConvT$_{1}$

Table A.1: Detailed network architecture used for our experiments for different datasets.

both mean and variance for each input.

A.1.3 Evaluation Setup

We compute the FID of the reconstructions of random validation samples against the test set to evaluate reconstruction quality. For evaluating generative modeling capabilities, we compute the FID between the test data and randomly drawn samples from a single Gaussian that is either the isotropic $P(Z)$ fixed for VAEs and WAEs, a learned second stage VAE for 2sVAEs, or a single Gaussian fit to $q_\delta(Z)$ for CV-VAEs and RAEs. For all models, we also evaluate random samples from a 10-component Gaussian Mixture model (GMM) fit to $q_\delta(Z)$. Using only 10 components prevents us from overfitting (which would indeed give good FIDs when compared with the test set)[1].

For interpolations, we report the FID of the furthest interpolation points obtained by applying spherical interpolation to randomly selected validation reconstruction pairs.

We use 10k samples for all FID and PRD evaluations. Scores for random samples are evaluated against the test set. Reconstruction scores are computed from validation set reconstructions against the respective test set. Interpolation scores are computed by interpolating latent codes of a pair of randomly chosen validation embeddings vs test set samples. The visualized interpolation samples are interpolations between two randomly chosen test set images.

[1]We note that fitting GMMs with up to 100 components only improved results marginally. Additionally, we provide nearest-neighbours from the training set in Appendix A.1.6 to show that our models are not overfitting.

A.1.4 Evaluation by Precision and Recall

	MNIST		CIFAR-10		CelebA	
	$\mathcal{N}$	GMM	$\mathcal{N}$	GMM	$\mathcal{N}$	GMM
VAE	0.96 / 0.92	0.95 / 0.96	0.25 / 0.55	0.37 / 0.56	0.54 / 0.66	0.50 / 0.66
CV-VAE	0.84 / 0.73	0.96 / 0.89	0.31 / 0.64	0.42 / 0.68	0.25 / 0.43	0.32 / 0.55
WAE	0.93 / 0.88	**0.98** / 0.95	0.38 / 0.68	0.51 / **0.81**	0.59 / 0.68	**0.69** / **0.77**
RAE-GP	0.93 / 0.87	0.97 / **0.98**	0.36 / 0.70	0.46 / 0.77	0.38 / 0.55	0.44 / 0.67
RAE-L2	0.92 / 0.87	**0.98** / **0.98**	0.41 / 0.77	**0.57** / **0.81**	0.36 / 0.64	0.44 / 0.65
RAE-SN	0.89 / 0.95	**0.98** / 0.97	0.36 / 0.73	0.52 / **0.81**	0.54 / 0.68	0.55 / 0.74
RAE	0.92 / 0.85	**0.98** / **0.98**	0.45 / 0.73	0.53 / 0.80	0.46 / 0.59	0.52 / 0.69
AE	0.90 / 0.90	**0.98** / 0.97	0.37 / 0.73	0.50 / 0.80	0.45 / 0.66	0.47 / 0.71

Table A.2: Evaluation of random sample quality by precision / recall [Sajjadi et al., 2018] (higher numbers are better, best value for each dataset in bold). It is notable that the proposed ex-post density estimation improves not only precision, but also recall throughout the experiment. For example, WAE seems to have a comparably low recall of only 0.88 on MNIST, which is raised considerably to 0.95 by fitting a GMM. In all cases, GMM gives the best results. Another interesting point is the low precision but high recall of all models on CIFAR-10 – this is also visible upon inspection of the samples in Fig. A.6.

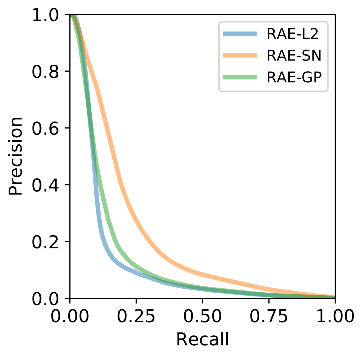
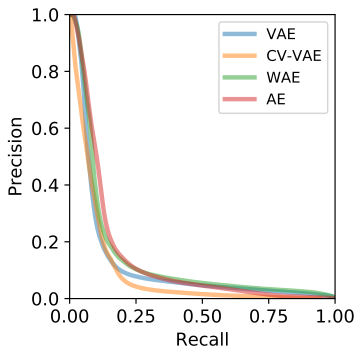
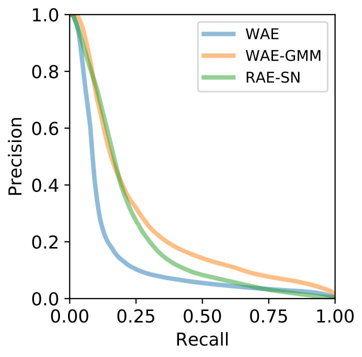
PRD all RAEs	PRD all traditional VAEs	WAE, RAE-SN, WAE-GMM
		

Figure A.1: PRD curves of all RAE methods (left), reflects a similar story as FID scores do. RAE-SN seems to perform the best in both precision and recall metric. PRD curves of all traditional VAE variants (middle). Similar to the conclusion predicted by FID scores, there is no clear winner. PRD curves for the WAE (with isotropic Gaussian prior), WAE-GMM model with ex-post density estimation by a 10-component GMM and RAE+SN-GMM (right). This finer grained view shows how the WAE-GMM scores higher recall but lower precision than a RAE+SN-GMM while scoring comparable FID scores. Note that ex-post density estimation greatly boosts the WAE model in both PRD and FID scores.

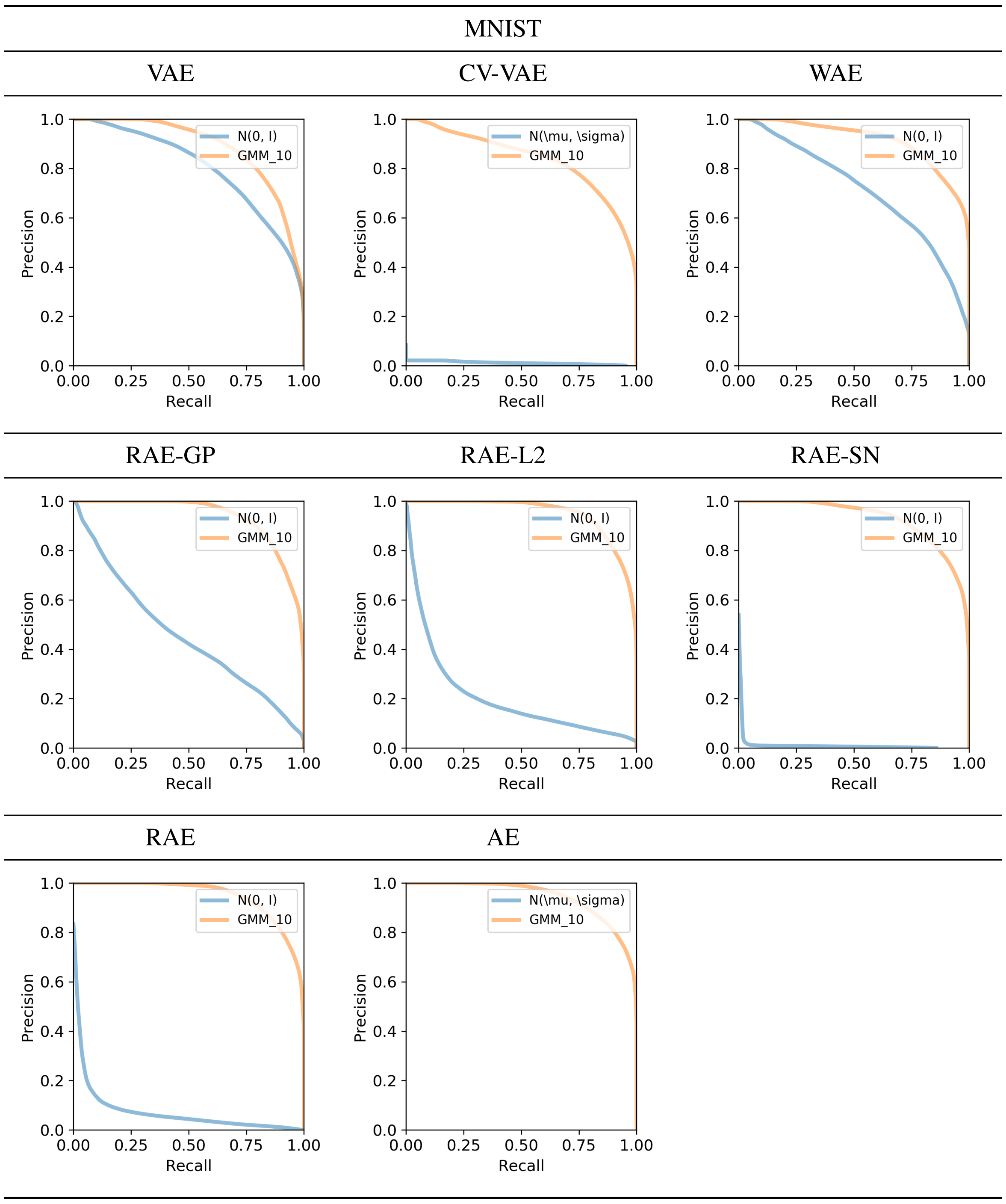
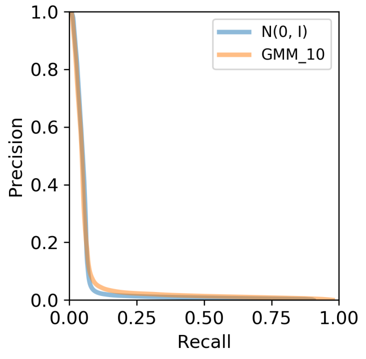


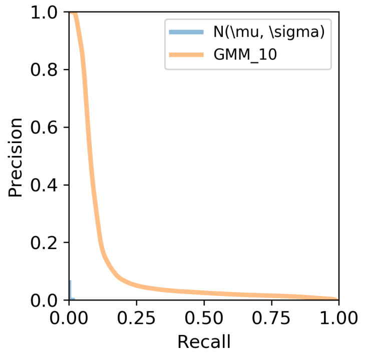
Figure A.2: PRD curves of all methods on image data experiments on MNIST. For each plot, we show the PRD curve when applying the fixed or the fitted one by ex-post density estimation (XPDE). XPDE greatly boosts both precision and recall for all models.

Cifar 10

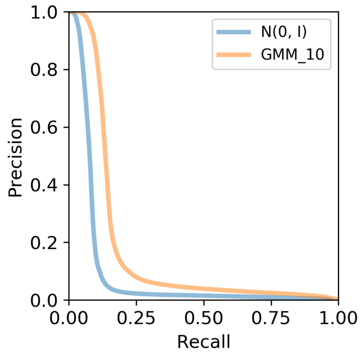
VAE



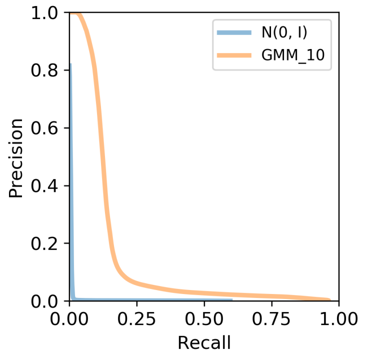
CV-VAE



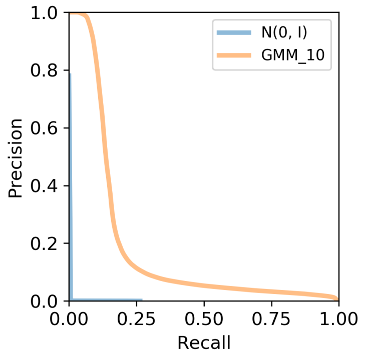
WAE



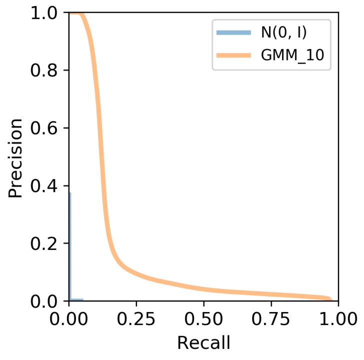
RAE-GP



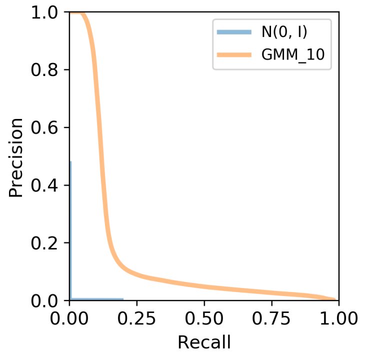
RAE-L2



RAE-SN



RAE



AE

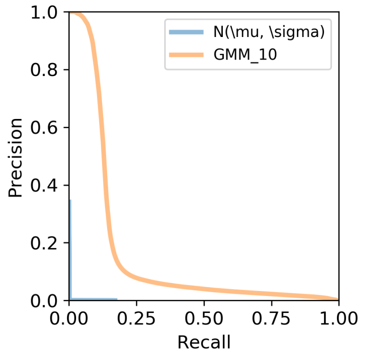


Figure A.3: PRD curves of all methods on image data experiments on CIFAR10. For each plot, we show the PRD curve when applying the fixed or the fitted one by ex-post density estimation (XPDE). XPDE greatly boosts both precision and recall for all models.

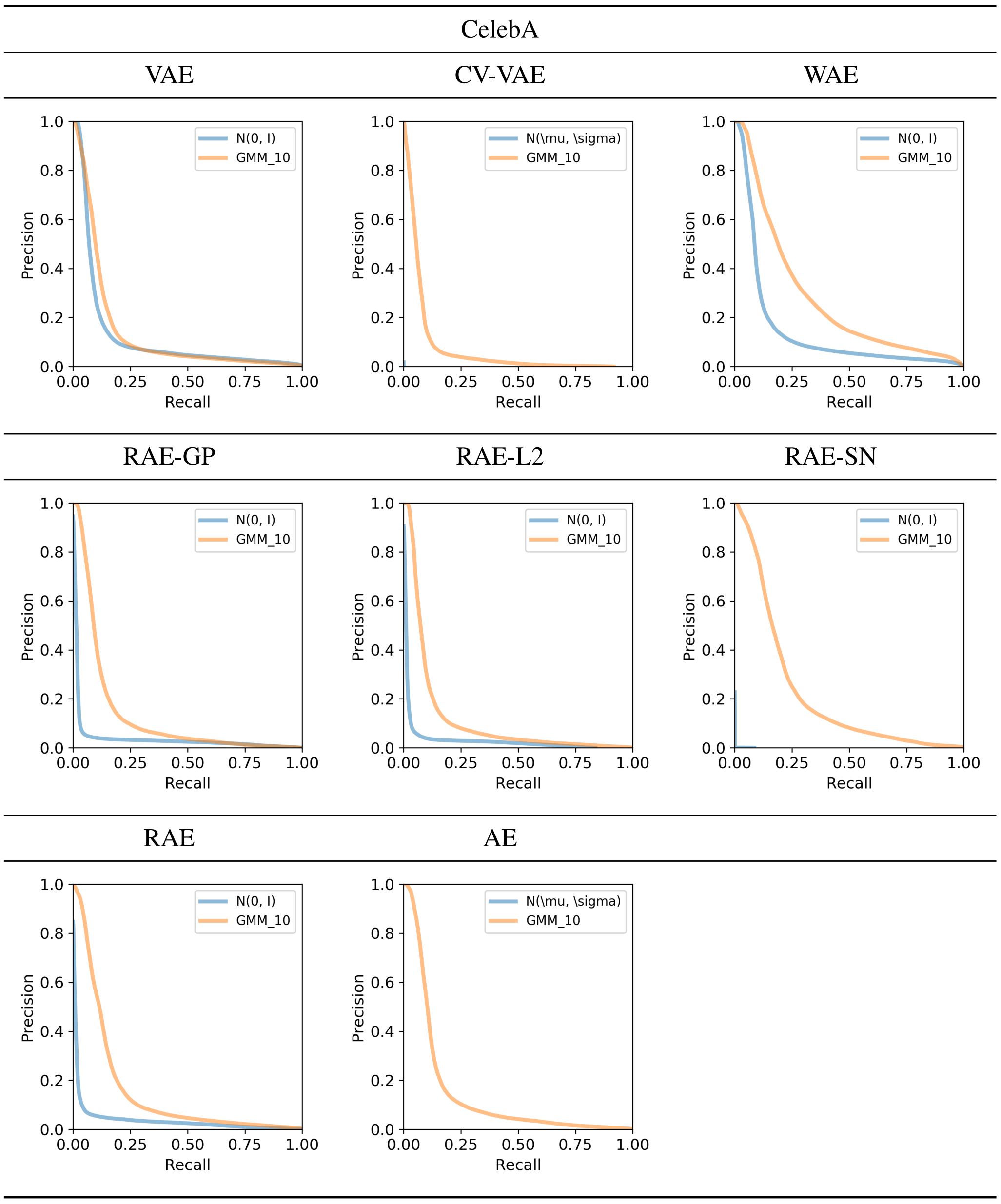


Figure A.4: PRD curves of all methods on image data experiments on CELEBA. For each plot, we show the PRD curve when applying the fixed or the fitted one by ex-post density estimation (XPDE). XPDE greatly boosts both precision and recall for all models.

A.1.5 More Qualitative Results

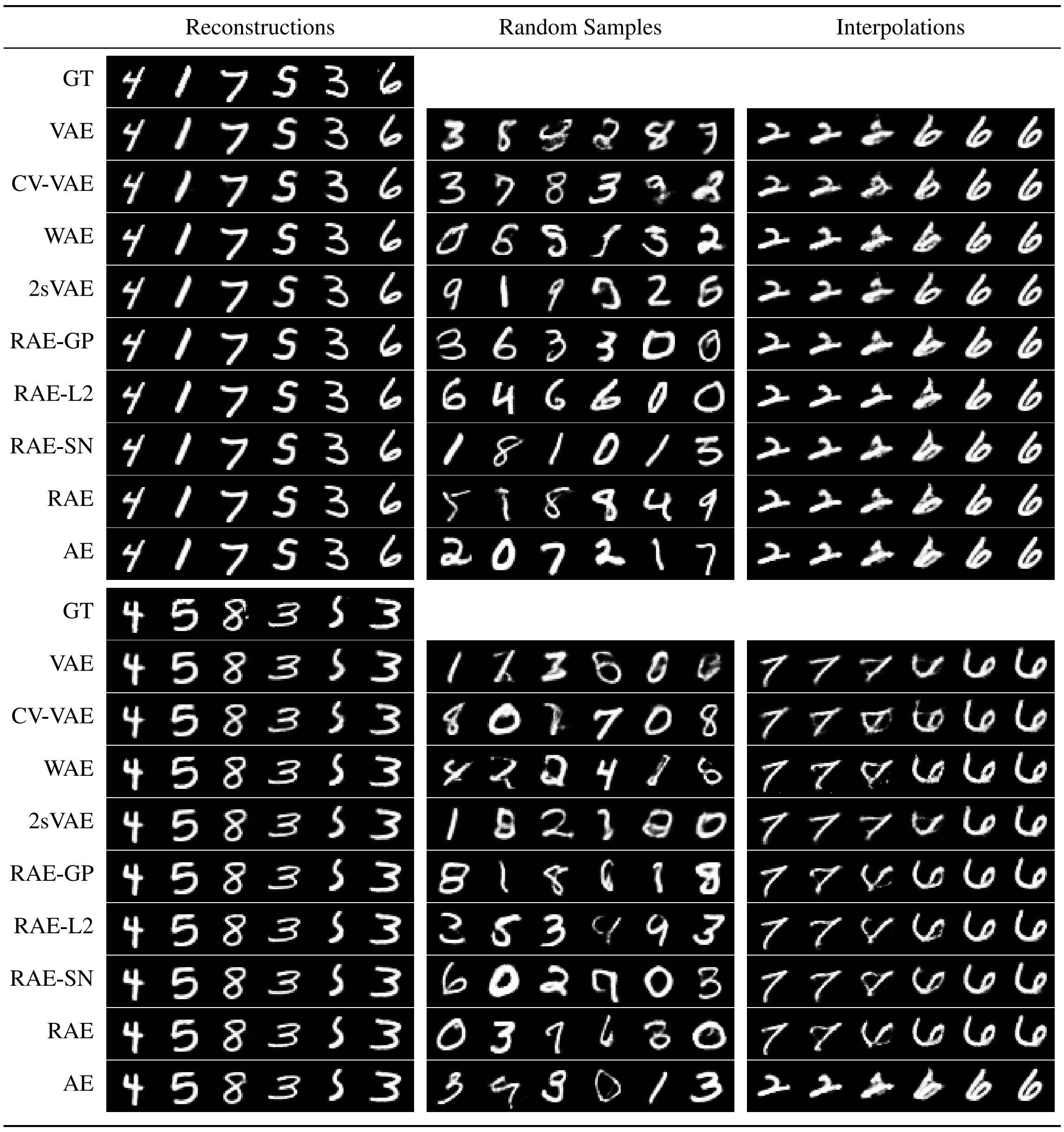


Figure A.5: Qualitative evaluation for sample quality for VAEs, WAEs and RAEs on MNIST. Left: reconstructed samples (top row is ground truth). Middle: randomly generated samples. Right: spherical interpolations between two images (first and last column).

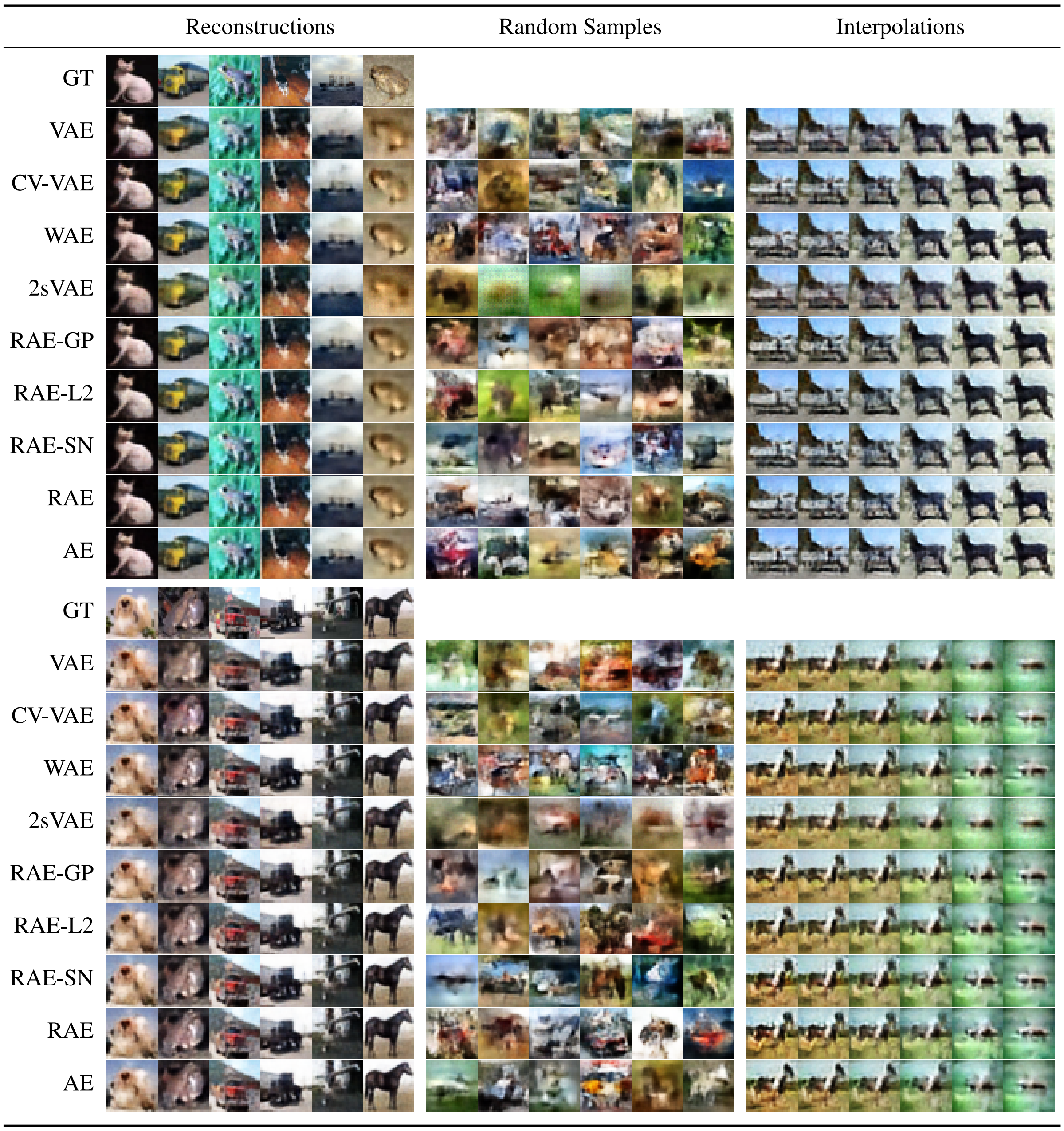


Figure A.6: Qualitative evaluation for sample quality for VAEs, WAEs and RAEs on CIFAR-10. Left: reconstructed samples (top row is ground truth). Middle: randomly generated samples. Right: spherical interpolations between two images (first and last column).

A.1.6 Investigating Overfitting

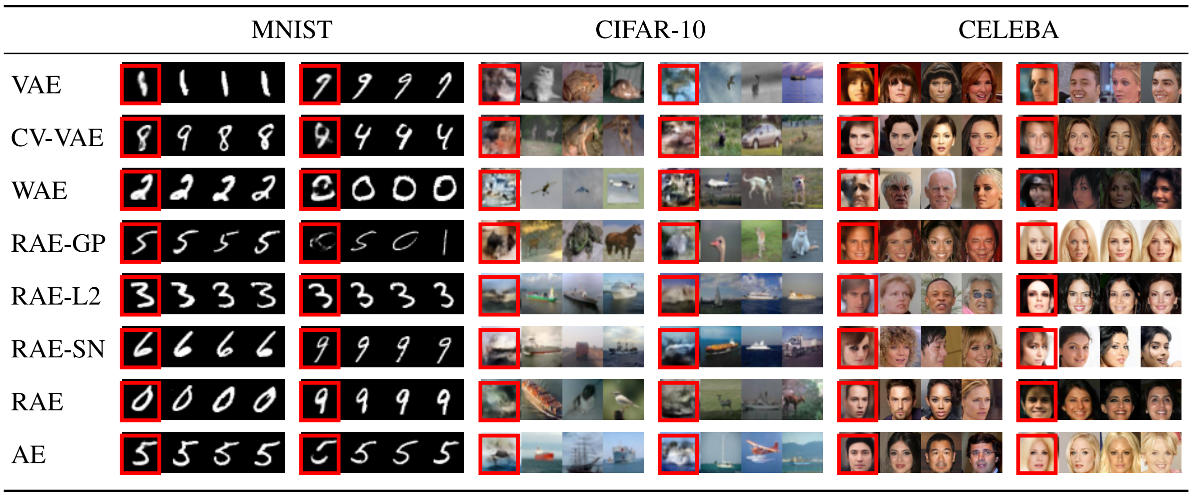


Figure A.7: Nearest neighbors to generated samples (leftmost image, red box) from training set. It seems that the models have generalized well and fitting only 10 Gaussians to the latent space prevents overfitting.

A.1.7 Visualizing Ex-Post Density Estimation

To visualize that ex-post density estimation does indeed help reduce the mismatch between the aggregated posterior and the prior we train a VAE on the MNIST dataset whose latent space is 2 dimensional. The unique advantage of this setting is that one can simply visualize the density of test sample in the latent space by plotting them as a scatterplot. As it can be seen from figure A.8, an expressive density estimator effectively fixes the miss-match and this as reported earlier results in better sample quality.

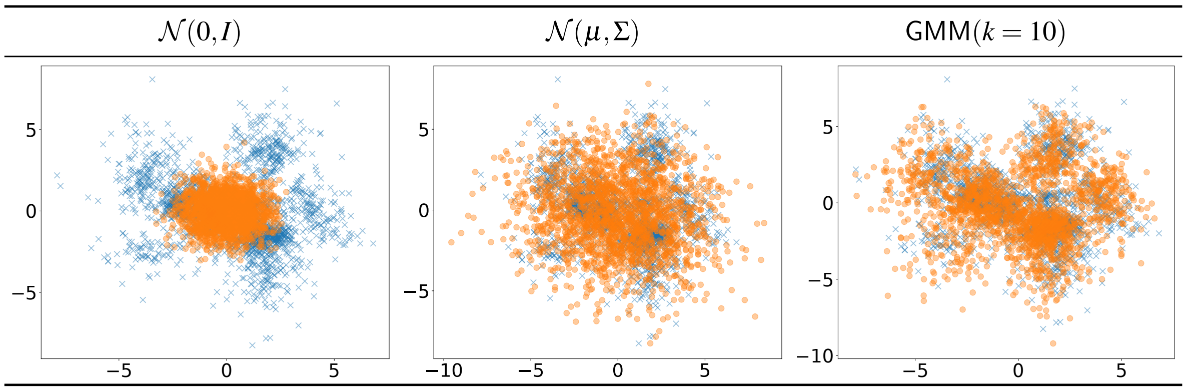


Figure A.8: Different density estimations of the 2-dimensional latent space of a VAE learned on MNIST. The blue points are 2000 test set samples while the orange ones are drawn from the estimator indicated in each column: isotropic Gaussian (left), multivariate Gaussian with mean and covariance estimated on the training set (center) and a 10-component GMM (right). This clearly shows the aggregated posterior mismatch w.r.t. to the isotropic Gaussian prior imposed by VAEs and how ex-post density estimation can help fix the estimate.

Here in figure A.9 we perform the same visualization on with all the models trained on the MNIST dataset as employed on our large evaluation in Table 1. Clearly, every model depicts rather large mismatch between aggregate posterior and prior. Once again, the advantage of ex-post density estimate is clearly visible.

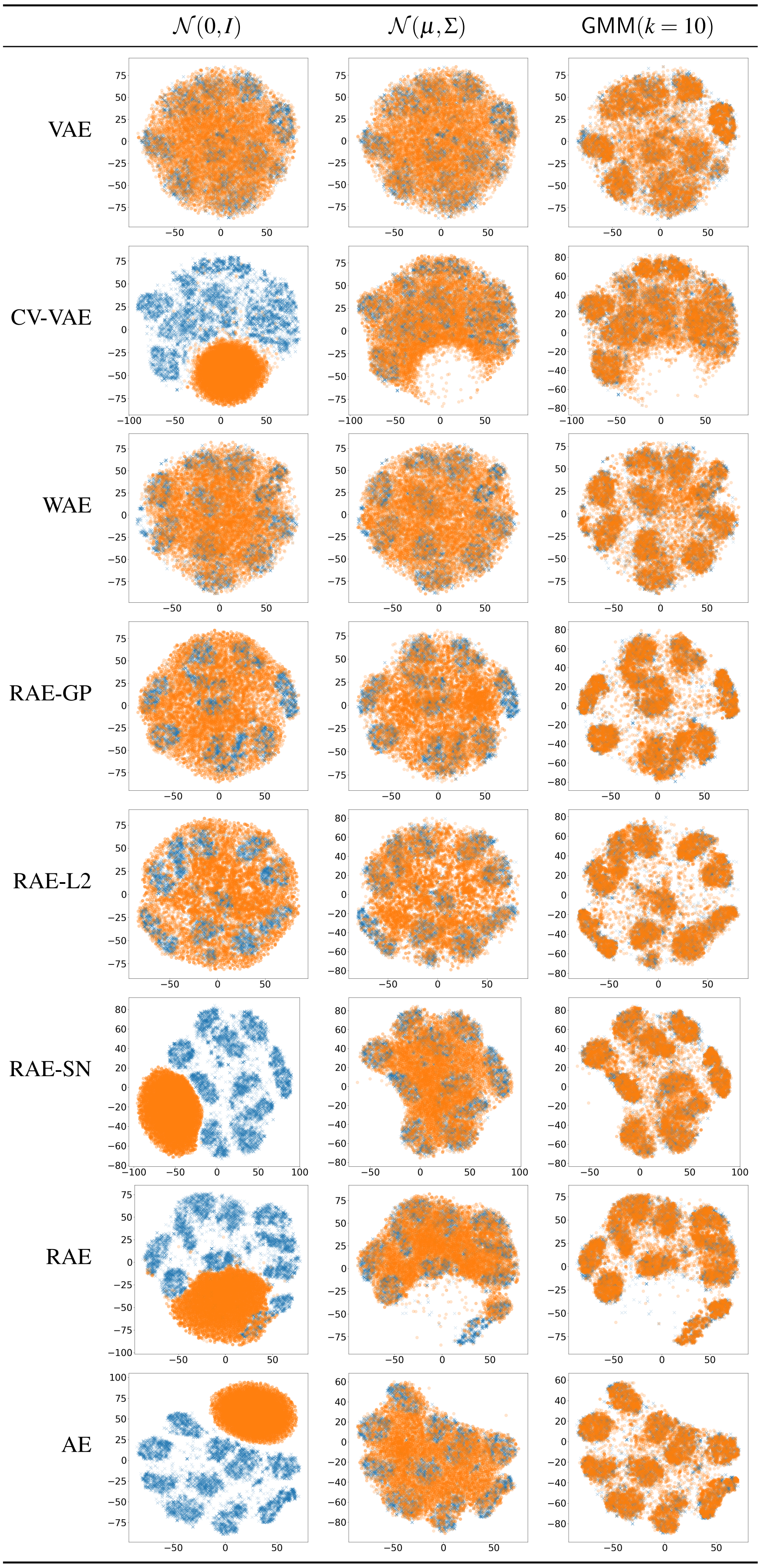


Figure A.9: Different density estimations of the 16-dimensional latent spaces learned by all models on MNIST (see Table 1) here projected in 2D via T-SNE. The blue points are 2000 test set samples while the orange ones are drawn from the estimator indicated in each column:

	MNIST				CIFAR				CelebA			
	Rec.	Samples			Rec.	Samples			Rec.	Samples		
		$\mathcal{N}$	GMM	Interp.		$\mathcal{N}$	GMM	Interp.		$\mathcal{N}$	GMM	Interp.
RAE-GP	14.04	**22.21**	11.54	15.32	32.17	83.05	76.33	64.08	**39.71**	116.30	45.63	47.00
RAE-L2	10.53	22.22	**8.69**	**14.54**	32.24	**80.80**	74.16	62.54	43.52	51.13	47.97	45.98
RAE-SN	15.65	19.67	11.74	15.15	27.61	84.25	75.30	63.62	36.01	**44.74**	**40.95**	**39.53**
RAE	11.67	23.92	9.81	14.67	29.05	83.87	76.28	63.27	40.18	48.20	44.68	43.67
AE	12.95	58.73	10.66	17.12	30.52	84.74	76.47	**61.57**	40.79	127.85	45.10	50.94
AE-L2	11.19	315.15	9.36	17.15	34.35	247.48	75.40	61.09	44.72	346.29	48.42	56.16
RAE-GP-L2	**9.70**	72.64	9.07	16.07	33.25	187.07	79.03	62.48	47.06	72.09	51.55	50.28
RAE-L2-SN	10.67	50.63	9.42	15.73	**24.17**	240.27	**74.10**	61.71	39.90	180.39	44.39	42.97
RAE-SN-GP	17.00	139.61	13.12	16.62	33.04	284.36	75.23	62.86	63.75	299.69	71.05	68.87
RAE-L2-SN-GP	16.75	144.51	13.93	16.75	29.96	290.34	74.22	61.93	68.86	318.67	75.04	74.29

Table A.3: Comparing multiple regularization schemes for RAE models. The improvement in reconstruction, random sample quality and interpolated test samples is generally comparable, but hardly much better. This can be explained with the fact that the additional regularization losses make tuning their hyperparameters more difficult, in practice.

A.1.8 Combining multiple regularization terms

The rather intriguing facts that AE without explicit decoder regularization performs reasonably well, as seen from table 3.1, indicates that convolutional neural networks when combined with gradient-based optimizers inherit some implicit regularization. This motivates us to investigate a few different combinations of regularizations *e.g.*, we regularize the decoder of an auto-encoder while drop the regularization in the z space. The results of this experiment are reported in the row marked AE-L2 in table A.3.

Further, a recent GAN literature [Lucic et al., 2018] report that often a combination of regularizations boosts performance of neural networks. Following this, we combine multiple regularization techniques in our framework. However, note that this rather drastically increases the number of hyperparameters and the models become harder to train and goes against the core theme of this work, which strives for simplicity. Hence, we perform simplistic effort to tune all the hyperparameters to see if this can provide boost in the performance, which seem not to be the case. These experiments are summarized in the second half of the table A.3

Appendix B

Generative interpretable Faces

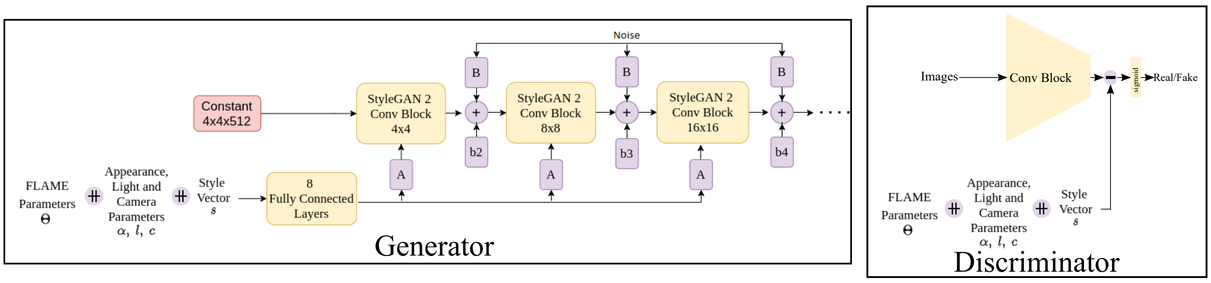


Figure B.1: Architecture of Vector conditioning model. Here $+\!\!+$ represents a concatenation.

B.1.1 Vector conditioning architecture

Here we describe the model architecture of the vector condition model, used as one of the baseline models in Sec. 5.3.5. For this model, we pass the vector values conditioning parameters FLAME (β, θ, ψ), appearance (α) and lighting (l) as a 236 dimensional vector through the dimensions of style vector of the original StyleGAN2 architecture as shown in Figure B.1. We further input the same conditioning vector to the discriminator at the last fully connected layer of the discriminator by subtracting it from the last layer activation.

B.1.2 Image centering for extreme rotations

As discussed in Sec. 5.3.6 of the main paper, GIF produces artifacts for extreme head poses close to profile view. This is due to the pixel alignment of the FLAME renderings and the generated images, which requires the images to be similarly eye-centered as the FFHQ training data. For profile views, however, it is unclear how the centering within the training data was achieved. The centering strategy used in GIF causes a zoom in for profile views, effectively cropping parts of the face, and hence the generator struggles to generate realistic images as shown in Figure B.2.

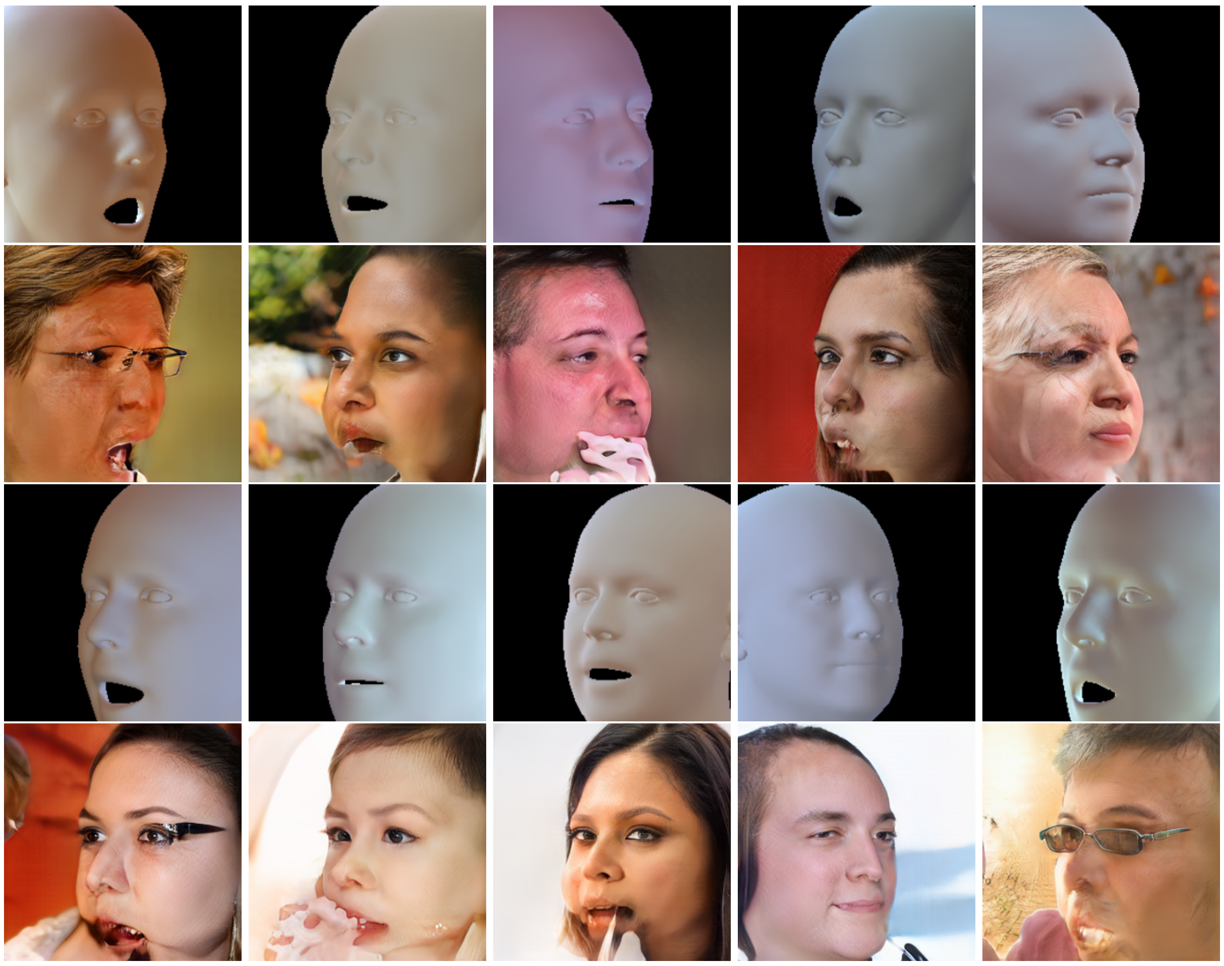


Figure B.2: In order for the eyes to be places at a given pixel location a profile view causes the face image to be highly zoomed in. This causes the generated images to become unrealistic.

Appendix C

Invertible GANs

Video Key Framing

Temporal interpolation of video frames: Here we present one of the application of InvGAN. Specifically, we demonstrate that one can boost the frame rate of a video post capture. We infer the tiled latent space of consecutive frames in a video, and linearly interpolate each tile to generate one or more intermediate frames. Results are shown in Figure C.1. This can be used to create slow motion video, post capture. We find the latent code of each frame in a video sequence, and then derive intermediate latent codes by weighted averaging of neighboring latent codes using a Gaussian window. Even though this in effect interpolates between latent codes for background patch and foreground patch for fast-moving small objects leading to blur, it results in smooth slow motion video. We use the UCF101 data set [Soomro et al., 2012] for this task.

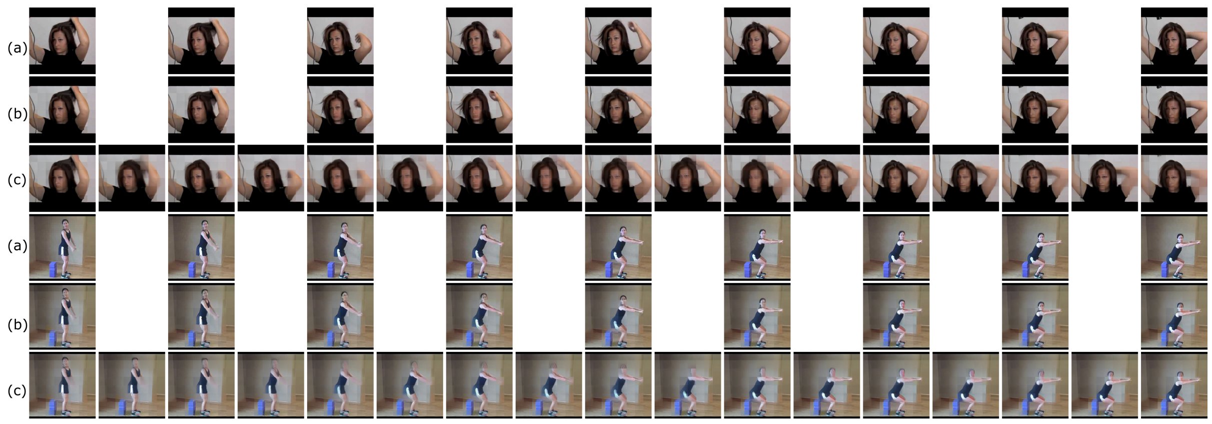


Figure C.1: Tiled reconstruction of a video sequence. (a) original sequence, (b) reconstructed sequence, (c) up-sampled in time sequence.

CelebA Reconstructions

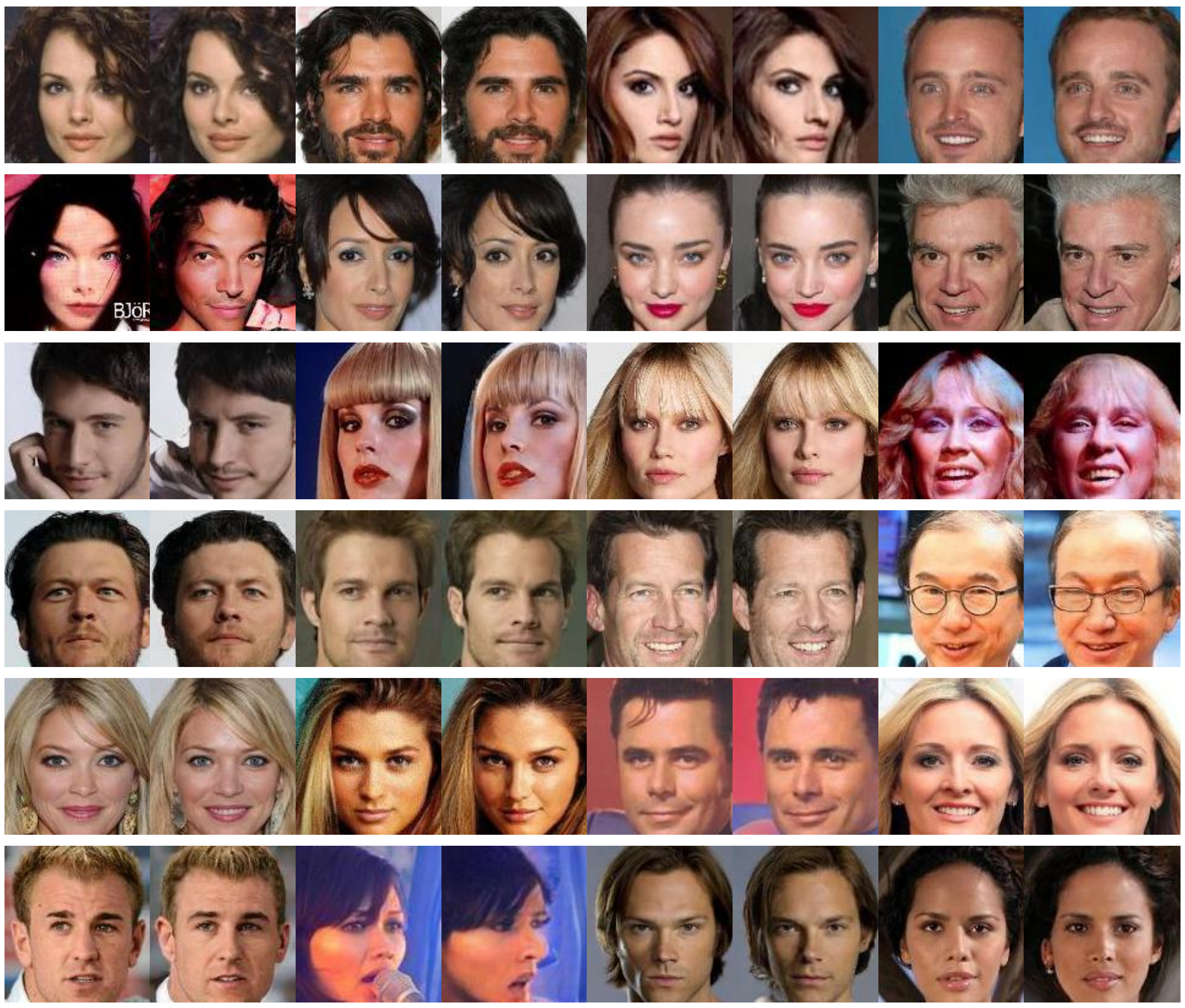


Figure C.2: InvGAN reconstructions of CelebA at 128 $\times$ 128 resolution. Alternating (from left to right) original and reconstructed images.

Additional FID Evaluations

For completeness of Table 6.1 in Sec. 6.4. We additionally evaluate the FID scores on reconstructions using the approach of ReStyle [Alaluf et al., 2021]. The resulting scores are presented in Table C.1. Unfortunately, we can not compute test set MAE and FIDs, since ReStyle has been trained on the whole FFHQ set.

models	**RandFID**	**RandRecFID**
1itr FFHQ ReStyle [Alaluf et al., 2021]	40.33/4.71	57.63/29.56
2itr FFHQ ReStyle [Alaluf et al., 2021]	40.33/4.71	53.09/22.88
3itr FFHQ ReStyle [Alaluf et al., 2021]	40.33/4.71	51.68/20.80
4itr FFHQ ReStyle [Alaluf et al., 2021]	40.33/4.71	51.49/19.93

Table C.1: Random sample FID (RandFID), FID of reconstructed random samples (RandRecFID). FID scores are here evaluated using 500 and 50000 samples. They are separated by '/'.

Additional Baseline for Semantic Reconstruction

We conduct the same experiment as presented in Sec. 6.4.1 with reconstructions using ReStyle [Alaluf et al., 2021]. The resulting mean average precision evaluated on the reconstructed evaluation set is 0.80 ± 0.15. Evaluated on original evaluation set images, the performance drops to 0.78 ± 0.17, which indicates a weaker transfer as compared to both the tiled reconstruction and the full reconstruction using InvGAN.

CelebA Image Editing

We conducted the same image editing operations shown in Figure 6.3 on CelebA. The results are shown in Figure C.3.

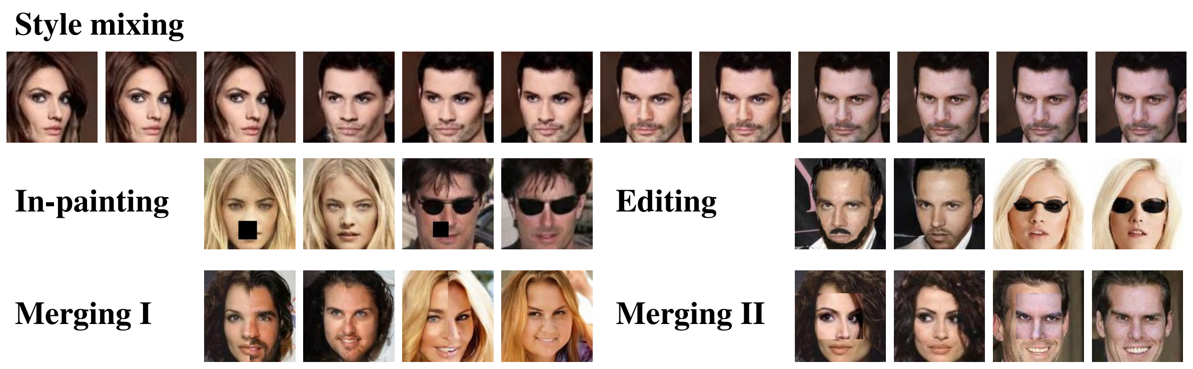


Figure C.3: Benchmark image editing tasks on CelebA (128 px). Style mixing: We transfer the first $0, 1, 2, \ldots, 11$ style vectors from one image to another. For the other image editing tasks, pairs of images are input image (left) and reconstruction (right).

Bibliography

van den Oord Aäron, Kalchbrenner Nal, Kavukcuoglu Koray, and Weinberger Kilian Q. Pixel Recurrent Neural Networks. In *International proceedings on Machine Learning (ICML)*, 2016.

Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. ReStyle: A Residual-Based StyleGAN Encoder via Iterative Refinement. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2021.

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy. Fixing a Broken ELBO. In *International proceedings on Machine Learning (ICML)*, 2018.

Brian Amberg, Reinhard Knothe, and Thomas Vetter. Expression Invariant 3D Face Recognition with a Morphable Model. In *International proceedings on Automatic Face & Gesture Recognition (FG)*, 2008.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. In *International proceedings on Machine Learning (ICML)*, 2017.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated Gradients give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *International proceedings on Machine Learning (ICML)*, 2018a.

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *International proceedings on Machine Learning (ICML)*, 2018b.

Guha Balakrishnan, Yuanjun Xiong, Wei Xia, and Pietro Perona. Towards Causal Benchmarking of Bias in Face Analysis Algorithms. In *European proceedings on Computer Vision (ECCV)*, 2020.

Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end Optimized Image Compression. In *International proceedings on Learning Representations (ICLR)*, 2017.

Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-GAN: Unsupervised Video Retargeting. In *European proceedings on Computer Vision (ECCV)*, 2018.

David Bau, Hendrik Strobelt, William S. Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. Semantic Photo Manipulation with a Generative Image Prior. In *ACM Transactions on Graphics (TOG)*, 2019.

M. Bauer and A. Mnih. Resampled Priors for Variational Autoencoders. In *Artificial Intelligence and Statistics (AISTATS)*, 2019.

Peter N. Belhumeur, João P Hespanha, and David J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using Class Specific Linear Projection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 1997.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2013a.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized Denoising Autoencoders as Generative Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013b.

Pascal Bérard, Derek Bradley, Maurizio Nitti, Thabo Beeler, and Markus Gross. High-quality Capture of Eyes. In *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 2014.

Amit Bermano, Thabo Beeler, Yeara Kozlov, Derek Bradley, Bernd Bickel, and Markus Gross. Detailed spatio-temporal reconstruction of eyelids. In *ACM Transactions on Graphics, (Proc. SIGGRAPH)*, 2015.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006. ISBN 0387310738.

Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *ACM Transactions on Graphics, (Proc. SIGGRAPH)*, 1999.

Volker Blanz, Curzio Basso, Tomaso Poggio, and Thomas Vetter. Reanimating Faces in Images and Video. In *Computer graphics forum*, 2003.

Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the Latent Space of Generative Networks. In *International proceedings on Machine Learning (ICML)*, 2018.

Timo Bolkart and Stefanie Wuhrer. A Groupwise Multilinear Correspondence Optimization for 3D Faces. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2015.

James Booth, Anastasios Roussos, Allan Ponniah, David Dunaway, and Stefanos Zafeiriou. Large scale 3D Morphable Models. In *International journal of Computer Vision (IJCV)*, 2018.

Olivier Bousquet, Sylvain Gelly, Ilya Tolstikhin, Carl-Johann Simon-Gabriel, and Bernhard Schöelkopf. From optimal transport to generative modeling: the VEGAN cookbook. In *arXiv preprint arXiv:1705.07642*, 2017.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. In *inproceedings on Computational Natural Language Learning (CoNLL)*, 2016.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International proceedings on Learning Representations (ICLR)*, 2018.

Alan Brunton, Timo Bolkart, and Stefanie Wuhrer. Multilinear Wavelets: A Statistical Shape Space for Human Faces. In *European proceedings on Computer Vision (ECCV)*, 2014a.

Alan Brunton, Augusto Salazar, Timo Bolkart, and Stefanie Wuhrer. Review of Statistical Shape Spaces for 3D data with Comparative Analysis for Human Faces. In *Computer Vision and Image Understanding (CVIU)*, 2014b.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders. In *International proceedings on Learning Representations (ICLR)*, 2016.

Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. FaceWarehouse: A 3D facial expression database for visual computing. In *IEEE Transactions on Visualization and Computer Graphics*, 2013.

Nicholas Carlini and David Wagner. Defensive Distillation is not Robust to Adversarial Examples. In *arXiv preprint arXiv:1607.04311*, 2016.

Nicholas Carlini and David Wagner. Magnet and "Efficient Defenses Against Adversarial Attacks" are not Robust to Adversarial Examples. In *arXiv preprint arXiv:1711.08478*, 2017a.

Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2017b.

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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative Pretraining from Pixels. In *International proceedings on Learning Representations (ICLR)*, 2020.

Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks Without Training Substitute Models. In *Proceedings of the ACM Workshop on Artificial Intelligence and Security*, 2017a.

Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples. In *Proceedings of the AAAI proceedings on Artificial Intelligence*, 2018.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. In *International proceedings on Learning Representations (ICLR)*, 2017b.

Yu Cheng, Zhe Gan, Yitong Li, Jingjing Liu, and Jianfeng Gao. Sequential Attention GAN for Interactive Image Editing. In *Proceedings of the ACM International inproceedings on Multimedia*, 2020.

Rewon Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. In *International proceedings on Learning Representations (ICLR)*, 2021.

Timothy F Cootes, Christopher J Taylor, David H Cooper, and Jim Graham. Active Shape Models–Their Training and Application. In *Computer Vision and Image Understanding (CVIU)*, 1995.

Timothy F Cootes, Gareth J Edwards, and Christopher J Taylor. Active Appearance Models. In *European proceedings on Computer Vision (ECCV)*, 1998.

Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active Appearance Models. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2001.

Mary Kathryn Cowles and Bradley P Carlin. Markov chain Monte Carlo Convergence Diagnostics: a Comparative Review. In *booktitle of the American Statistical Association*, 1996.

Ian Craw and Peter Cameron. Parameterising Images for Recognition and Reconstruction. In *Proceedings of the British Machine Vision inproceedings (BMVC)*. 1991.

Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael Black. Capture, Learning, and Synthesis of 3D Speaking Styles. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Bin Dai and David Wipf. Diagnosing and Enhancing VAE Models. In *International proceedings on Learning Representations (ICLR)*, 2019.

Hang Dai, Nick Pears, William AP Smith, and Christian Duncan. A 3D Morphable Model of Craniofacial Shape and Texture Variation. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2017.

Hang Dai, Nick Pears, and William Smith. A Data-augmented 3D Morphable Model of the Ear. In *International proceedings on Automatic Face & Gesture Recognition (FG)*, 2018.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and Controllable Face Image Generation via 3D Imitative-Contrastive Learning. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Emily L Denton, Soumith Chintala, arthur szlam, and Rob Fergus. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Yann LeCun Diederik P. Kingma, Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International proceedings on Learning Representations (ICLR)*, 2015.

Jeff Donahue and Karen Simonyan. Large Scale Adversarial Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial Feature Learning. In *International proceedings on Learning Representations (ICLR)*, 2017.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting Adversarial Attacks with Momentum. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Fabio Henrique Kiyoiti dos Santos Tanaka and Claus Aranha. Data Augmentation Using GANs. In *arXiv preprint arXiv:1904.09135*, 2019.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially Learned Inference. In *International proceedings on Learning Representations (ICLR)*, 2017.

Bernhard Egger, William AP Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, et al. 3D Morphable Face Models–Past, Present and Future. In *ACM Transactions on Graphics (TOG)*, 2020.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust Physical-world Attacks on Deep Learning Visual Classification. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an Animatable Detailed 3D Face Model from In-the-Wild Images. In *ACM Transactions on Graphics, (Proc. SIGGRAPH)*, 2021.

Samuel G Finlayson, Isaac S Kohane, and Andrew L Beam. Adversarial attacks against medical deep learning systems. In *arXiv preprint arXiv:1804.05296*, 2018.

Asja Fischer and Christian Igel. Empirical Analysis of the Divergence of Gibbs Sampling based Learning Algorithms for Restricted Boltzmann Machines. In *International inproceedings on artificial neural networks (ICANN)*, 2010.

Thomas Fischer. A Pyramid Vector Quantizer. In *IEEE transactions on information theory*, 1986.

Ysbrand Galama and Thomas Mensink. IterGANs: Iterative GANs to Learn and Control 3D Object Transformation. In *Computer Vision and Image Understanding (CVIU)*, 2019.

Ruiqi Gao, Erik Nijkamp, Diederik P Kingma, Zhen Xu, Andrew M Dai, and Ying Nian Wu. Flow Contrastive Estimation of Energy-based Models. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Baris Gecer, Binod Bhattarai, Josef Kittler, and Tae-Kyun Kim. Semi-supervised Adversarial Learning to Generate Photorealistic Face Images of new Identities from 3D Morphable Model. In *European proceedings on Computer Vision (ECCV)*, 2018.

Baris Gecer, Stylianos Ploumpis, Irene Kotsia, and Stefanos Zafeiriou. GANFIT: Generative Adversarial Network Fitting for High Fidelity 3D Face Reconstruction. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Zhenglin Geng, Chen Cao, and Sergey Tulyakov. 3D Guided Fine-grained Face Manipulation. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked Autoencoder for Distribution Estimation. In *International proceedings on Machine Learning (ICML)*, 2015.

Partha Ghosh, Arpan Losalka, and Michael J Black. Resisting Adversarial Attacks using Gaussian Mixture Variational Autoencoders. In *Proceedings of the AAAI proceedings on Artificial Intelligence*, 2019.

Partha Ghosh, Pravir Singh Gupta, Roy Uziel, Anurag Ranjan, Michael J. Black, and Timo Bolkart. GIF: Generative Interpretable Faces. In *International proceedings on 3D Vision (3DV)*, 2020a.

Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From Variational to Deterministic Autoencoders. In *International proceedings on Learning Representations (ICLR)*, 2020b.

Partha Ghosh, Dominik Zietlow, Michael J. Black, Larry S. Davis, and Xiaochen Hu. InvGAN: Invertible GANs. In *German Conference on Pattern Recognition (GCPR)*, 2022.

Aleksey Golovinskiy, Wojciech Matusik, Hanspeter Pfister, Szymon Rusinkiewicz, and Thomas Funkhouser. A Statistical Model for Synthesis of Detailed Facial Geometry. In *ACM Transactions on Graphics (TOG)*, 2006.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic Chemical Design Using a Data-driven Continuous Representation of Molecules. In *American Chemical Society Central Science*, 2018.

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

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International proceedings on Learning Representations (ICLR)*, 2015a.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International proceedings on Learning Representations (ICLR)*, 2015b.

Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. In *arXiv preprint arXiv:1412.5068*, 2014.

Shanyan Guan, Ying Tai, Bingbing Ni, Feida Zhu, Feiyue Huang, and Xiaokang Yang. Collaborative Learning for Faster StyleGAN Embedding. In *arXiv preprint arXiv:2007.01758*, 2020.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

An Guozhong. The Effects of Adding Noise During Backpropagation Training on a Generalization Performance. In *Neural Computation*, 1996.

Michael Gutmann and Jun-ichiro Hirayama. Bregman Divergence as General Framework to Estimate Unnormalized Statistical Models. In *arXiv preprint arXiv:1202.3727*, 2012.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep Into Rectifiers: Surpassing Human-level Performance on Imagenet Classification. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2015.

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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017a.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Nash Equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017b.

Alexander Hewer, Stefanie Wuhrer, Ingmar Steiner, and Korin Richmond. A Multilinear Tongue Model Derived from Speech Related MRI data of the Human Vocal Tract. In *Computer Speech & Language*, 2018.

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 *International proceedings on Learning Representations (ICLR)*, 2017.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. In *Neural computation*, 2002.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Matthew D Hoffman and Matthew J Johnson. Elbo Surgery: Yet Another Way to Carve up the Variational Evidence Lower Bound. In *Workshop in Advances in Approximate Bayesian Inference, NeurIPS*, 2016.

Liwen Hu, Derek Bradley, Hao Li, and Thabo Beeler. Simulation-ready Hair Capture. In *Computer Graphics Forum*, 2017.

Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely Connected Convolutional Networks. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International proceedings on Machine Learning (ICML)*, 2015.

Wengong Jin, Regina Barzilay, and T. Jaakkola. Junction Tree Variational Autoencoder for Molecular Graph Generation. In *International proceedings on Machine Learning (ICML)*, 2018.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual Losses for Real-time Style Transfer and Super-resolution. In *European proceedings on Computer Vision (ECCV)*, 2016.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International proceedings on Learning Representations (ICLR)*, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based Generator Architecture for Generative Adversarial networks. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019a.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and Improving the Image Quality of StyleGAN. In *arXiv preprint arXiv:1912.04958*, 2019b.

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 proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Nießner, Patrick Pérez, Christian Richardt, Michael Zollhöfer, and Christian Theobalt. Deep Video Portraits. In *ACM Transactions on Graphics (TOG)*, 2018.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International proceedings on Learning Representations (ICLR)*, 2014.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improving Variational Inference with Inverse Autoregressive Flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Ivan Kobyzev, Simon Prince, and Marcus Brubaker. Normalizing Flows: An Introduction and Review of Current Methods. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2020.

Marek Kowalski, Stephan J. Garbin, Virginia Estellers, Tadas Baltrušaitis, Matthew Johnson, and Jamie Shotton. CONFIG: Controllable Neural Face Image Generation. In *European proceedings on Computer Vision (ECCV)*, 2020.

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. In *Technical Report TR-2009, University of Toronto, Toronto*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. The GAN landscape: Losses, architectures, regularization, and normalization. In *arXiv preprint arXiv:1807.04720*, 2018.

Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar Variational Autoencoder. In *International proceedings on Machine Learning (ICML)*, 2017.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building Machines that Learn and Think like People. In *Proceedings of the Behavioral and brain sciences*, 2017.

Hugo Larochelle and Iain Murray. The Neural Autoregressive Distribution Estimator. In *Artificial Intelligence and Statistics (AISTATS)*, 2011.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based Learning Applied to Document Recognition. In *Proceedings of the IEEE*, 1998a.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998b. doi: 10.1109/5.726791.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. In *Nature Publishing Group*, 2015.

Chen-Yu Lee, Patrick W Gallagher, and Zhuowen Tu. Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree. In *Artificial Intelligence and Statistics (AISTATS)*, 2016.

Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a Model of Facial Shape and Expression from 4D Scans. In *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 2017.

Chieh Hubert Lin, Chia-Che Chang, Yu-Sheng Chen, Da-Cheng Juan, Wei Wei, and Hwann-Tzong Chen. CoCo-gan: Generation by Parts via Conditional Coordinating. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft CoCo: Common Objects in Context. In *European proceedings on Computer Vision (ECCV)*, 2014.

Zachary C Lipton and Subarna Tripathi. Precise Recovery of Latent Vectors from Generative Adversarial Networks. In *International proceedings on Learning Representations Workshop (ICLR-W)*, 2017.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into Transferable Adversarial Examples and Black-box Attacks. In *International proceedings on Learning Representations (ICLR)*, 2017.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2015.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *International proceedings on Learning Representations (ICLR)*, 2019.

Stephen Lombardi, Jason Saragih, Tomas Simon, and Yaser Sheikh. Deep Appearance Models for Face Rendering. In *ACM Transactions on Graphics (TOG)*, 2018.

Matthew M. Loper and Michael J. Black. OpenDR: An Approximate Differentiable Renderer. In *European proceedings on Computer Vision (ECCV)*, 2014.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are GANs Created Equal? A Large-Scale Study. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International proceedings on Learning Representations (ICLR)*, 2018.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial Autoencoders. In *International proceedings on Learning Representations (ICLR)*, 2016.

Richard T. Marriott, Safa Madiouni, Sami Romdhani, Stephane Gentric, and Liming Chen. An Assessment of GANs for Identity-related Applications. In *Proceedings of the IEEE International Joint inproceedings on Biometrics (IJCB)*, 2020.

Dongyu Meng and Hao Chen. Magnet: A Two-pronged Defense Against Adversarial Examples. In *ACM proceedings on Computer and Communications Security (Proc. SIGSAC)*, 2017.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which Training Methods for GANs do actually Converge? In *International proceedings on Machine Learning (ICML)*, 2018.

Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. In *arXiv preprint arXiv:1411.1784*, 2014.

Takeru Miyato and Masanori Koyama. cGANs with Projection Discriminator. In *International proceedings on Learning Representations (ICLR)*, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *International proceedings on Learning Representations (ICLR)*, 2018.

Seyed Mohsen Moosavi Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and Accurate Method to Fool Deep Neural Networks. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Koki Nagano, Jaewoo Seo, Jun Xing, Lingyu Wei, Zimo Li, Shunsuke Saito, Aviral Agarwal, Jens Fursund, Hao Li, Richard Roberts, et al. paGAN: Real-time Avatars Using Dynamic Textures. In *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 2018.

S Nayar, S Nene, and Hiroshi Murase. Columbia Object Image Library (coil 100). In *Department of Comp. Science, Columbia University, Tech. Rep. CUCS-006-96*, 1996.

Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia Object Image Library (coil-20). 1996.

Debanga R Neog, João L Cardoso, Anurag Ranjan, and Dinesh K Pai. Interactive Gaze Driven Animation of the Eye Region. In *Proceedings of the 21st International inproceedings on Web3D Technology*, 2016.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *Advances in Neural Information Processing Systems Workshop(NeurIPS-W)*, 2011.

Behnam Neyshabur, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro. Geometry of optimization and implicit regularization in deep learning. In *arXiv preprint arXiv:1705.03071*, 2017.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2019.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional Image Synthesis with Auxiliary Classifier GANs. In *International proceedings on Machine Learning (ICML)*, 2017.

Nicolas Papernot, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Fartash Faghri, Alexander Matyasko, Karen Hambardzumyan, Yi-Lin Juang, Alexey Kurakin, Ryan Sheatsley, et al. Cleverhans v2. 0.0: An Adversarial Machine Learning Library. In *arXiv preprint arXiv:1610.00768*, 2016a.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: From Phenomena to Black-Box Attacks Using Adversarial Samples. In *arXiv preprint arXiv:1605.07277*, 2016b.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings. In *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016c.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2016d.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Asia inproceedings on Computer and Communications Security (ASIACCS)*, 2017.

Frederick Ira Parke. A Parametric Model for Human Faces. Technical report, Utah University Salt-Lake City Department of Computer Science, 1974.

Emanuel Parzen. On Estimation of a Probability Density Function and Mode. In *The Annals of Mathematical Statistics*. Institute of Mathematical Statistics, 1962.

Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive Body Capture: 3D Hands, Face, and Body from a Single Image. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3D Face Model for Pose and Illumination Invariant Face Recognition. In *International inproceedings on Advanced Video and Signal Based Surveillance*, 2009.

Judea Pearl. Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, 1988. ISBN 978-0-08-051489-5. doi: https://doi.org/10.1016/B978-0-08-051489-5.50007-2.

Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M. Álvarez. Invertible Conditional GANs for image editing. In *Advances in Neural Information Processing Systems Workshop(NeurIPS-W)*, 2016.

Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial Latent Autoencoders. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Rama Chellappa Pouya Samangouei, Maya Kabkab. Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. In *International proceedings on Learning Representations (ICLR)*, 2018.

Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware Facial Animation From a Single Image. In *European proceedings on Computer Vision (ECCV)*, 2018.

Edward O Pyzer-Knapp, Changwon Suh, Rafael Gómez-Bombarelli, Jorge Aguilera-Iparraguirre, and Alán Aspuru-Guzik. What is High-throughput Virtual Screening? A Perspective from Organic Materials Discovery. In *Annual Review of Materials Research*, 2015.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In *International proceedings on Learning Representations (ICLR)*, 2016.

Vikram V. Ramaswamy, Sunnie S. Y. Kim, and Olga Russakovsky. Fair Attribute Classification through Latent Space De-biasing. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D Faces Using Convolutional Mesh Autoencoders. In *European proceedings on Computer Vision (ECCV)*, 2018.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. PyTorch3D. 2020.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating Diverse High-fidelity Images with VQ-VAE-2. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International proceedings on Learning Representations (ICLR)*, 2016.

Danilo Jimenez Rezende and Fabio Viola. Taming VAEs. In *arXiv preprint arXiv:1810.00597*, 2018.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International proceedings on Machine Learning (ICML)*, 2014.

Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in Style: a StyleGAN Encoder for Image-to-Image Translation. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive Auto-encoders: Explicit Invariance During Feature Extraction. In *International proceedings on Machine Learning (ICML)*, 2011.

Salah Rifai, Yoshua Bengio, Yann Dauphin, and Pascal Vincent. A Generative Process for Sampling Contractive Auto-encoders. In *International proceedings on Machine Learning (ICML)*, 2012.

Mihaela Rosca, Balaji Lakshminarayanan, and Shakir Mohamed. Distribution matching in variational inference. In *International proceedings on Learning Representations Workshop (ICLR-W)*, 2018.

Murray Rosenblatt. Remarks on Some Nonparametric Estimates of a Density Function. In *The Annals of Mathematical Statistics*. Institute of Mathematical Statistics, 1956.

Shunsuke Saito, Lingyu Wei, Liwen Hu, Koki Nagano, and Hao Li. Photorealistic Facial Texture Inference using Deep Neural Networks. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Shunsuke Saito, Liwen Hu, Chongyang Ma, Hikaru Ibayashi, Linjie Luo, and Hao Li. 3D hair Synthesis Using volumetric Variational Autoencoders. In *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 2018.

Mehdi S. M. Sajjadi, Bernhard Schölkopf, and Michael Hirsch. EnhanceNet: Single Image Super-Resolution Through Automated Texture Synthesis. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2017.

Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing Generative Models via Precision and Recall. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and other Modifications. In *International proceedings on Learning Representations (ICLR)*, 2017.

Soubhik Sanyal, Timo Bolkart, Haiwen Feng, and Michael J Black. Learning to Regress 3D Face Shape and Expression From an Image Without 3D Supervision. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Prasanna Sattigeri, Samuel C Hoffman, Vijil Chenthamarakshan, and Kush R Varshney. Fairness gan. In *arXiv preprint arXiv:1805.09910*, 2018.

Hannes Schulz, Andreas Müller, Sven Behnke, et al. Investigating convergence of restricted boltzmann machine learning. In *Advances in Neural Information Processing Systems Workshop(NeurIPS-W)*, 2010.

Aliaksei Severyn, Erhardt Barth, and Stanislau Semeniuta. A Hybrid Convolutional Variational Autoencoder for Text Generation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

Viktoriia Sharmanska, Lisa Anne Hendricks, Trevor Darrell, and Novi Quadrianto. Contrastive Examples for Addressing the Tyranny of the Majority. In *arXiv preprint arXiv:2004.06524*, 2020.

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the Latent Space of GANs for Semantic Face Editing. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Il-Kyu Shin, A Cengiz Öztireli, Hyeon-Joong Kim, Thabo Beeler, Markus Gross, and Soo-Mi Choi. Extraction and Transfer of Facial Expression Wrinkles for Facial Performance Enhancement. In *Pacific inproceedings on Computer Graphics and Applications*, 2014.

Jocelyn Sietsma and Robert JF Dow. Creating Artificial Neural Networks that Generalize. In *Neural networks*, 1991.

Lawrence Sirovich and Michael Kirby. Low-dimensional Procedure for the Characterization of Human Faces. In *Optical Society of America A*, 1987.

Ron Slossberg, Gil Shamai, and Ron Kimmel. High quality Facial Surface and Texture Synthesis via Generative Adversarial Networks. In *European proceedings on Computer Vision Workshops (ECCV-W)*, 2018.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *International proceedings on Machine Learning (ICML)*, 2015.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning Structured Output Representation using Deep Conditional Generative Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised MAP Inference for Image Super-resolution. In *International proceedings on Learning Representations (ICLR)*, 2017.

Yang Song and Diederik P Kingma. How to Train your Energy-based Models. In *arXiv preprint arXiv:2101.03288*, 2021.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International proceedings on Learning Representations (ICLR)*, 2018a.

Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Generative Adversarial Examples. In *arXiv preprint arXiv:1805.07894*, 2018b.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in the Wild. In *arXiv preprint arXiv1212.0402*, 2012.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. In *The booktitle of Machine Learning Research (JMLR)*, 2014.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing Properties of Neural Networks. In *International proceedings on Learning Representations (ICLR)*, 2014.

Esteban G Tabak and Eric Vanden-Eijnden. Density Estimation by Dual Ascent of the Log-likelihood. In *Proceedings of the Communications in Mathematical Sciences*, 2010.

Shufeng Tan and Michael L Mayrovouniotis. Reducing Data Dimensionality Through Optimizing Neural Network Inputs. *AIChE Journal*, 1995.

Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. StyleRig: Rigging StyleGAN for 3D Control over Portrait Images. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Lucas Theis and Matthias Bethge. Generative Image Modeling Using Spatial LSTMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. In *International proceedings on Learning Representations (ICLR)*, 2016.

Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Niessner. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred Neural Rendering: Image Synthesis Using Neural Textures. In *ACM Transactions on Graphics (TOG)*, 2019.

Andrey N Tikhonov and Vasilii Iakkovlevich Arsenin. *Solutions of Ill-posed Problems*. Society for Industrial and Applied Mathematics, Winston, Washington, DC, 1977.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein Auto-Encoders. In *International proceedings on Learning Representations (ICLR)*, 2017.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein Auto-Encoders. In *International proceedings on Learning Representations (ICLR)*, 2018.

Jakub Tomczak and Max Welling. VAE with a VampPrior. In *Artificial Intelligence and Statistics (AISTATS)*, 2018.

Soumya Tripathy, Juho Kannala, and Esa Rahtu. ICface: Interpretable and Controllable Face Reenactment Using GANs. In *Winter proceedings on Applications of Computer Vision (WACV)*, 2020.

George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein. REBAR: Low-Variance, Unbiased Gradient Estimates for Discrete Latent Variable Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Matthew Turk and Alex Pentland. Eigenfaces for Recognition. In *booktitle of Cognitive Neuroscience*, 1991.

Aaron van den Oord, Oriol Vinyals, et al. Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *International proceedings on Machine Learning (ICML)*, 2016.

M Alex O Vasilescu and Demetri Terzopoulos. Multilinear Analysis of Image Ensembles: Tensorfaces. In *European proceedings on Computer Vision (ECCV)*, 2002.

Evangelos Ververas and Stefanos Zafeiriou. SliderGAN: Synthesizing Expressive Face Images by Sliding 3D Blendshape Parameters. In *International journal of Computer Vision (IJCV)*, 2020.

Pascal Vincent. A Connection between Score Matching and Denoising Autoencoders. In *Neural computation*. MIT Press, 2011.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *International proceedings on Machine Learning (ICML)*, 2008.

D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face Transfer with Multilinear Models. In *ACM Transactions on Graphics, (Proc. SIGGRAPH)*, 2005.

Andrey Voynov and Artem Babenko. Unsupervised Discovery of Interpretable Directions in the GAN Latent Space. In *International proceedings on Machine Learning (ICML)*, 2020.

Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of Neural Networks Using Dropconnect. In *International proceedings on Machine Learning (ICML)*, 2013.

Lingyu Wei, Liwen Hu, Vladimir Kim, Ersin Yumer, and Hao Li. Real-time Hair Rendering Using Sequential Adversarial Networks. In *European proceedings on Computer Vision (ECCV)*, 2018.

Tianyi Wei, Dongdong Chen, Wenbo Zhou, Jing Liao, Weiming Zhang, Lu Yuan, Gang Hua, and Nenghai Yu. A Simple Baseline for StyleGAN Inversion. In *arXiv preprint arXiv:2104.07661*, 2021.

Wikipedia. Density estimation. *Wikimedia Foundation*, 2022a.

Wikipedia. Function (mathematics). *Wikimedia Foundation*, 2022b.

Chenglei Wu, Derek Bradley, Pablo Garrido, Michael Zollhöfer, Christian Theobalt, Markus H Gross, and Thabo Beeler. Model-based Teeth Reconstruction. In *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 2016.

Dan Wu, Jiasong Wu, Rui Zeng, Longyu Jiang, Lotfi Senhadji, and Huazhong Shu. Kernel Principal Component Analysis Network for Image Classification. In *arXiv preprint arXiv:1512.06337*, 2015.

Wayne Wu, Yunxuan Zhang, Cheng Li, Chen Qian, and Chen Change Loy. Reenactgan: Learning to Reenact Faces via Boundary Transfer. In *European proceedings on Computer Vision (ECCV)*, 2018.

Jonas Wulff and Antonio Torralba. Improving Inversion and Generation Diversity in Stylegan Using a Gaussianized Latent Space. In *arXiv preprint arXiv:2009.06529*, 2020.

Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN Inversion: A Survey. In *arXiv preprint arXiv:2101.05278*, 2021.

Yinghao Xu, Yujun Shen, Jiapeng Zhu, Ceyuan Yang, and Bolei Zhou. Generative Hierarchical Features from Synthesizing Images. In *Proceedings of the IEEE proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. In *Proceedings of the British Machine Vision inproceedings (BMVC)*, 2016.

Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot Adversarial Learning of Realistic Neural Talking Head Models. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2019.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding Deep Learning Requires Rethinking Generalization. In *International proceedings on Learning Representations (ICLR)*, 2017a.

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 *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2017b.

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 proceedings on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Wenlong Zhang, Yihao Liu, Chao Dong, and Yu Qiao. Ranksrgan: Generative Adversarial Networks with Ranker for Image Super-Resolution. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, pages 3096–3105, 2019.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. Towards Deeper understanding of variational autoencoding models. In *arXiv preprint arXiv:1702.08658*, 2017.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating Natural Adversarial Examples. In *International proceedings on Learning Representations (ICLR)*, 2018.

Jiapeng Zhu, Deli Zhao, and Bo Zhang. LIA: Latently Invertible Autoencoder with Adversarial Learning. In *arXiv preprint arXiv1906.08090*, 2019.

Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain GAN Inversion for Real Image Editing. In *European proceedings on Computer Vision (ECCV)*, 2020.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative Visual Manipulation on the Natural Image Manifold. In *European proceedings on Computer Vision (ECCV)*, 2016.

Dominik Zietlow, Michael Lohaus, Guha Balakrishnan, Matthäus Kleindessner, Francesco Locatello, Bernhard Schölkopf, and Chris Russell. Leveling Down in Computer Vision: Pareto Inefficiencies in Fair Deep Classifiers. *arXiv preprint arXiv:2203.04913*, 2022.

Silvia Zuffi, Angjoo Kanazawa, Tanya Berger-Wolf, and Michael J Black. Three-D Safari: Learning to Estimate Zebra Pose, Shape, and Texture From Images. In *Proceedings of the IEEE International inproceedings on Computer Vision (ICCV)*, 2019.

MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS