Refining Hand Pose Training Data with Adversarial Autoencoder Networks

Lucas Murphy

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

A dissertation presented in part fulfilment of the requirements of the Degree of Master of Science at The University of Glasgow

12/12/21

Abstract

Many supervised machine learning tasks require non-trivial labelling of data, where it is disadvantageous to annotate conventionally. RGB hand pose training image data is difficult for humans to annotate due to complex configurations of hands with several ambiguities with regards to scaling, occlusion, camera positioning and 3D inference. A common solution is to use simulated images of hands, as data can be acquired en masse with perfect annotations, leading to higher accuracy and throughput. One disadvantage of this approach is that simulated data may not be representative of images of hands encountered in reality. Generative Adversarial Networks (GANs) are a relatively recent development which can generate data that could belong to a target distribution. Several adversarial autoencoder networks were developed to transform simulated images of hands into more realistic images, while maintaining useful annotation. The contribution of this thesis is a CycleGAN variant which uses a cycle consistency loss in the discriminators' convolutional feature space, which promotes a significantly more realistic and diversified distribution of outputs according to a GAN evaluation phase, with better hand pose estimation performance when compared with other methods. While hand pose performance did not beat a network trained on the original data, the project has outlined several methods which could improve results in future research.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: Lucas Murphy Signature: 

Acknowledgements

I'd like to thank Dr. Kevin Bryson for his support throughout the project, even into the night when he was missing parties. I'd finally like to extend my thanks to everyone who asked what my thesis was about for their patience.

Contents

Chapter 1

Introduction

1.1 Context

A significant challenge when training supervised machine-learning (ML) models is the acquisition of data, with several important criteria needing fulfilled.

1. Datasets need to be of sufficient size to enable models to generalise well. Having too few training samples can lead to overfitting.
2. Data should be generalisable to enable models to cope with a variety of *in the wild* examples.
3. Training data ideally should be exactly representative of the data encountered in the wild.
4. Annotative labels associated with raw data should be accurate and precise.

For many supervised ML tasks, such as classifying dogs and cats, traditional acquisition techniques are sufficient. However, for more nuanced problems, acquisition can be much more difficult. Regression problems which require humans to annotate data can be more difficult to gather useful data for. Examples of such tasks are eye-tracking [21] or autonomous navigation. The problem investigated in this project is hand-pose estimation.

Hand-pose estimation has many applications, such as remote robotic control, sign language inference, and hand-tracking in virtual reality. The benefits of visual as opposed to mechanical input systems (such as glove based techniques) include lower costs and higher convenience, robustness, and reliability. The computer vision task is to take an image of a hand as input and output a representation of the hand in terms of locations. This representation may be the 2D or 3D coordinates of certain keypoints of the hand. It is typical for hands to be represented with 21 keypoints. The detailed problem is ill posed, and struggles to achieve all of the four communicated acquisition criteria. The manual annotation of 21 3D keypoints is a very laborious task for humans to complete, leading to a reduction in labelling throughput compared with simpler tasks. While markers could be placed on the hand to make the annotation task easier, they are not suitable for markerless estimation as the model would learn to look for the markers, not for the hand keypoint in question. There is also the issue of scale and depth ambiguity when annotating 2D RGB images. Human annotators will have to infer locations in the Z dimension. This issue is minimised when using depth

images, however, depth cameras are an expensive commodity with few owners in the real world. By contrast, RGB cameras are widely available in all modern phones and webcams. Depth cameras are also less effective in certain lighting conditions and limited to an effective distance range from the target [32]. Both RGB and depth cameras suffer from issues with occlusion, with human annotators having to infer occluded keypoints. This issue can be minimised with a multi-camera setup from multiple angles, however this is not without additional complexity and expense. The result of human annotation for such a task is small datasets with annotations subject to human error.

A solution to the above acquisition problem is to make use of simulated data, as it directly addresses many of the aforementioned issues when using real images. It requires little effort to generate many different hand-poses from many camera angles. Regardless of lighting conditions, occlusions or distances from the simulated camera, every image can have unambigious annotations, as they are provided by the simulation engine as opposed to being inferred from the image itself, sidestepping the limitations of human annotators.

An issue with simulated data is that it may not be exactly representative of the data encountered in the wild. The lighting, colouring, texture, etc. may not be consistent with real-life images. Generative Adversarial Networks (GANs) [6], have been proven to effectively generate realistic looking data which could have come from a target distribution, but their often deeply entangled latent space means obtaining useful annotations from GAN generated data alone is very difficult. Autoencoders have been shown to effectively reduce data to a lower dimensional latent space while preserving representative features, such that inputs can be converted from one form to another (low resolution to high resolution images, grayscale to colour images, etc.). Previous works have demonstrated hybrid Autoencoder/Adversarial architectures which can effectively convert data from one form to a realistic new form, while preserving useful annotative features of the input. This project investigates the effectiveness of several developed Autoencoder/Adversarial network architectures in generating realistic training data for the hand-pose estimation task.

1.2 Aim and Objectives

The aim of the project is to devise, develop, and evaluate a network architecture which takes a synthetic image of a hand as input, and output a more realistic appearing image of the hand, such that the associated annotative information is still useful in training a hand-pose estimator. The refined training data will be evaluated by training a hand-pose estimator, running a test set of human-annotated real captures of hands through the hand-pose estimator, and comparing the performance when trained with original synthetic data, vs refined synthetic data.

The high-level objectives for the project are as follows:

1. Carry out literature review on the current state-of-the-art with regards to adversarial networks, autoencoders, and hand-pose estimation.
2. Develop adversarial encoder architecture which converts synthetic images into truer-to-life images while maintaining useful annotative information (pose info).
3. Evaluate refined data using several metrics, including performance when used to train a hand-pose estimator.

Chapter 2

Survey

2.1 Background of Methods

2.1.1 Autoencoders

Autoencoder (AE) architectures encode data into a lower dimensional code and then decode the low dimensional representation into a higher dimensional output [5, pp. 499-523]. They can be used for compression, or for tasks which add information, such as colourisers or resolution upscalers. Word embeddings are an example of using autoencoders to encode the lower dimensional semantics of language. A key component of the autoencoder architecture is the element-wise comparison between the prediction and the target. In the context of images, these elements refer to pixels. A typical image autoencoder may minimise the L_2 norm between the input and predicted output pixels with the cost function of Equation 2.1.

$$J(\theta) = \left\| x' - x \right\|_2 \tag{2.1}$$

where:
$J(\theta)$ = loss over model parameters θ
x = input vector of image pixels
x' = output vector of image pixels

2.1.2 Variational Autoencoders

Variational Autoencoders (VAEs) are a branch of autoencoders with an inbuilt regulariser [10]. They attempt to tackle a potential issue brought forth with base autoencoders, which is the difficulty in learning a latent manifold of a distribution. With traditional autoencoders, models learn to replicate the target exactly (a single data point). This can cause the model to not learn generalisable representations of the target distribution, but rather overfit the model to the samples provided. This can mean that two targets can be adequately learned, but the model cannot adequately represent a data point interpolated between the learned targets. The curse of dimensionality only exacerbates this problem.

Instead of encoding a latent vector representation, VAEs attempt to circumvent the above issue by encoding the parameters associated with a distribution, and then sample a point according to this learned distribution (typically a normal distribution), which is then decoded. This sampling reduces the ability for the model to overfit the data, and ideally allows the model to better generalise by sampling many points along the latent manifold. To constrain the model, an added regularisation term, the Kullback-Leibler (KL) divergence, penalises the model for encoding distribution parameters which vary from a target distribution (typically a standard normal). A typical image VAE cost function is shown in Equation 2.2.

$$J(\theta) = \left\|x' - x\right\|_2 + D_{KL}(p(z|x)||\mathcal{N}(0,1)) \tag{2.2}$$

where:

$J(\theta)$ = loss over model parameters θ
x = input vector of image pixels
x' = output vector of image pixels
$D_{KL}()$ = Kullback-Leibler Divergence function
$p(z|x)$ = probability density over latent vector z for a given x
$\mathcal{N}(0,1)$ = Standard normal of same dimensional size as z

2.1.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) were introduced in 2014 [6] as a solution to the problem of generating realistic data from a low dimensional latent representation. Generative models often undergo supervised training by comparing the prediction to a target, and minimising this difference. In image colourisation, the difference between the pixel values of a colourised prediction and that of the coloured target is minimised. In resolution upscaling, the pixels of the high-resolution predictions may be directly compared with the high-resolution target. GANs seek to generate realistic data of a target distribution with a low dimensional noise input, and thus have no target from which to compare and minimise with directly. Rather, GANs learn their own cost function as part of their training process, and thus learn to represent higher-level, abstract features of the target distribution in a low-dimensional latent manifold.

The training process of a GAN was proposed by Goodfellow as a game between two networks - a Generator G and a Discriminator D. The goal of G is to generate realistic data from a noise distribution p_{noise} (typically a standard normal) which fools D into believing the generated data belongs to the real data distribution p_{data}. D attempts to discern between data originating from the data distribution p_{data} and noise distribution p_{noise}. As both networks train in tandem, they improve each other by iteratively exploiting the weaknesses of their adversary. The ideal end state of training is G generates data which is indistinguishable from the real distribution, and D can only guess which samples are real and which are fake. A diagram of the training process is shown in Figure 2.1. The process is described by Goodfellow as the minimax game of Equation 2.3.

$$\min_G \max_D J(\theta_G, \theta_D) = E_{x \sim p_{data}}[log(D(x)] + E_{z \sim p_{noise}}[log(1 - D(G(z)))] \tag{2.3}$$

where:

$J(\theta_G, \theta_D)$ = objective over generator and discriminator parameters θ_G, θ_D
p_{data} = data distribution
p_{noise} = noise distribution
E = Expectation
D = Discriminator function
G = Generator function

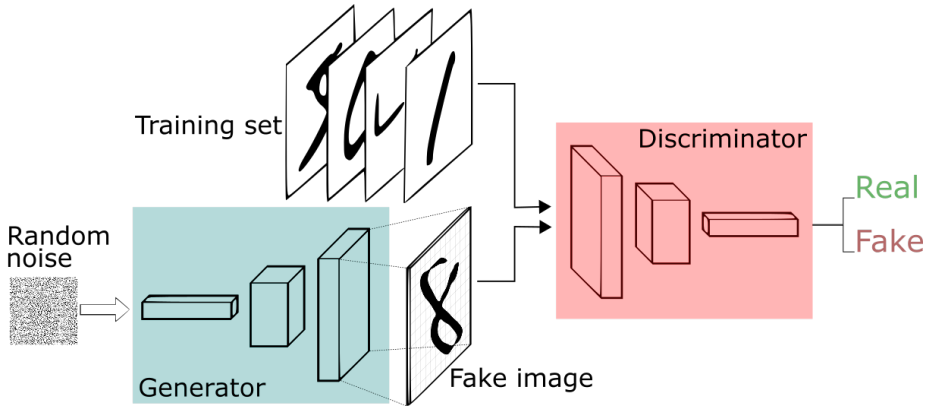

Figure 2.1: Generative Adversarial Network training process [22, Figure source].

2.1.4 VAE/GAN Hybrid Architectures

One difficulty with AE and VAE architectures is that while they can effectively minimise an element-wise cost function (such as L_1 or L_2 loss) between a prediction and target, they tend to blur high frequency texture details in order to obtain high peak signal-to-noise ratios (PSNRs) [12]. GANs have the benefits of minimising a 'perceptual' loss, as they learn higher-level, abstract features as opposed to features at the pixel level.

VAE/GAN architectures take components of the VAE and GAN and combine them into one model. The VAE encodes a latent distribution which is then sampled from. The decoder and generator are one in the same, and decode the latent sample. The cost function for this hybrid is the conjunction of the traditional VAE loss and an adversarial loss. A useful property of this hybrid is that GANs have been shown to train well when latent noise is sampled from a standard isotropic Gaussian, and VAEs enforce their latent representations to be standard isotropic Gaussians. The network overview is shown in Figure 2.2.

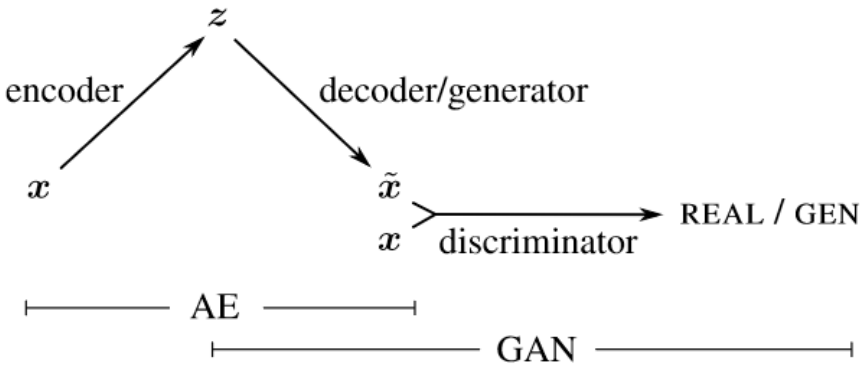

Figure 2.2: Overview of the Variational Autoencoder / Generative Adversarial Network architecture [11, Figure source].

2.1.5 CycleGANs

CycleGANs are a variant of generative adversarial networks which convert images from one style to another while preserving their *content* [30]. They are made up of two GAN architectures. One GAN converts an input from style A to style B, and the second GAN converts an input from style B to style A. To preserve the important content of the inputs, each GAN generator's output is fed into the other GAN generator as input, with that generator's output ideally being very similar to the original input. This constrains the generators to output images which fool their respective discriminators while maintaining the useful content of the original inputs. The training process is described by Figure 2.3.

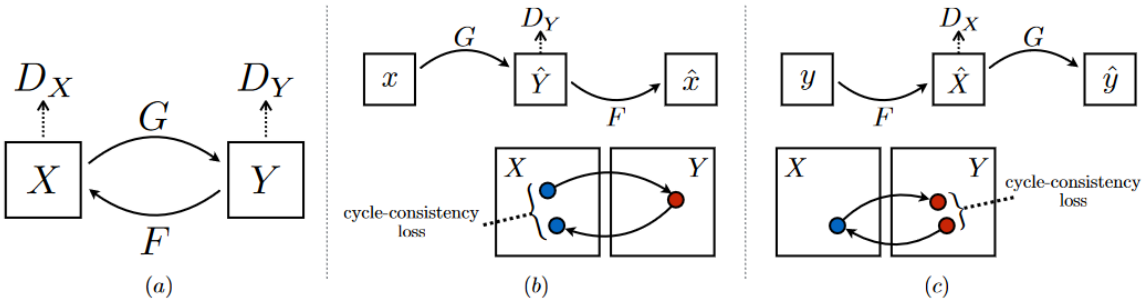

Figure 2.3: Training process for CycleGANs. There are two mapping functions, F and G, which convert inputs into their respective styles. As part of the training process, the models are constrained by mapping conversions back to their initial style, and minimising the cycle-consistency loss (such as L_1 norm), between the initial input and the cycled output [30, Figure source].

2.2 Related Work

Since their inception in 2014, GANs have drawn considerable attention from the academic community, driving research into the use of adversaries as part of training processes. In late 2014, a team demonstrated how class labels could be used to condition GANs to generate specific class images, as well as minimise issues with mode collapse [15]. One paper reported using an adversarial network to produce realistic human pose predictions [9]. In 2016 a VAE/GAN architecture was proposed which uses a discriminator intermediate feature map as reconstruction loss [11], which was found to outperform VAEs in terms of visual fidelity. A large body of work has made use of adversarial networks to upscale images, as it overcomes issues with blurry texture when using pixel-wise reconstruction error [12, 26, 18]. Several teams have also made use of discriminator intermediate feature map as reconstruction loss [12, 26], and concluded this was successful in recreating plausible high frequency details. Another team made use of adversarial networks to refine simulated data for eye-tracking and hand-pose prediction computer vision tasks [21], and concluded that performance with real-life data improved when compared with a network trained on the raw simulated data. They evaluated the perceptual realism of images using a Visual Turing Test with humans deciding if images are real or fake. The team also regularised outputs by penalising large pixel-level differences between synthetic and refined images. The journal also proposed training with a history of refined images to prevent mode collapse, and having the discriminator determine realism individually on a grid of patches over an image, as opposed to an overall determination for the complete image. These contributions have been integral to recent works [8], especially CycleGAN [30], which has demonstrated qualitative state-of-the-art results in style conversion without paired examples.

Adversarial networks are notoriously difficult to evaluate, as human based evaluation can be biased towards image quality and not expose issues such as mode collapse [27]. An empirical study was carried out to determine which metrics are useful in quantatitively evaluating GANs [27]. The study found that most useful metrics are with respect to a separate, independent and comprehensive network, such as ResNet-34. Works have reported developing their own metrics of evaluation which are specific to the problem being aimed to solve. A UCL/Cambridge bioinformatics team developed their own metrics of assessing the realism of artificial gene expression data, as empirical evaluation is not intuitive in high dimensional space [25].

Several works have investigated the use of depth images to estimate hand pose [24, 23, 17]. In the field of RGB hand pose prediction, multiple works predict hand pose as part of a pipeline predicting entire body pose [28, 20]. Zimmerman et al. have developed multiple pipelines which take an RGB image as input, segment the hand, infer the pixel locations of 21 keypoints in the image and then lift those keypoints into 3 dimensions [32, 31]. They developed two datasets of real hands with human annotation and simulated hands with engine annotation respectively [2, 32], and used both to train their model. They also predict pose in a scale and translation invariant frame. A team developed a superior model by utilising a kinematic model of the hand in their pipeline, and also by passing their synthetic data through a modified CycleGAN to create more realistic hands [16]. Their modified generator outperforms traditional CycleGAN by utilising a geometric loss. In 2020, Google Research released a low-overhead, real-time hand tracking pipeline [29]. While their training process does mention the use of real and synthetic data, it is unclear whether GANs were used to refine said data.

Chapter 3

Experiments

3.1 Software and Hardware

The project throughout was completed in Python 3.7, with the majority of work being conducted with PyTorch. Some parts of the evaluation utilised TensorFlow 1.3.

The workload was split between Google Colab Pro+ machines and a workstation PC with an RTX 3070 FE, Ryzen 3600 and 16 GB of RAM.

3.2 Datasets

Two datasets were used through the project, the FreiHAND Dataset [2], which is the real distribution, and Rendered Hand Dataset [32], which is the synthetic distribution. Both datasets were produced by Zimmerman et al. at the University of Freiburg, and come packaged with masks of the hands, and the annotation of 21 keypoints of each hand.

Samples of each dataset are shown in Figure 3.1.

3.2.1 Preprocessing

There were several preprocessing steps carried out to make the datasets consistent with each other. Both datasets saw tighter crops around the hands. As the real dataset has already been processed to have the hands take center of images, this only required a center crop. The synthetic dataset, however, could have hands appear anywhere in the frame, large or small, or be omitted from the frame entirely. A cropping algorithm was devised which compares the masks of each hand and crops around the one with the highest variance (based on the assumption that a higher variance implies a larger (and therefore clearer) picture of a hand). Further algorithm details can be found in GitHub repository linked in Appendix A.1. The backgrounds of both datasets were masked out to maintain consistency and reduce workload on the generators. As the real dataset contains only Caucasian skin, whereas the synthetic dataset contains a variety of skin colours, the synthetic hands

Figure 3.1: Samples drawn from the FreiHAND [2] and Rendered Hand [32] Datasets.

were normalised such as they were devoid of colour, as this was easier for the generators to apply appropriate modifications. Lastly, all images were resized to 64x64 pixels and normalised.

Samples of the preprocessed datasets are shown in Figure 3.2.

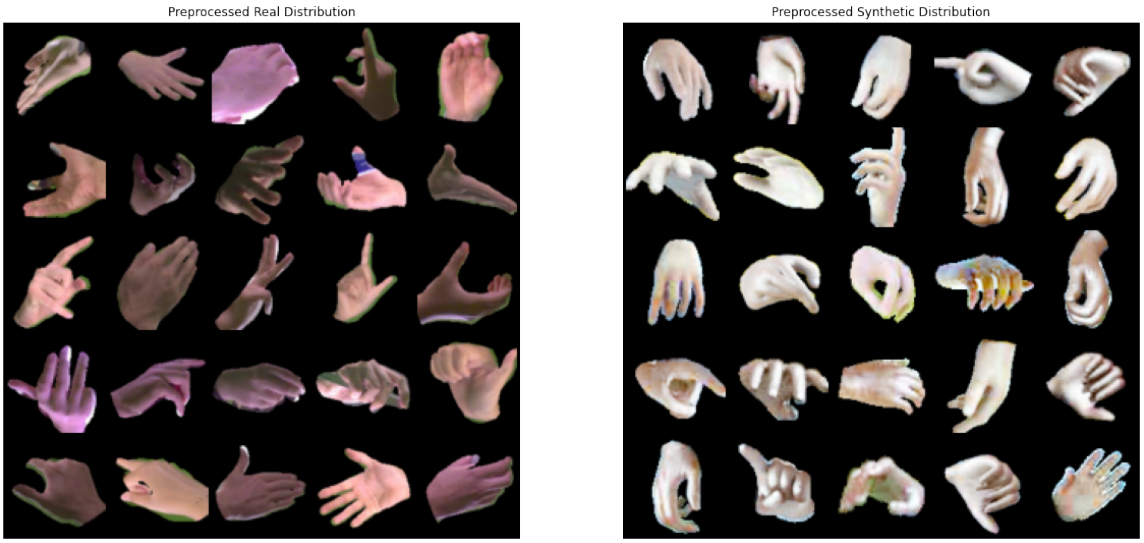

Figure 3.2: Samples drawn from the preprocessed FreiHAND [2] and Rendered Hand [32] Datasets.

3.3 Adversarial Architectures

Several autoencoder architectures were developed to transform synthetic images of hands into more life-like equivalents. The architectures of the models evaluated are detailed below. It was not possible to fine-tune hyperparameters as the only useful quantitive measure to train for is hand-pose prediction performance. This would require training the Autoencoder and GAN network, then generating a dataset of refined hands, then training an independent hand pose estimator network with the dataset. Due to time and computing constraints of Google Colab it was not feasible to retrain the pipeline several times. Instead, hyperparameters were based on the the advice and research in Section 2.2. Also, as developing a full-fledged hand pose estimator was outside the scope of this project, the source code from Zimmerman et al. [32] was reverse engineered and used. Adam optimizers were used to train every network with learning rate 0.0002 and Beta values 0.5 and 0.999 respectively. Adam was chosen as it has been shown to be resistant to initial hyperparameter configurations [19].

3.3.1 VAEGAN

The architecture for the Variational Encoder-Decoder/Generator is as demonstrated in Figure 3.3, and the Discriminator architecture as demonstrated in Figure 3.4. The VAE and GAN weights were each initialised separately in a pretraining phase, as per the advice of Larson et al. [11]. The VAE and GAN were pretrained for 30 epochs. This was required as the VAE and GAN separately had stable convergence, but the VAEGAN together was very unstable and diverged easily. Samples from the model are shown in Figure 3.5.

For pretraining the GAN, the generator was fed a 128 element vector of standard normal noise which was decoded as in Figure 3.3. The loss functions the generator G and discriminator D minimise are shown in Equations 3.1 and 3.2.

$$J_G(\theta_G, \theta_D) = 1 - D(G(z)) \tag{3.1}$$

$$J_D(\theta_G, \theta_D) = (1 - D(x)) + D(G(z)) \tag{3.2}$$

where:

$J(\theta_G, \theta_D)$ = objective function over generator and discriminator parameters θ_G, θ_D.
$x \sim$ real data distribution $p_{data(real)}$
$z \sim$ noise distribution p_{noise}
D = Discriminator function
G = Generator function

The VAE was pretrained by minimising the pixel wise difference between synthetic and real images alike. Both datasets were normalised to remove skin pigment. The loss function of Equation 2.2 was minimised.

Following the pretraining phase, the VAEGAN was trained for 5 epochs using normalised real data inputs and unnormalised real data targets. The encoder weights were frozen such that encodings were still generalised for synthetic inputs. The decoder aimed to minimise the L_2 loss between the first layer of discriminator feature maps between the input and target, as per the workings of Larson et al. [11]. They also reported higher stability when the generator was also fed Gaussian noise, and therefore this was adopted in training.

The loss functions minimised by the discriminator D and autoencoder/generator G are shown in respectively in Equations 3.3 and 3.4.

$$J_D(\theta_D, \theta_G) = (1 - D(x)) + (D(G(x))) + (D(G(z))) \tag{3.3}$$

$$J_G(\theta_D, \theta_G) = (1 - D(G(x))) + (1 - D(G(z))) + \lambda \left\| D_m(G(x)) - D_m(x) \right\|_2 \tag{3.4}$$

where:

$J(\theta_G, \theta_D)$ = objective function over generator and discriminator parameters θ_G, θ_D
$x \sim$ real data distribution $p_{data(real)}$
$z \sim$ noise distribution p_{noise}
D = Discriminator function
G = Generator function
D_m = first discriminator feature map
λ = tuning parameter ($6e^{-3}$ in experiments)

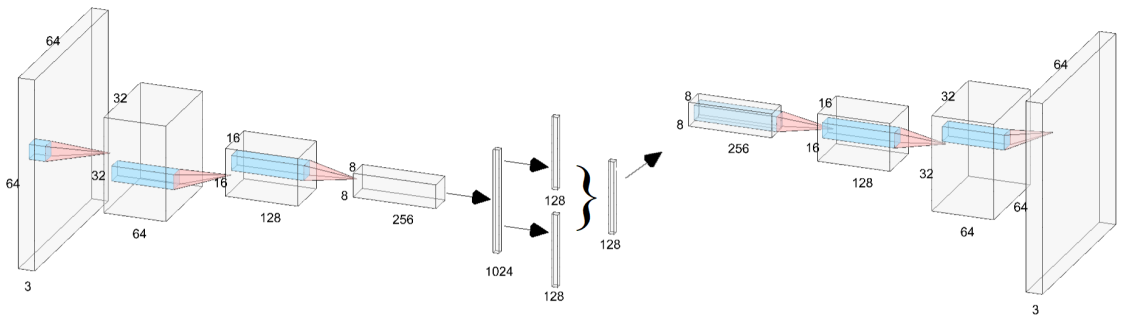

Figure 3.3: Variational Autoencoder architecture in VAEGAN hybrid. The first half encodes an image into a 128 length mean and log variance. These parameter vectors are then used to sample from a 128 dimensional Gaussian, and decoded into an RGB image. Every convolution operation is followed by a batch normalisation (momentum 0.9) and leaky ReLU (negative slope of 0.2), except the last, which is followed by the hyperbolic tangent. Created using NN-SVG [13].

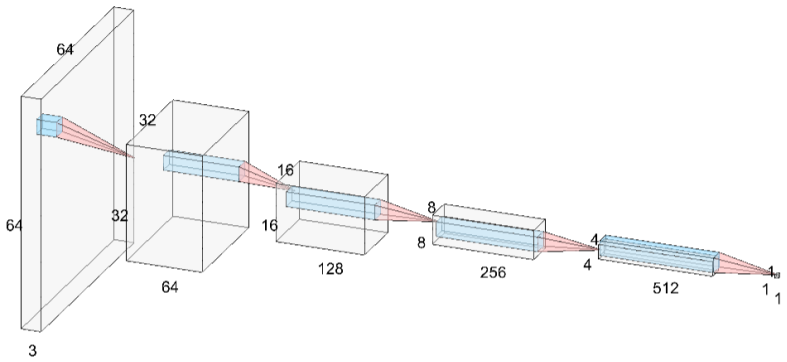

Figure 3.4: Discriminator architecture in VAEGAN hybrid. Every convolution operation is followed by a batch normalisation (momentum 0.9) and leaky ReLU (negative slope of 0.2), except the last, which is followed by the sigmoid. Created using NN-SVG [13].

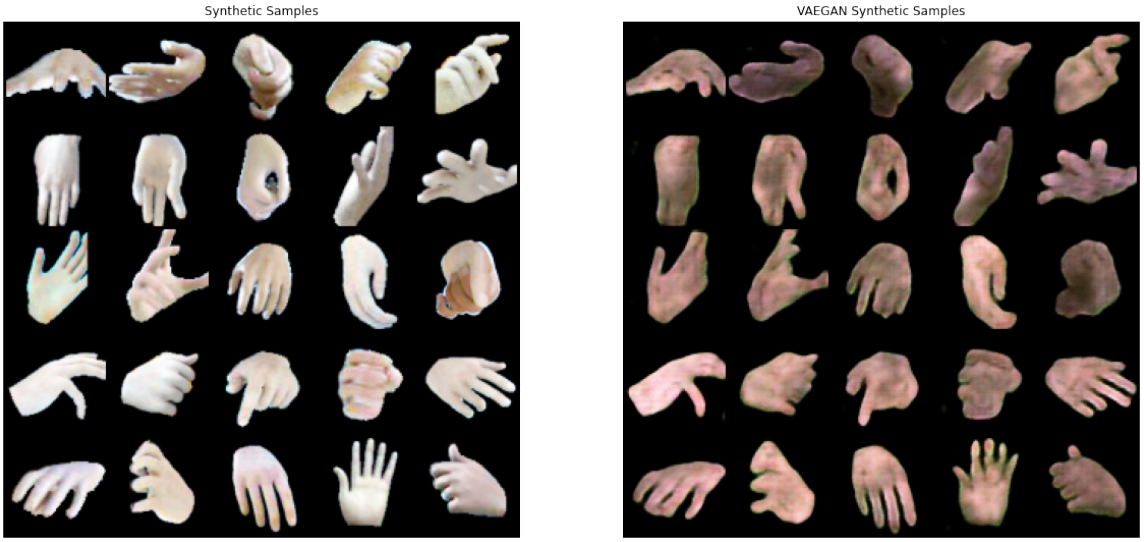

Figure 3.5: Synthetic inputs and outputs with the VAEGAN model.

3.3.2 Authentic CycleGAN

The CycleGAN architectures were developed as the VAEGAN was deemed to not produce high quality images when assessed qualitatively. The bottleneck appeared to be the VAE, which produced blurry outputs with a lack of high-frequency textures even when the loss function was pixel wise difference. The architecture was also very shallow, with resolution being reduced by $\frac{1}{4}$ at every convolutional layer. The training process also did not allow for the use of synthetic data following the pretraining phase, as they could not be used as a supervised target else the model would learn to generate images that look like synthetic hands. CycleGAN was found to address both of these issues. CycleGANs were also reported to work well with texture altering tasks and do not optimise for geometric alteration.

The Authentic CycleGAN largely follows the architecture set out in the CycleGAN paper, but with modifications for a lower resolution dataset. The autoencoder and patch discriminator architectures for the CycleGANs are shown respectively in Figures 3.6 and 3.7. A patch discriminator is adopted as in the CycleGAN paper [30] such that the discriminator assesses the realism of each respective patch of the image. Samples from the model are shown in Figure 3.8. The model was trained from scratch for 30 epochs.

The discriminators D_X, D_Y were optimised by minimising the loss function of Equation 3.5. The discriminator loss was halved to slow discriminator training relative to the generator, as per Zhu et al. [30]. The generators F and G minimised both an adversarial loss and cycle consistency loss (shown in Equation 3.6), culminating in the loss function of Equation 3.7.

$$J_D(\theta_G, \theta_F, \theta_{DX}, \theta_{DY}) = \frac{1}{2}(\|1 - D_X(x)\|_2 + \|D_X(F(y))\|_2 + \|1 - D_Y(y)\|_2 + \|D_Y(G(x))\|_2) \tag{3.5}$$

$$J_{cyc}(\theta_G, \theta_F) = \|x - F(G(x))\|_1 + \|y - G(F(y))\|_1 \tag{3.6}$$

$$J_{G,F}(\theta_G, \theta_F, \theta_{DX}, \theta_{DY}) = \|1 - D_X(F(y))\|_2 + \|1 - D_Y(G(x))\|_2 + \lambda J_{cyc}(\theta_G, \theta_F) \tag{3.7}$$

where:

$J(\theta)$ = objective function over stated parameters
$x \sim$ real data distribution $p_{data(real)}$
$y \sim$ synthetic data distribution $p_{data(synth)}$
D_X = real Discriminator function
D_Y = synthetic Discriminator function
G = Generator which converts real to synthetic
F = Generator which converts synthetic to real
D_m = first discriminator feature map
λ = tuning parameter (10 used in experiments)

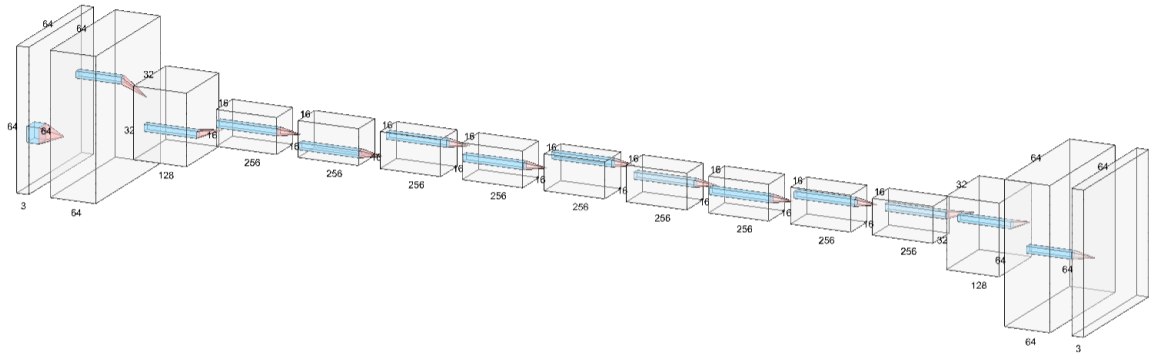

Figure 3.6: CycleGAN autoencoder architecture. The feed forward process is a convolution downscaling to 16x16x256 tensor, followed by a residual convolution phase and then an upscaling to an RGB image. Each convolution operation is followed by instance normalisation and leaky ReLU (negative slope 0.2), except the last, which is followed by the hyperbolic tangent. Created using NN-SVG [13].

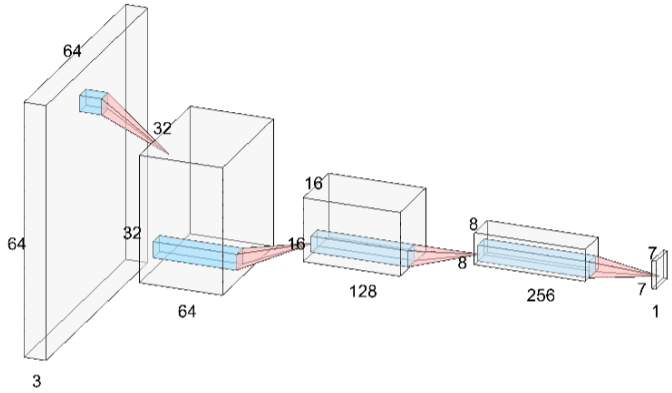

Figure 3.7: Patch Discriminator architecture for CycleGANs. Every convolution operation is followed by a batch normalisation (momentum 0.9) and leaky ReLU (negative slope of 0.2), except the last, which is followed by the sigmoid. Created using NN-SVG [13].

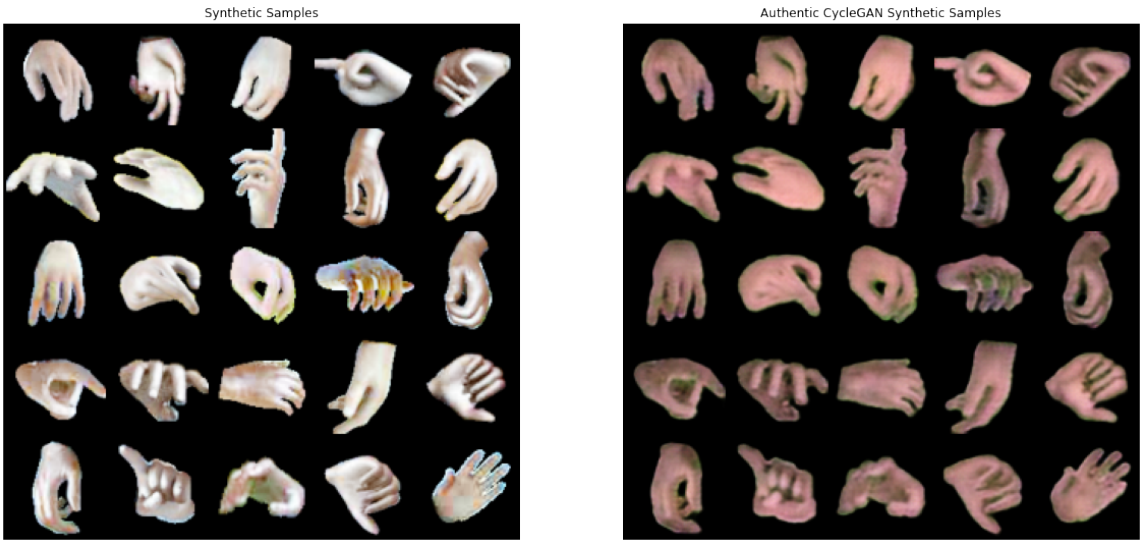

Figure 3.8: Synthetic inputs and outputs with the Authentic CycleGAN model.

3.3.3 Discriminator Feature Map CycleGAN

The Discriminator Feature Map CycleGAN has the same autoencoder and discriminator architectures as shown in Figures 3.6 and 3.7, but replaces the pixel-wise reconstruction error with a high-level discriminator feature map reconstruction error. The feature map following the first discriminator convolution was used for the reconstruction error. The logic behind this design decision was that all aspects of the model's loss function are with respect to the discriminators, which will optimise reconstructions to also look realistic to its respective discriminator. As feature map resolution reduces the deeper into the discriminator, the first layer was chosen to maintain reasonable resolution. Samples from the model are shown in Figure 3.9.

The discriminators D_X, D_Y were optimised by minimising the same loss function of Equation 3.5. The cycle consistency loss function was modified as in Equation 3.8, resulting in the new loss function for generators G and F shown in Equation 3.9.

$$J_{cyc}(\theta_G, \theta_F, \theta_{DX}, \theta_{DY}) = \|D_{Xm}(x) - D_{Xm}(F(G(x)))\|_1 + \|D_{Ym}(y) - D_{Ym}(G(F(y)))\|_1 \tag{3.8}$$

$$J_{G,F}(\theta_G, \theta_F, \theta_{DX}, \theta_{DY}) = \|1 - D_X(F(y))\|_2 + \|1 - D_Y(G(x))\|_2 + \lambda J_{cyc}(\theta_G, \theta_F, \theta_{DX}, \theta_{DY}) \tag{3.9}$$

where:

$J(\theta)$ = objective function over stated parameters
$x \sim$ real data distribution $p_{data(real)}$
$y \sim$ synthetic data distribution $p_{data(synth)}$
D_X = real Discriminator function
D_Y = synthetic Discriminator function
G = Generator which converts real to synthetic
F = Generator which converts synthetic to real
D_m = first discriminator feature map
λ = tuning parameter (10 used in experiments)

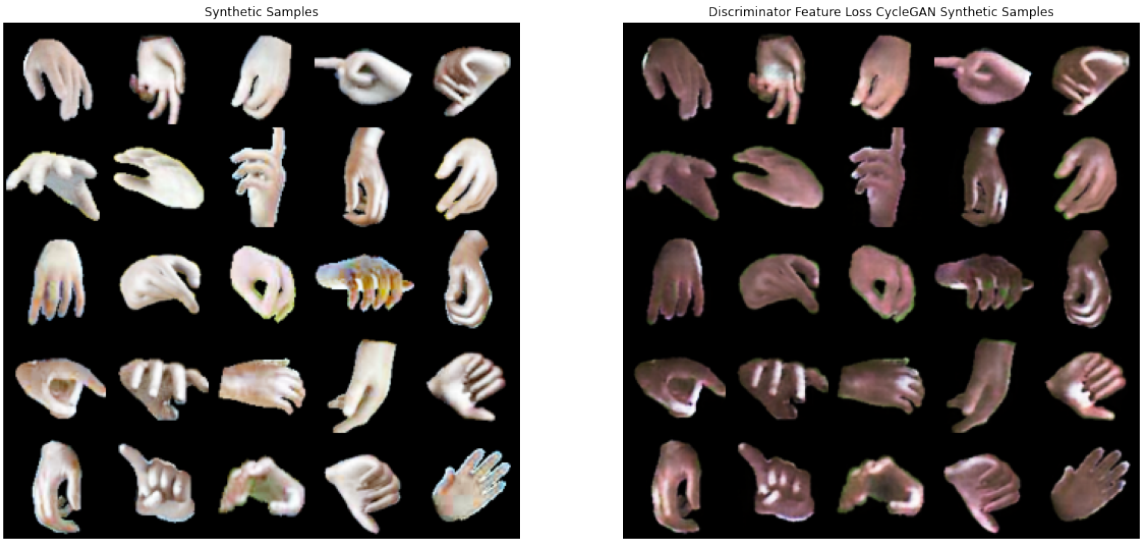

Figure 3.9: Synthetic inputs and outputs with the Discriminator Feature Loss CycleGAN model.

3.4 Hand Pose Estimator

Considering the application of the refined data is to provide training data for a hand pose estimation network, the most useful evaluation metric would be in the context of a hand pose estimator's performance when trained with the given data. It was initially attempted to modify the Discriminator with a secondary 21 3D keypoint output to additionally optimise for. This was to kill two birds with one stone - research has suggested using labels is beneficial for adversarial training stability [15], and the discriminator could then have a secondary use when the system is evaluated using hand pose predictions. Ultimately, the addition of the secondary output provided no advantage, as the architecture was not initially designed with hand pose estimation in mind (all keypoint predictions approached the natural prior present within the dataset). A dedicated hand pose estimation system was required for the evaluation.

For this, a currently available hand pose estimator was requisitioned and reverse engineered for the dataset available. The *hand3d* network trained in the paper developed by Zimmerman et al. [32, 4] was selected for this, as it made use of the same datasets in its experiments.

The hand3d network is a pipeline consisting of three stages with a high level architecture shown in Figure 3.10:

1. A hand segmentation phase which segments the hand and crops the image.
2. A keypoint detection phase which produces 64*64 likelihood scoremaps for 21 keypoints based on a cropped segmented hand.
3. A lifting phase which takes scoremaps as input and outputs the predicted 3D position of each keypoint.

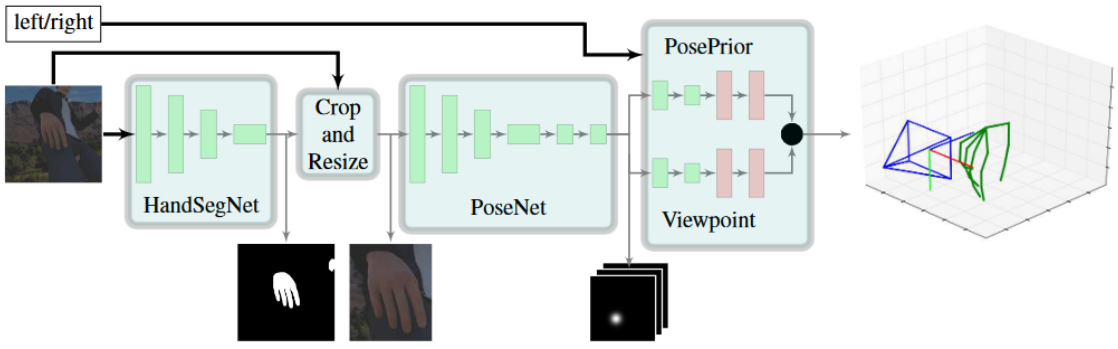

Figure 3.10: High level overview of the hand3d network [32, Figure source]. The section trained and evaluated in isolation is PoseNet.

As the images were preprocessed to be cropped and masked, it would not be useful to evaluate stage 1 performance of the pipeline. Instead, stage 2 was chosen as an evaluation. This stage was trained from scratch using each of the versions of the synthetic dataset as training sets (which were refined against a training set of real images), and then evaluated using a test set of real images. As the details of hand pose estimation are outside the scope of this project, specifics on architecture and training process are omitted from this report.

Chapter 4

Evaluation

Evaluation was carried out quantitatively and qualitatively. Quantitative evaluation covered the hand 2D keypoint detection results, and empirically studied metrics of evaluating GANs [27]. Qualitative evaluation covered visual inspection of images and of UMAP [14] embeddings of the various datasets.

4.1 Keypoint Detection in 2D

PoseNet takes a cropped, masked image of a hand as input and outputs 21 location scoremaps for each of the 21 keypoints of the hand. The network was trained from scratch using different versions of the synthetic dataset, and evaluated using a test set from the FreiHAND [2] dataset. Across the training datasets, all 41258 samples had consistent annotations, such that the only variation was the content of the images themselves. The network was fed ground truth crops and segmentations.

The measures reported in Table 4.1 are the endpoint error (EPE) in pixels and the area under the curve (AUC) on the percentage of correct keypoints (PCK) for varying error thresholds (0 to 30 pixels). Due to time constraints, the networks could not be trained more than once, which does bring into question the statistical significance of results.

Table 4.1: PoseNet results with FreiHAND (real) test set, when trained with different modifications of the Rendered Hand (synthetic) dataset. Synthetic 'w/' and 'wo/' Norm refers to the synthetic dataset *with* or *without* colour normalisation.

	Synthetic wo/ Norm	Synthetic w/ Norm	VAEGAN	CycleGAN Authentic	CycleGAN D Feat. Map 1
EPE median	16.081	15.664	15.941	15.888	15.802
EPE mean	16.799	16.186	16.933	16.608	16.365
AUC	0.465	0.483	0.462	0.472	0.478

From the results of Table 4.1, it can be observed that the highest performing dataset on all measures are that of the synthetic dataset with colour normalisation. It is hypothesised this is due to the de-

colourised hands being more consistent with the Caucasian skin of the real dataset. Additionally, the decolourisation step does not remove important information such as shading or high resolution details, which can be observed in the samples belonging to the adversarially modified data. Of the adversarially modified data, VAEGAN had the lowest performance, bettered by Authentic CycleGAN, and topped out by CycleGAN with Discriminator Feature Map reconstruction loss. Both CycleGAN versions outperform the raw synthetic data with no decolourisation.

This would appear to agree with a qualitative visual inspection of the samples produced by the models. The VAEGAN outputs (Figure 3.5) lose a lot of high frequency details which would be important in discerning keypoint locations. The CycleGAN architectures do not downscale inputs to the degree of the VAEGAN architecture, which aids in preserving important details. While the authentic CycleGAN (Figure 3.8) produces outputs of a higher resolution, much of the features which contribute to a realistic image, such as shading, are lost, as the cycle consistency loss penalises pixel-wise differences as opposed to differences with respect to the discriminators. CycleGAN with the 1st Discriminator Feature map reconstruction (Figure 3.9) however preserves realistic features such as shading, but loses high frequency details due to the 1st feature map being 32*32 image as opposed to 64*64 pixels.

4.2 GAN Evaluation Metrics

GAN evaluation is notoriously difficult, with many papers resorting to solely qualitative inspection, or setting up trials with human participants, which tend to be biased towards image quality and less towards overall distribution characteristics [27]. In order to quantitatively evaluate the models fairly, a human removed, scientifically backed set of metrics are required.

The datasets were evaluated using several methods outlined by Xu et al. in their empirical study of GAN evaluation techniques [27]. In their findings, they demonstrated that ImageNet-trained ResNet-34's [3] convolutional feature space was a better choice than its pixel feature space, as it is invariant to minor rotational and translational transformations. This evaluation followed the same setup as Xu et al., with the code being sourced from their paper. All experiments were repeated 5 times with shuffled permutations of 12800 drawn samples, and the error bars in plots represent the standard deviation of the trials. FreiHand was included in every experiment as a control.

4.2.1 1-Nearest Neighbour Score

The 1NN can be used to measure the leave-one-out (LOO) accuracy of a 1NN classifier trained on two distributions. When both distributions match and are of a large size, the classifier should achieve an accuracy of $\sim 50\%$ [27]. The experiment was carried out in the first and fourth feature space of ResNet-34, and the results are shown in Figures 4.1 and 4.2.

It is clear from Figure 4.1 that in the shallow layers all of the synthetic datasets are easily identified by the classifier, with the classifier having an accuracy of $93\%+$ for all synthetic datasets. In deeper feature space, however, Figure 4.2 demonstrates that the synthetic datasets are less separable from the real hands. In this space, the modified CycleGAN dataset brings the 1NN classifier accuracy down to 68.8%, 11% lower than the second best performing dataset.

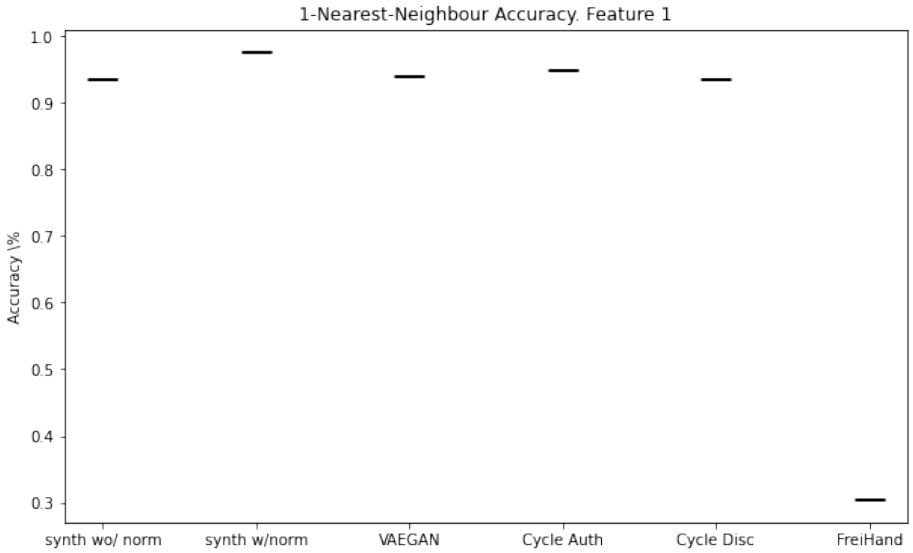

Figure 4.1: 1NN accuracy scores for the experimental datasets on the first feature in ResNet convolutional space.

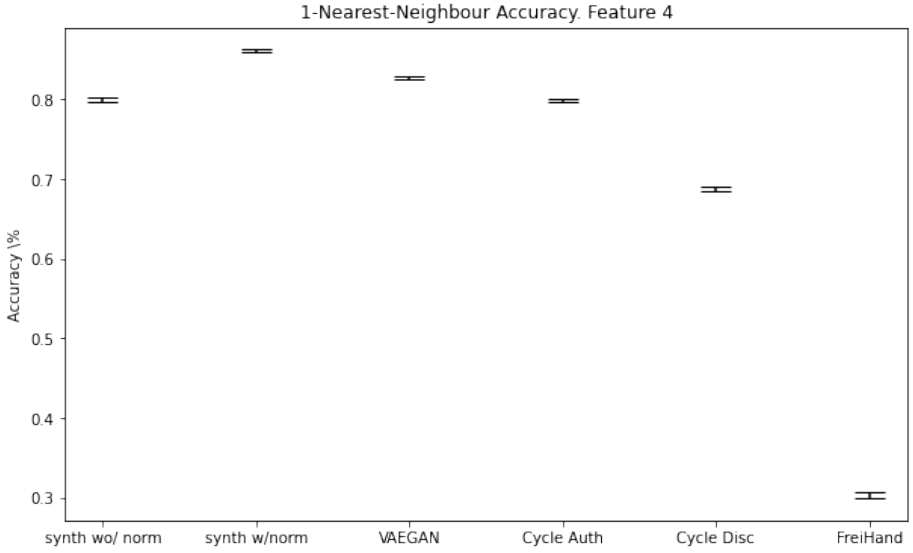

Figure 4.2: 1NN accuracy scores for the experimental datasets on the fourth feature in ResNet convolutional space.

A one-way ANOVA test with a significance value of 0.05 was carried out for the first and fourth feature 1NN experiments with the synthetic distributions (not FreiHand). The respective f-ratios were 1195 and 3347, both resulting in a p-value $<.00001$, which is significant.

4.2.2 Inception Score

The inception score is a widely adopted metric which uses an independent classification model M to provide a score representing the quality and diversity of a dataset [27]. It assesses distributions independently from a target distribution (i.e. the score is not reflective of a relationship between p_{real} and p_{fake}). Inception scores for ResNet-34's fourth feature space are shown in Figure 4.3. The score is computed as follows:

$$IS(p) = e^{E_{x \sim p_{data}}[KL(p_M(y|x)||p_M(y))]} \tag{4.1}$$

where:

$p_M(y|x)$ = label distribution of x as predicted by M
$p_M(y)$ = marginal of $p_M(y|x)$ over distribution p_{data}

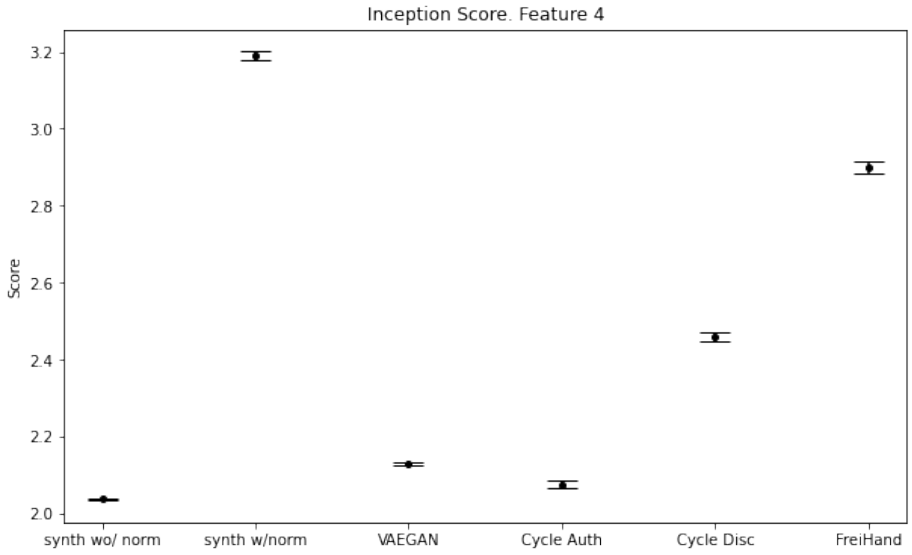

Figure 4.3: Inception scores for the experimental datasets.

As can be observed from Figure 4.3, the decolourised raw synthetic dataset has the highest score, which suggests it is a high quality image generation model. The reasoning behind the high score may be that the sharp features of the hand are distinct and without noise. The dark lighting along with darker skin tones against a black background may therefore be the reasoning behind the original colourised dataset having the lowest score. From observing the adversarial model scores with respect to the FreiHand dataset, it is clear that the modified CycleGAN vastly outperforms the VAEGAN and Authentic CycleGAN models. This agrees with intuition, as the lighting in the images produced by the modified CycleGAN is harsh, creating sharp and distinctive features. VAEGAN outputs lack high frequency details, and Authentic CycleGAN produces images with flat lighting and little colour variation.

A one-way ANOVA test with a significance value of 0.05 was carried out for the inception experiment with the synthetic distributions (not FreiHand). The f-ratio was computed as 13150, resulting in a p-value $<$.00001, which is significant.

4.2.3 Frechét Inception Distance

The Frechét inception distance (FID) was introduced in 2017 [7] as a metric of GAN evaluation. Unlike the inception score, the FID considers the synthetic distributions with respect to the real distribution. It treats both distributions p_{real}, p_{fake} in a given feature space as Gaussian random variables with empirical means and covariances, and computes the distance between them. It was deemed by Xu et al. to perform well in terms of discriminality and robustness [27], and is computed as shown in Equation 4.2. FID scores for the experimental datasets are shown in Figure 4.4.

With regards to Figure 4.4, while all the adversarial model distributions have lower distances than the colourised and decolourised raw synthetic datasets, the modified CycleGAN sees significant reduction in distance compared with the other models.

$$FID(p_r, p_f) = \|\mu_r - \mu_f\| + tr(\Sigma_r + \Sigma_f - 2(\Sigma_r\Sigma_f)^{1/2}) \tag{4.2}$$

where:

r denotes real distribution

f denotes fake distribution

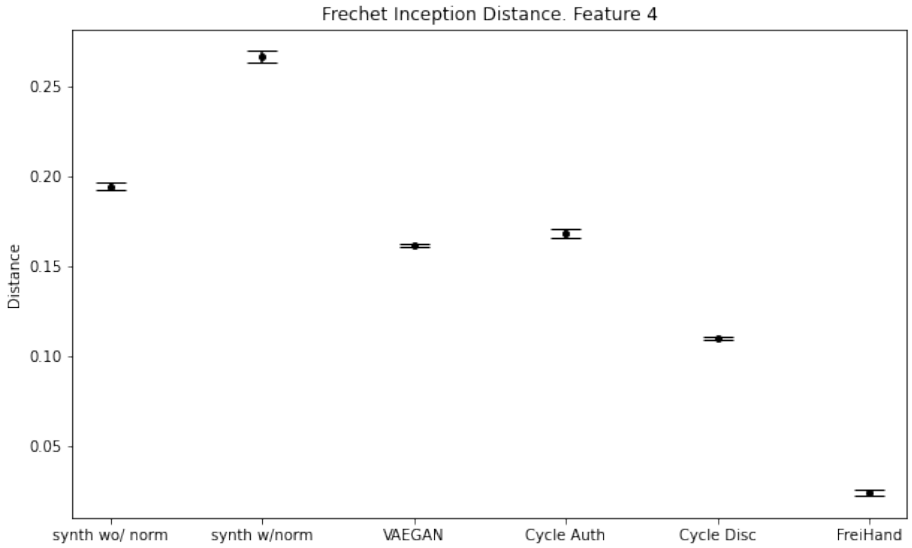

Figure 4.4: Frechét Inception Distance for each distribution with respect to the FreiHand distribution.

A one-way ANOVA test with a significance value of 0.05 was carried out for the FID experiment with the synthetic distributions (not FreiHand). The f-ratio was computed as 3054, resulting in a p-value $<.00001$, which is significant.

4.3 UMAP

Uniform Manifold Approximation and Projection [14] is a dimensionality reduction technique used for visualisation of high dimensional data for interpretation. It was tested with 1024 sample subsets of the experimental datasets with two number of neighbour choices: 5 and 19. This was to assess the local and global structure of the manifold directly. The UMAP outputs are shown in Figures 4.5 and 4.6.

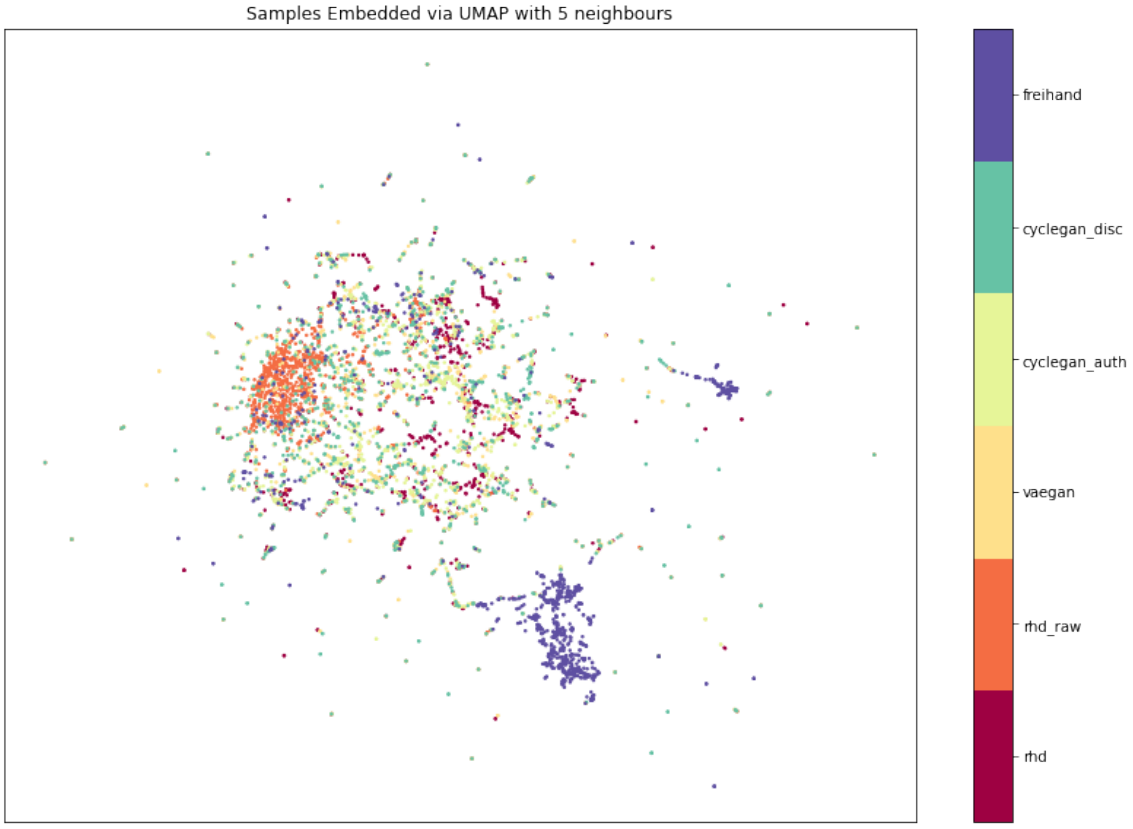

Figure 4.5: UMAP embedding of 1024 samples from each dataset. 'rhd' refers to Rendered Hand Dataset prior to being passed through a network ('raw' denotes without colour normalisation).

Observing Figure 4.5, it appears the Rendered Hand Dataset (synthetic) has well defined locality at a low level, as does FreiHand (real). The decolourised synthetic and the adversarially augmented datasets however do not appear to have well defined low level structure.

From observing Figure 4.6, it is clear that each of the datasets are quite separable, with the decolourised synthetic data, and adversarially modified data clusters being relatively close compared with the coloured synthetic data and freihand data. This makes sense, as the decolourised synthetic data acted as input for each of the developed models.

The implication of these visualisations is that the models do not refine synthetic images such that they appear to come from a distribution of real images.

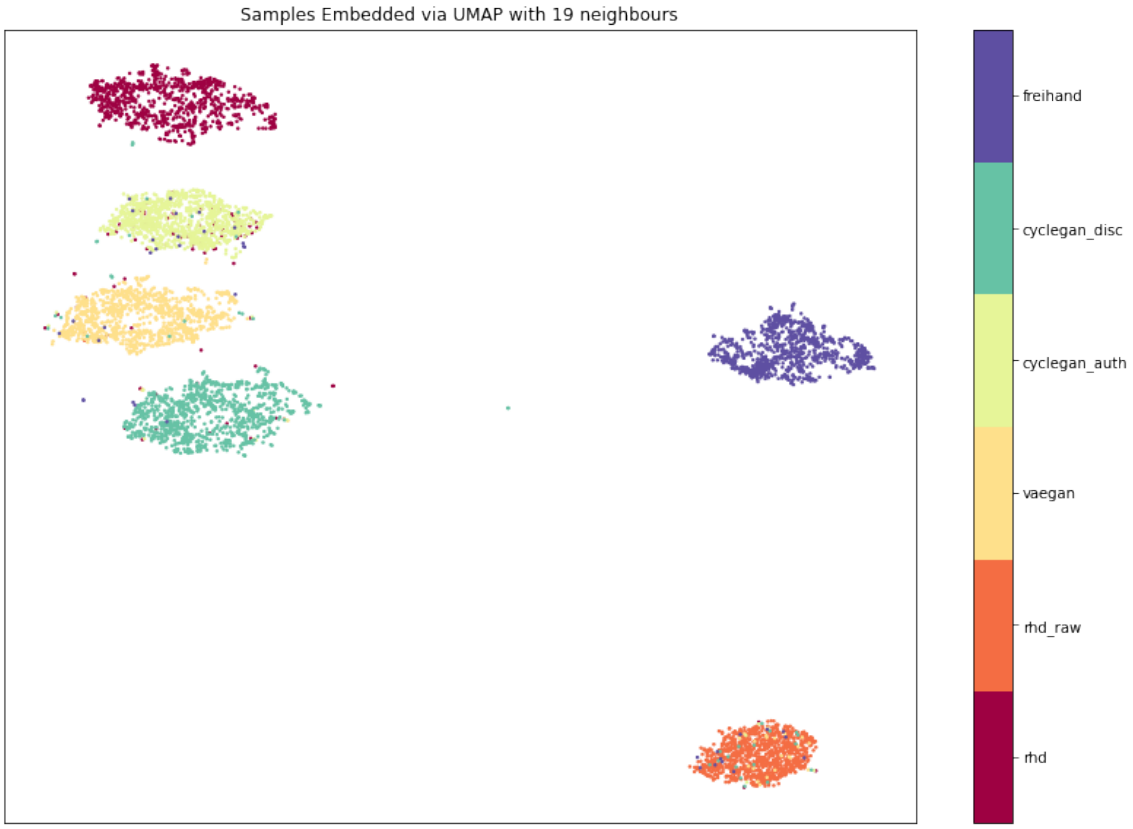

Figure 4.6: UMAP embedding of 1024 samples from each dataset. 'rhd' refers to Rendered Hand Dataset prior to being passed through a network ('raw' denotes without colour normalisation).

Chapter 5

Conclusion

This project defined the research, development, and evaluation of solutions which address the problems that arise when using data from one domain to solve a machine learning problem in a different domain. The combination of autoencoders (AEs) and generative adversarial networks (GANs) were investigated as a solution to the problem of hand pose estimation with simulated hands. Three architectures were developed and evaluated; a variational AE / GAN hybrid (VAEGAN), an authentic CycleGAN architecture and a modified CycleGAN architecture with a discriminator feature reconstruction loss.

A qualitative evaluation stage assessed the visual quality of model samples, as well as assessed a UMAP embedding of the datasets. Visually it was found that VAEGAN outputs held the overall geometric outline of inputs, but neglected to maintain high frequency details such as texture or shading. CycleGAN outputs preserved more discernible sharp features of the hands, such as fingers. The modified CycleGAN function had a more diverse dataset in terms of shading, but a noisier signal due to the reconstruction loss taking place in a 32*32 element feature space. UMAP was able to separate the real dataset and the raw simulated dataset with low number of neighbours, and quite easily separated the vast majority of samples into their correct respective dataset clusters as the number of neighbours increased, implying that all datasets were distinct from each other.

A quantitative evaluation stage assessed hand pose prediction performance when training with different versions of the simulated dataset with a test set of real hands. It found that decolourising the simulated dataset (without any further model augmentations), provided the best performance with the real dataset (which was made up of exclusively white hands). VAEGAN had the lowest performance of all tested datasets, with both CycleGANs outperforming the colourised simulated dataset. The modified CycleGAN had the best performance of the models in all metrics, perhaps due to the realistic reconstructions. It was hypothesised that the decolourised simulated data had the best performance due to its preservation of the important original information. Metrics were then used to assess the datasets' quality, diversity and comparison to the real distribution. Of the model distributions, it was found that when in a deeper convolutional feature space, the modified CycleGAN was the most comparable to the target distribution with regards to 1NN classification performance and the Frechét inception distance. Inception score quantified the modified CycleGAN as being the best generational model of those developed.

Overall, the models produced did not beat out the baseline, the decolourised synthetic dataset, in terms of hand pose prediction performance, however the modified CycleGAN did come reasonably

close. When evaluated exclusively as generational models, the modified CycleGAN was deemed to produce outputs significantly closer to the real data than other distributions evaluated. The contribution of this thesis is the proposal of CycleGAN with a discriminator feature loss as a method of converting the style of images more realistically by utilising a cycle consistency that is with respect to the discriminators. The project provided insights into the strengths and weaknesses of several model architectures, and laid the groundwork for further research to take place to improve results. Several ideas for this are proposed in the following section.

5.1 Future Work

5.1.1 Hyperparameter Optimization with respect to Hand Pose Estimation

An area which was omitted from the project was the optimisation of hyperparameters with respect to some value. Tuning hyperparameters of the adversarial autoencoder networks according to hand pose estimation performance may be a key step in generating realistic images which maintain important content of the input images. The data pipeline would have to be altered to accommodate this, as well as considerable computational and time allowances.

5.1.2 Wasserstein GAN

The Wasserstein GAN (WGAN) [1] is an architecture who's loss function is the earth moving distance between two distributions. The benefit of such as architecture is that it provides an end state for training, as the loss function correlates with image quality. Additionally, the WGAN loss function has been found to stabilise training.

5.1.3 Better Training Data

A key problem throughout the project was the difference between the simulated and real datasets. Although the preprocessing phase addressed the issue of the simulated hands being a variety of skin tones since the real data is only one tone, the decolourisation modification to accommodate this could be considered a 'hack' that only made the simulated dataset less realistic and reflective of the diversity of skin tones in reality. Simulated data which was of the same skin tone as the real hands to begin with would likely produce better augmentations. The problem posed in the project is inverse to the types of problems GANs aim to solve in the real world. The wide diversity of skin tones in the real world should be reflected in the GAN's real dataset. The problem adversarial networks may aim to solve in this case is generalising simulated data (which may be of only one skin colour), into a distribution with a variety of skin colours.

5.1.4 Higher Resolution Generations

While the discriminator feature loss CycleGAN had the highest performance of all the models with regards to hand pose estimation, it was limited by a reconstruction loss in a lower x and y image

dimension, which produced noisier outputs. This could potentially be addressed with an initial residual block as the first layer of the discriminator. In this case, reconstruction in convolutional feature space could be achieved while also using a feature map of the same resolution as the input. Experimentation with higher resolution images could also be evaluated as this would maintain more information which PoseNet could use.

5.1.5 Thorough Evaluation

A limitation of the evaluation stage was the relatively low number of samples in the datasets compared with the likes of ImageNet [3], which it is hypothesised lead to the lack of convergence of some of the metrics used to evaluate the distributions. This could be improved with datasets of a larger size.

Appendix A

First appendix

A.1 GitHub Repository

The codebase for the project is hosted at the following GitHub repository:
`https://github.com/LMurphy99/GAN_Hands`

Bibliography

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[2] Jimei Yang Bryan Russell Max Argus Christian Zimmermann, Duygu Ceylan and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[4] Zimmerman et al. Hand3d github. `https://github.com/lmb-freiburg/hand3d`.

[5] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[7] 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. *CoRR*, abs/1706.08500, 2017.

[8] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[9] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7122–7131, 2018.

[10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[11] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric, 2016.

[12] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016.

[13] Alex Lenail. Nn-svg. `http://alexlenail.me/NN-SVG/AlexNet.html`.

[14] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[15] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[16] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. Ganerated hands for real-time 3d hand tracking from monocular RGB. *CoRR*, abs/1712.01057, 2017.

[17] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Training a feedback loop for hand pose estimation. *CoRR*, abs/1609.09698, 2016.

[18] Seong-Jin Park, Hyeongseok Son, Sunghyun Cho, Ki-Sang Hong, and Seungyong Lee. Srfeat: Single image super-resolution with feature discrimination. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[19] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[20] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):245:1–245:17, November 2017.

[21] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. *CoRR*, abs/1612.07828, 2016.

[22] Thalles Santos Silva. A short introduction to generative adversarial networks. *https://sthalles.github.io*, 2017. Figure Source.

[23] James S. Supancic, Grégory Rogez, Yi Yang, Jamie Shotton, and Deva Ramanan. Depth-based hand pose estimation: Data, methods, and challenges. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1868–1876, 2015.

[24] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (ToG)*, 33(5):1–10, 2014.

[25] Ramon Viñas, Helena Andrés-Terré, Pietro Liò, and Kevin Bryson. Adversarial generation of gene expression data. *Bioinformatics*, 01 2021. btab035.

[26] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. ESRGAN: enhanced super-resolution generative adversarial networks. *CoRR*, abs/1809.00219, 2018.

[27] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks, 2018.

[28] Duncan Zauss, Sven Kreiss, and Alexandre Alahi. Keypoint communities, 2021.

[29] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. Mediapipe hands: On-device real-time hand tracking. *CoRR*, abs/2006.10214, 2020.

[30] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.

[31] Christian Zimmermann, Max Argus, and Thomas Brox. Contrastive representation learning for hand shape estimation. *CoRR*, abs/2106.04324, 2021.

[32] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images. Technical report, arXiv:1705.01389, 2017. https://arxiv.org/abs/1705.01389.