DEGREE PROJECT IN TECHNOLOGY,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2020

Unsupervised 3D Human Pose Estimation

KTH Thesis Report
Final Draft

Sri Datta Budaraju

KTH ROYAL INSTITUTE OF TECHNOLOGY
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Authors

Sri Datta Budaraju <budaraju@kth.se>
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology

Place for Project

Stockholm, Sweden
Stuttgart, Germany

Examiner

Danica Kragic Jensfelt
Stockholm, Sweden
KTH Royal Institute of Technology

Supervisor

Hedvig Kjellström
Stockholm, Sweden
KTH Royal Institute of Technology

Supervisor - Host

Arij Bouazizi
Stuttgart, Germany
Mercedes-Benz AG, Research and Development

Abstract

The thesis proposes an unsupervised representation learning method to predict 3D human pose from a 2D skeleton via a VAE-GAN hybrid network. The method learns to lift poses from 2D to 3D using self-supervision and adversarial learning techniques. The method does not use images, heatmaps, 3D pose annotations, paired/unpaired 2D-to-3D skeletons, 3D priors, synthetic 2D skeletons, multi-view or temporal information in any shape or form. The 2D skeleton input is taken by a VAE that encodes it in a latent space and then decodes that latent representation to a 3D pose. The 3D pose is then reprojected to 2D for a constrained, self-supervised optimization using the input 2D pose. Parallelly, the 3D pose is also randomly rotated and reprojected to 2D to generate a 'novel' 2D view for unconstrained adversarial optimization using a discriminator network. The combination of the optimizations of the original and the novel 2D views of the predicted 3D pose results in a 'realistic' 3D pose generation. The thesis shows that the encoding and decoding process of the VAE addresses the major challenge of erroneous and incomplete skeletons from 2D detection networks as inputs and that the variance of the VAE can be altered to get various plausible 3D poses for a given 2D input. Additionally, the latent representation could be used for cross-modal training and many downstream applications. The results on Human3.6M datasets outperform previous unsupervised approaches with less model complexity while addressing more hurdles in scaling the task to the real world.

Keywords

Computer Vision, Projective Geometry, Deep Learning, Unsupervised Learning, 3D Human Pose Estimation, GAN, AutoEncoder, Hybrid Generative Model, Self-Supervision

Abstract

YET TO BE TRANSLATED

Svenskt abstract Svensk version av abstract – samma titel på svenska som på engelska.

Skriv samma abstract på svenska. Introducera ämnet för projektet och beskriv problemen som löses i materialet. Presentera

Nyckelord

Kandidat examensarbete, ...

Acknowledgements

I would like to express my sincere gratitude to my supervisor at KTH, Hedvig Kjellström, and my industrial supervisor, Arij Bouazizi for their patient guidance, encouragement, and constructive critique of this research work. I am thankful to my examiner, Danica Kragic Jensfelt for her feedback and evaluation of this thesis. I would like to thank Mercedes Benz for hosting my thesis and providing me with the required infrastructure. I would like to express my appreciation to Ulrich Kressel and Julian Wiederer for welcoming me to their research group. I would like to acknowledge the helpful feedback, support, and company from my thesis group at KTH and would like to thank Ruibo Tu, Jade Cock, and Oscar Örnberg for their immense help. I am thankful to my friends Nik Vaessen and Vivek Chalumuri for their helpful discussions, comments, and advice throughout the whole process. Finally, I am grateful to my family for their lifelong support. This thesis was carried out during a period of unprecedented uncertainty and was only possible as a result of the incredible patience and empathy shown by everyone involved in the process.

Acronyms

ANN Artificial Neural Network

AR/VR Augmented Reality/Virtual Reality

BCE Binary Cross Entropy

BH Best Hypothesis

β-VAE Beta Variational Auto-Encoder

EM distance Earth Mover's Distance

FC Fully Connected

FOV Field of View

GAN Generative Adversarial Network

GMM Gaussian Mixture Model

HPE Human Pose Estimation

JS-divergence Jensen–Shannon Divergence

KLD Kullback–Leibler Divergence

L1 Least Absolute Deviations

MDN Mixture Density Network

MMVAE Mixture-of-Experts Multimodal Variational Auto-Enocder

MoCap Motion Capture

MPJPE Mean Per Joint Position Estimate

MSE Mean Squared Error

MVAE Multimodal Variational Auto-Enocder

NRSfM Non-Rigid Structure from Motion

POV Point of View

ReLU Rectified Linear Unit

SOTA State-of-The-Art

UMAP Uniform Manifold Approximation and Projection

VAE Variational Auto-Encoder

WGAN Wasserstein Generative Adversarial Network

WGAN-GP WGAN with Gradient Penalty

ZV Zero Variance

Contents

Chapter 1

Introduction

With rapid advancements in deep learning facilitated by the developments in computational hardware, there has been tremendous growth in computer vision research and its applications [1]. One of the major tasks of computer vision that is required for real-world applications is to perceive and understand dynamic objects and more importantly humans.

Human Pose Estimation, in the following, referred to as HPE, is a fundamental computer vision task that also forms a basis for advanced tasks such as human action and gesture recognition as well as human motion prediction. HPE is defined as the localization of human joints (also known as keypoints, including head, eyes, ears, nose, etc) mainly in images and videos in either a 2D or 3D coordinate space. The widely available and used data like images and videos are 2-dimensional data and lack spatial information which is crucial for most of the applications like autonomous driving, Augmented Reality/Virtual Reality (AR/VR), social robots, etc. Hence the focus of this thesis is on 3D HPE.

1.1 Background

There has been much research done in 3D HPE and more advancements have been made in the past few years leveraging the power of deep learning. The current research explores various ways to solve the task using RGB/Depth image channels, 2D poses, 3D poses, multi-view, and sequential images/videos.

[1] AI and Compute `https://openai.com/blog/ai-and-compute/`

Most of them are supervised learning approaches that require 3D ground truth poses that can only be acquired using physical sensors. Supervised learning methods that learn 3D pose from images, follow a complex cascading approach with 2D poses in some form is an intermediate output. And other learning approaches mostly make use of images from multiple views to estimate the 3D pose.

Assuming that 2D poses are obtained from the State-of-The-Art (SOTA) models specialized on 2D HPE models, some works focus on estimating the 3D pose from these 2D poses instead of images. Such networks are called *lifting* networks. These lifting networks can be simple without requiring computationally expensive convolutional layers as it only needs to learn the features of 2D poses that are low dimensional compared to images. However since 2D poses are naturally obtained by projecting 3D poses to a plane, it is an inverse problem. Also, there are multiple feasible 3D poses that when the projected result in the same 2D pose. Thus making the task of lifting 2D-to-3D, a *ill-posed inverse* problem due to its inherent ambiguity.

Non-supervised (Weakly/Self/Unsupervised) learning regimes, that are less dependent on 3D pose ground truth, have also gained traction in recent years. Weakly supervised approaches use 3D ground truth indirectly by generating more 2D poses from more views or, for training a discriminator network of a Generative Adversarial Network (GAN). While unsupervised learning (self-supervised) approaches do not use 3D ground truth poses in any shape or form. Many of the deep learning techniques that have already improved the results in other computer vision tasks are yet to be explored in 3D HPE.

1.2 Problem

More specifically, the research questions addressed in this thesis are:

1. How can we learn a strong visual representation of the data to tackle the task of 3D HPE?
2. Could data as its own supervisory goal (self-supervision) resolve the ambiguities in 2D-to-3D pose lifting?

1.3 Goal

The main aim of the thesis is to investigate unsupervised learning approaches and 2D-to-3D lifting methods that could help tackle the challenges in scaling 3D HPE to the real-world.

Improvements in the aspects of ease of training procedure i.e requiring less data or less labor-intense labeling, inference speed, and most importantly accuracy is important and will directly impact its super tasks such as action and gesture recognition, motion prediction and intention, behavior prediction.

1.4 Societal, Ethical and Sustainability Aspects

The ability to percieve and understand human state, motion and behavior are vital for integraton of systems that interact with the wro HPE plays a very important role to enable autonomous vehicles and robots to safely interact with humans. It also plays a vital role in developing higher dimensional communication platforms with AR/VR. It is crucial for surveillance systems to ensure public safety. However such important technologies are only as good as the intentions of its users. Mass surveillance of citizens by their governments is a matter of debate.

1.5 Methodology

The problem of 3D HPE has 3 aspects to be addressed and explored.

The neural network: The architecture and the kind of neural network to be used. 3D poses can be predicted using a regular linear neural network or using various other architectures like autoencoders. These models can use linear, convolutional, and graph layers to learn features. This thesis focuses on investigating the merit in using architectures like Variational Auto-Encoders (VAEs) to solve the 3D HPE within the context of leveraging probabilistic inference models, as a deterministic approach for an inherently ill-posed problem is not ideal.

The learning task: The model could either learn to directly predict the 3D coordinates of the keypoints or learn structural parameters that could model a 3D pose. The thesis only explores the former task.

The learning technique (or the cost): The model can be either trained by directly comparing the predicted 3D pose and the ground truth 3D pose thus requiring 3D annotations, or by projecting the prediction back to 2D to compare with the input (requires only 2D annotations that could be acquired from SOTA in 2D HPE) and use a different technique to ensure the correctness of pose in 3D. Adversarial training and self-supervision techniques have also given promising results in the last couple of years. The method proposed in the thesis is designed to learn 3D from 2D poses alone in an unsupervised-adversarial learning fashion after the capabilities of the method under supervised settings being verified.

The prime motivation behind the design choices is to address the challenges in scaling up 3D HPE to real-world.

1.6 Stakeholders

Daimler's 'Environment Perception for Autonomous Driving' R&D team in Stuttgart, Germany, conducts cutting-edge research in the field of Computer vision and Deep Learning to improve the State-Of-The-Art and to make Autonomous Driving a reality. This thesis is part of the team's on-going research in understanding the human state, motion, and behavior, which would help autonomous cars better perceive, understand, and interact with humans. Daimler/Mercedes-Benz autonomous cars try to understand humans both, inside and outside the car and HPE is a critical element to accomplish this task.

The question is also of interest to the research area of Human State/Action Recognition in specific and also to areas of computer graphics to model humans in 3D space. Hence it is beneficial to various areas that try to understand and interact with humans. The scientific communities in the areas of Autonomous Driving, AR/VR, Motion Capture, Computer Graphics, and Human-Robot interaction could be interested in the contributions of this thesis.

1.7 Delimitations

This thesis focuses only on 3D pose estimation and not the intermediate 2D pose. Data collection is not part of the thesis study and uses only publicly available, widely used, and benchmarked datasets.

1.8 Outline

The theoretical knowledge required to understand the details of the thesis is presented next in chapter 2, Theoretical Background. This entails the explanation of some preliminary concepts 2.1, research area introduction 2.2, where the vast literature related to HPE is summarized, and the highlights of all the works 2.3 that are closely related to the thesis.

The background is followed by chapter 3, Data, providing details of the datasets used along with some visualizations. This chapter also explains the 3D projective geometry concepts required to understand the pre and post-processing steps the data undergoes.

The method, chapter 4, describes the proposed approach in detail. This covers the components of the proposed architecture, the motivation behind the choices, the training and evaluation procedure.

The experiments, chapter 5, entails details from other network choices to tricks explored to stabilize the GAN training. These details are important for the proper functioning and reproduction of the proposed method.

The results are presented and analyzed in chapter 6 and conclusions including discussion and future work are presented in chapter 7.

Chapter 2

Theoretical Background

This chapter provides the theory essential to understand the major components of the thesis. Prior knowledge of Artificial Neural Networks [1] and fundamental concepts of Deep Learning [9] is assumed.

2.1 Preliminary Concepts

2.1.1 Autoencoder

Autoencoders are a variant of Artificial Neural Networks (ANNs), which are designed to learn an identity function that generates the input data sample back. The network has a bottleneck(z), dividing the network into two parts, an encoder and a decoder as illustrated in Fig 2.1.1. The first network learns to compress the high dimensional input data to a low dimensional intermediate representation, *latent representation*, at the bottleneck. While the second network learns to reconstruct the data from the latent distribution. Thus learning to efficiently compress the data.

To put it in other words, in the process of learning to reconstruct the data, the encoder learns to filter the most important features of the given data distribution, so as it preserves the complete properties within the limits of the bottleneck. While the decoder learns comparatively general properties of the distribution which are used along with the compact latent representation from the encoder to fully recover the

[1] Artificial Neural Networks `https://en.wikipedia.org/wiki/Artificial_neural_network`

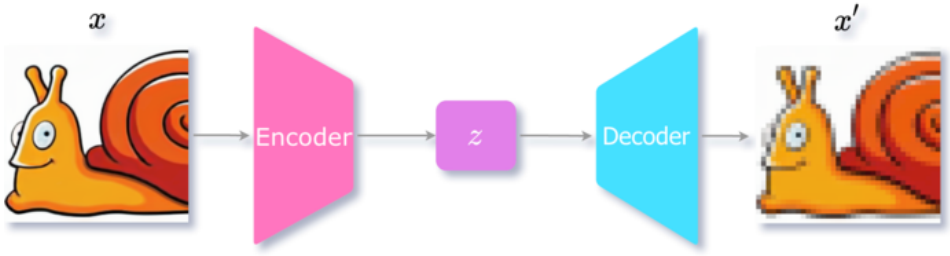


Figure 2.1.1: Illustration of autoencoder architecture. Image source [40]

data distribution. The network is trained to minimize the similarity between the reconstruction and the original data sample. This similarity can be determined by metrics such as Mean Squared Error (MSE), Least Absolute Deviations (L1) or Cross-Entropy loss.

The idea of an autoencoder dates back to the '80s proposed as a method for pre-training and feature learning [2, 32], learned dimensionality reduction [16]. In recent years, autoencoders are most popularly used as generative networks leveraging their ability to learn feature representations in an unsupervised way. Another interesting variant of autoencoders is the denoising autoencoder [41], where the input is a noised data and the decoder generates original data without noise. This variant is further evolved to accomplish the tasks of image denoising, watermark removal, inpainting, super-resolution, colorization, de-colorization, and compression [40, 49, 51].

2.1.2 Variational Autoencoders

VAEs, unlike standard autoencoders, learn to encode a data sample x as a probabilistic distribution rather than a deterministic value for the latent attribute [2]. The encoder q produces the probabilistic distribution by predicting two vectors that represent the mean μ and standard deviation σ of distribution for each of the latent attributes of x. And the decoder p takes a random sample z from this distribution to recover the sample as illustrated in Fig 2.1.2.

To put it formally, we have a hidden variable z which generates x. Since we only have x and would want to learn z i.e $p(z|x)$. But computing this posterior distribution is hard

[2] Variational Autoencoders `https://www.jeremyjordan.me/variational-autoencoders/`

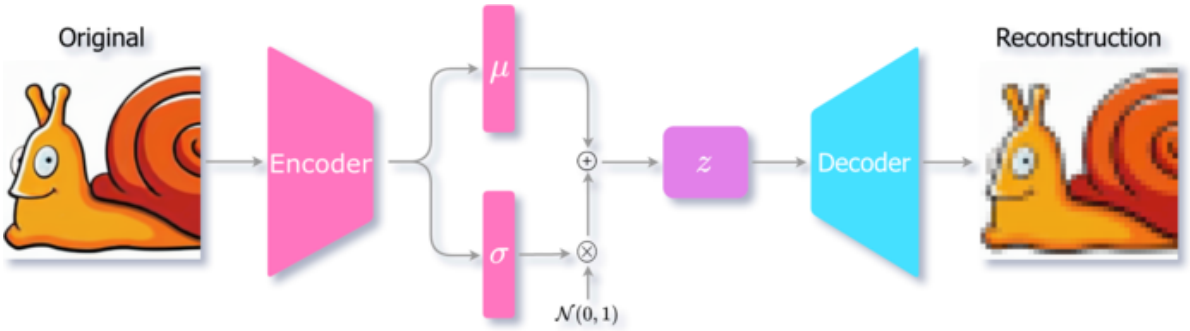


Figure 2.1.2: Illustration of Variational Autoencoder architecture.

as computing $p(x)$ Eqn. 2.1 is usually intractable [20].

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$
$$p(x) = \int p(x|z)p(z)dz \tag{2.1}$$

Hence, we try to approximate the posterior distribution by another distribution $q(z|x)$ (the encoder) using variational inference. Variational inference uses optimization to find a distribution that minimizes the Kullback–Leibler Divergence (KLD) to the posterior distribution, $\min D_{\mathrm{KL}}(q(z|x)||p(z|x))$ while trying to keep the learnt distribution close to the true prior distribution $p(z)$ [3]. The prior $p(z)$ is usually assumed to be a unit gaussian distribution. The above can also be achieved by maximizing:

$$\max \mathbb{E}_{q(z|x)} \log p(x|z) - D_{\mathrm{KL}}(q(z|x)||p(z)) \tag{2.2}$$

The first term in the above equation makes sure the reconstruction is close to the data sample x, while the second term tries to keep the learned distribution $q(z|x)$ close to the true prior $p(z)$. Hence the loss term to *minimize* while training the VAE is $\mathcal{L}_{\mathrm{VAE}}$ Eqn. 2.3.

$$\mathcal{L}_{\mathrm{VAE}} = -\mathbb{E}_{q(z|x)} \log p(x|z) + D_{\mathrm{KL}}(q(z|x)||p(z)) = \mathcal{L}_{\mathrm{recon}} + \mathcal{L}_{\mathrm{prior}} \tag{2.3}$$

However, the reconstruction error in the loss requires sampling z, which is a stochastic process and it is not possible to perform backpropagation. To address this problem, the ***reparametrization trick*** is used. Where ϵ is randomly sampled from a unit

gaussian distribution $\mathcal{N}(0,1)$ and is used to scale the standard deviation σ of the latent distribution represented by the encoder $q_\theta(z|x)$. Where θ is the parameters of the encoder. The sum of the mean μ and the scaled standard deviation $\sigma \odot \epsilon$ gives z, which is now differentiable while being stochastic as illustrated in Fig 2.1.3.

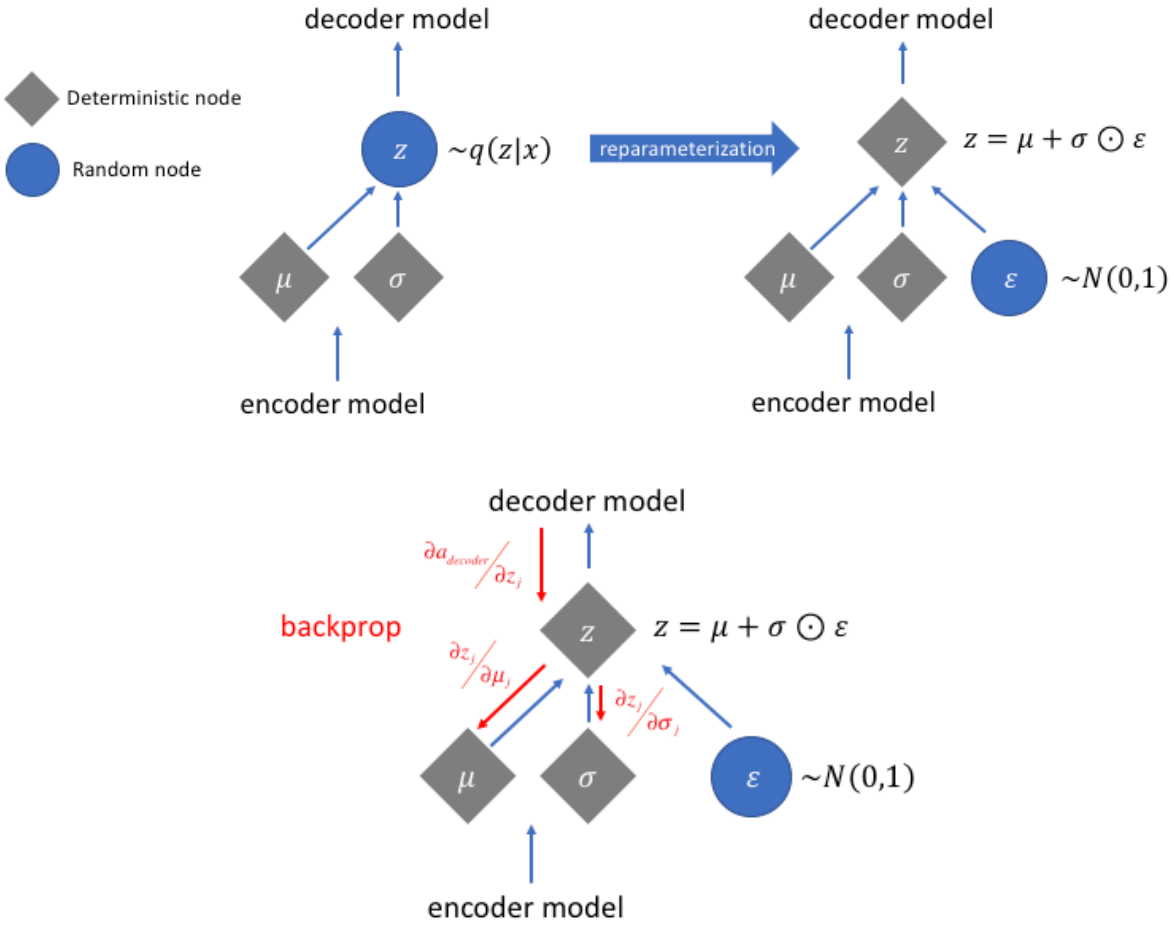


Figure 2.1.3: Comparing the data flow with and without reparametrization trick followed by the backprop calculation. Image source Note(2)

Hence the learned latent space of a VAE is continuous, while that of a standard autoencoder is discrete and clustered. As the decoder of the VAE is trained to generate data from this continuous space, it can generate realistic data by randomly sampling from this infinitely large latent space as illustrated in the Fig 2.1.4. This also enables smooth interpolation of data produced from one point in the latent space to another. In addition to that, we can also perform arithmetics in vector space, similar to the popular example from Natural Language Processing, $King - Man + Women = Queen$ but on much higher dimensional embedding space.

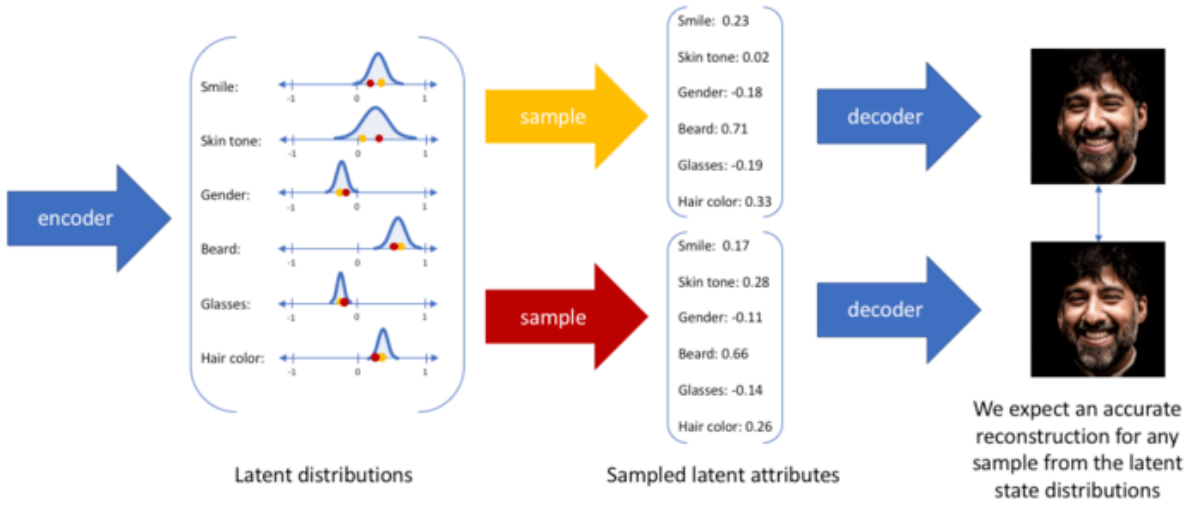


Figure 2.1.4: Probabilistic distribution of latent attributes. Image source Note(2)

2.1.3 Beta Variational Autoencoder

A VAE without the KLD term is effectively a standard autoencoder. As discussed the KLD term encourages the network to learn a distribution rather than a single value. If the variance of the distribution is not high, then it is again similar to an autoencoder. The more enforcement from KLD, the diverse the distribution. The VAE is forced to disentangle the representations, i.e the lesser is the correlation between each dimension in the latent space. Such disentangled representations are very useful for generative models. More importantly, it improves the interpretability of the latent space and can be leveraged to generalize to different downstream tasks. This emphasis on the latent space distributions can be achieved by disentangled variational autoencoders or Beta Variational Auto-Encoder (β-VAE), where β is the weight coefficient of the KLD term in the VAE loss function Eqn.2.3. So, the loss for the β-VAE is $\mathcal{L}_{\text{VAE}}$ Eqn. 2.4. The higher the beta the stronger the constrain on the disentanglement. However, this constraint will negatively affect the representation capability of the VAE.

$$\mathcal{L}_{\text{VAE}} = -\mathbb{E}_{q(z|x)} \log p(x|z) + \beta(D_{\text{KL}}(q(z|x)||p(z))) \tag{2.4}$$

2.1.4 Generative Adversarial Networks

GAN is an ANN that is used for generative tasks, to make the prediction *realistic*. A GAN is a combination of 2 networks namely, the generator G and the discriminator D. The generator learns to map a random sample or say, noise Z drawn from a latent distribution with density p_z to a higher dimensional data distribution with density p_g. Whereas the discriminator takes the output of the generator and tries to differentiate real data samples x from the fakes which do not belong to the real distribution p_r as illustrated in Fig 2.1.5.

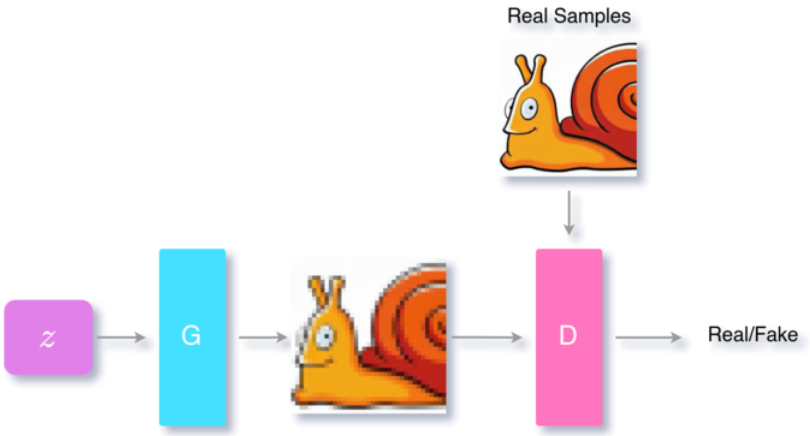


Figure 2.1.5: Illustration of GAN architecture. G is the generator network and D is the discriminator network.

The goal of the generator is to produce samples, $G(z)$, that fool the discriminator into believing them as real samples. While the goal of the discriminator is to distinguish between the samples produced by the generator and the real samples by predicting reals as 1 and 0 for fakes. This inverse goal of the two networks can be viewed as a game of tug of war or a 2-player minimax game. The result of the game is that the generator would ideally learn to produce realistic data samples by sampling from prior $p_g(z)$. We effectively train G to minimize $\log(1-D(G(z)))$ and D to maximize $\log(D(x))$ making the loss function as $\mathcal{L}_{\text{GAN}}$ Eqn. 2.5.

$$L_{\text{GAN}} = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.5}$$

However, when it comes to training a GANs, practice is very different from theory. The training of the discriminator and generator is done iteratively and sequentially. But training the discriminator network to the optimal solution and then training the

generator and repeating this loop is computationally challenging and would lead to overfitting the models on the finite dataset. To avoid this, the discriminator is trained for k mini-batch iterations before training the generator for one iteration. This is to keep the discriminator close to optimality while slowly training the generator [11]. The problem that arises here is that the generated samples are drastically different from the real samples as the generator has not yet learned to produce good samples. As the generator outputs samples close to noise, the discriminator easily distinguishes these samples from the real samples with high confidence. This saturates the loss term $\log(1 - D(G(z)))$ very quick and leads to the problem of *Vanishing Gradients*. Hence in practice, we train the generator G to maximize $\log D(G(z))$ instead of minimizing $\log(1 - D(G(z)))$, preventing the gradients from vanishing.

$$\begin{aligned} D^*_G(x) =& \frac{p_r(x)}{p_r(x) + p_g(x)} \\ =& \frac{1}{2} \end{aligned} \tag{2.6}$$

At global optimality $p_g = p_r$, and for a given generator G, the discriminator at optimality is $D^*_G(x)$ Eqn. 2.6. Hence the virtual cost $C(G)$ [11] when $p_g = p_r$ is:

$$\begin{aligned} C(G) &= \max_D V(G, D) \\ &= \mathbb{E}_{x \sim p_r} \left[\log D^*_G(x)\right] + \mathbb{E}_{z \sim p_z} \left[\log \left(1 - D^*_G(G(z))\right)\right] \\ &= \mathbb{E}_{x \sim p_r} \left[\log D^*_G(x)\right] + \mathbb{E}_{x \sim p_g} \left[\log \left(1 - D^*_G(x)\right)\right] \\ &= \mathbb{E}_{x \sim p_r} \left[\log \frac{p_r(x)}{P_r(x) + p_g(x)}\right] + \mathbb{E}_{x \sim p_g} \left[\log \frac{p_g(x)}{p_r(x) + p_g(x)}\right] \\ &= -\log(4) + D_{\text{KL}} \left(p_r \| \frac{p_r + p_g}{2}\right) + D_{\text{KL}} \left(p_g \| \frac{p_r + p_g}{2}\right) \\ &= -\log(4) + 2 \cdot \text{JSD}(p_r \| p_g) \end{aligned} \tag{2.7}$$

The Jensen–Shannon Divergence (JS-divergence) [3] between two distributions is always non-negative and will be equal to zero only when both the distributions are equal. As we derived in Eqn. 2.7 above, the best value of C i.e $-\log 4$ is possible only when $p_g = p_r$. It is hard to stabilize the GAN's minimax game [1]. It requires carefully tuned hyperparameters to maintain an equilibrium between the two players. Failing to find the proper balance between the networks leads to the problem of *Non-*

[3] Jensen–Shannon divergence `https://en.wikipedia.org/wiki/Jensen-Shannon_divergence`

Convergence, where the training oscillates and never converge. When the generator is not strong enough and learns to produce samples that fool the discriminator, it eventually would restrict itself to only learn to produce such samples. This problem is referred to as *Mode Collapse* [4]. There are many hacks as well as principled approaches that are formulated to handle these problems with considerable success [46].

2.1.5 Wasserstein GAN

Wasserstein Generative Adversarial Networks (WGANs) is a variant of GANs, where the second network is a critic that scores the samples on how real they look rather than a discriminator that only predicts binary labels of 1 and 0 for real or fake. WGAN use 1-Wasserstein distance [5] or Earth Mover's Distance (EM distance) instead of the JS-divergence used in the standard discriminator based GAN. Since the Wasserstein distance is non-evaluative, a modified version Eqn. 2.8 of it is proposed as the loss function in [13]. Where f being the critic network parameterized by w while clipping the weights to satisfy the Lipschitz constraint.

$$L = \mathbb{E}_{x \sim P_r} [f_w(x)] - \mathbb{E}_{x \sim P_g} [f_w(x)] \tag{2.8}$$

The fundamental goal of GANs is to minimize the distribution between the real and the generated distribution. This could be measured used either of KLD, JS-divergence, EM distance, or Wasserstein distance, the main difference being their impact on the convergence of these distributions. The interesting feature of the Wasserstein distance is that it is continuous and differentiable. Using this distance, the critic can train till optimality while having a reliable gradient throughout the training procedure. Hence the critic in WGAN does not have the saturation and vanishing gradient problems that exist in standard GANs. Due to continuous and clean gradients, the training is significantly stable and less sensitive to hyperparameters and model architecture. With WGAN the mode collapse problem is also significantly reduced. When it comes to practice, the most important problem that hinders training GANs is that there is no correlation between the quality of the generated data say, images, and the loss function. However, WGAN tries to converge the distributions while lowering the generation

[4] Common Problems in GANs `https://developers.google.com/machine-learning/gan/problems`

[5] Wasserstein Metric `https://en.wikipedia.org/wiki/Wasserstein_metric`

loss. And considerable relation between the loss and the quality of generations can be observed. WGAN with Gradient Penalty (WGAN-GP) [14] is an improved WGAN that uses gradient penalty to enforce Lipschitz constraint.

2.1.6 Hybrids - VAE-GAN

While the VAEs learn the latent space of the data very efficiently, the generative capabilities are limited in comparison to GANs. In the case of image generations, VAEs usually generate blurry images. While well trained GAN learn to generate photorealistic images. Though the task of the discriminator in GANs is to only learn what is real and what is fake, it implicitly learns rich a similarity metric in order do so [23]. The idea of a VAE-GAN, illustrated in fig. 2.1.6, is to exploit this ability of GANs as a learning metric for VAEs and the ability of VAEs to learn dense latent representation of the data.

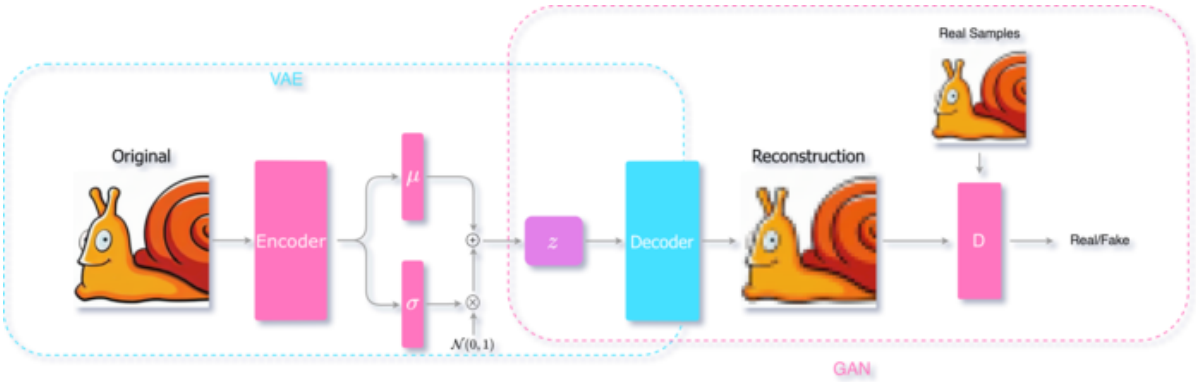


Figure 2.1.6: Illustration of the VAE-GAN architecture

Taking the example of images again, the element-wise error is a very poor similarity metric as a small deviation in a high-level feature, like eyebrow or head rotation would lead to high error as the pixel displacement propagates through huge parts of the image. These shifts, in reality, are plausible and realistic, probably indistinguishable from the human eye. Using the discriminator as a similar metric would address this problem as the error would be low for realistic deviations of features compared to unrealistic shifts say, the noise being upside down. This can be achieved by replacing the element-wise loss of the VAE Eqn. 2.3 with the hidden representation $D_l(x)$ of an intermediate layer l in the discriminator that would correspond to the hidden similarity metric. The

Gaussian distribution for $D_l(x)$ is :

$$p\left(D_l(x)|z\right) = \mathcal{N}\left(D_l(x)|D_l(\tilde{x}), I\right) \tag{2.9}$$

Where, $\tilde{x}$ is the generated sample from the VAE's decoder p. $D_l(tildex)$ is the mean of the Gaussian distribution and I is the identity covariance. Replacing this as the similarity metric in 2.3, we get the new $\mathcal{L}_{\text{recon}}$:

$$\mathcal{L}_{\text{recon}}^{D_l} = -\mathbb{E}_{q(z|x)} \log p\left(D_l(x)|z\right) \tag{2.10}$$

$\mathcal{L}_{\text{recon}}^{D_l}$ which uses the l^{th} layer of the discriminator is only the metric for the VAE, the VAE-GAN is trained on a triplet loss 2.11 with $\mathcal{L}_{GAN}$ from Eqn. 2.5 as a *style error*. Here the generator model is the same as the decoder of the VAE as it maps from z to x just like G.

$$\mathcal{L}_{\text{VAEGAN}} = \mathcal{L}_{\text{recon}}^{D_l} + \mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{GAN}} \tag{2.11}$$

Training VAE is hard but training GAN is harder. It is very important to consider that the training of the VAE and the GAN takes place simultaneously. While doing so it is required to *limit the error propagation* of the triplet loss to the entire model. The discriminator should not learn to minimize $\mathcal{L}_{\text{recon}}^{D_l}$, if it does, the discriminator collapses. Better results are observed by restricting the error signal to reach the encoder q as illustrated in Fig. 2.1.7.

As discussed in (2.1.3), VAE as a whole has two objectives - minimize the $\mathcal{L}_{\text{recon}}$ and the $\mathcal{L}_{\text{prior}}$ and a weighing factor β is used to maintain a trade-off between the quality of the reconstruction and the extent of disentanglement. Similarly, when it comes to VAE-GAN, the decoder alone has two objectives. One is to generate samples minimizing the $\mathcal{L}_{\text{recon}}^{D_l}$ and the other is to make sure that the generated samples can fool the discriminator. And the trade off is regulated by using γ to weigh $\mathcal{L}_{\text{recon}}^{D_l}$ and $\mathcal{L}_{\text{GAN}}$ as in Eqn. 2.12.

$$\theta_p \overset{+}{t} -\nabla_{\theta_p}\left(\gamma L_{\text{uike}}^{D} - \mathcal{L}_{\text{GAN}}\right) \tag{2.12}$$

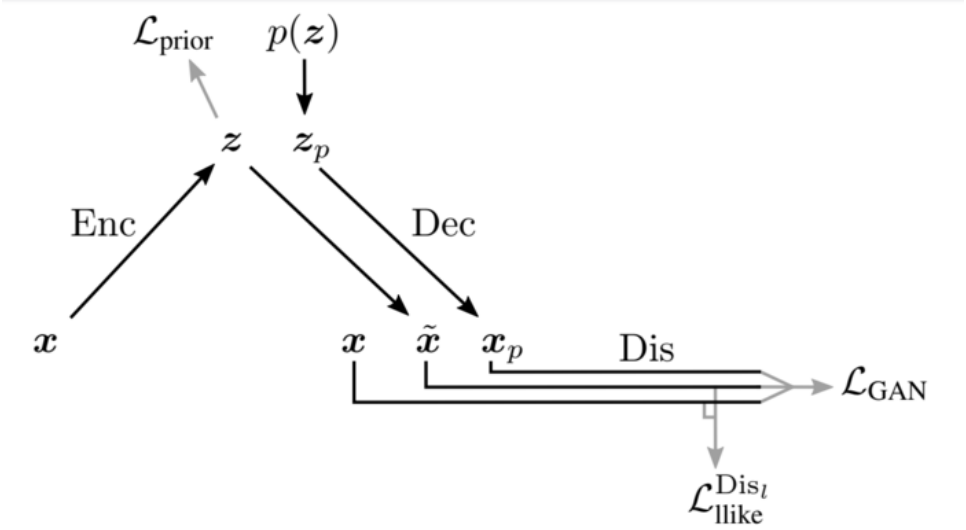


Figure 2.1.7: Illustration of the data flow and the loss of VAE-GAN, where $\mathcal{L}_{\text{recon}}^{D_l} = \mathcal{L}_{\text{llike}}^{Dis_l}$ Image source [23]

.

In the standard GAN training, samples from the prior $p(z)$ are passed to the decoder which generates samples that are then passed to the discriminator. Interesting observation when using VAE-GAN is, sampling x from the encoder $q(z|x)$ further improves the results. As the VAE tries to minimize $\mathcal{L}_{\text{prior}}$, the samples from $p(z)$ and $q(z|x)$ become similar during the training. As the generated samples $p(q(z|x))$ using the encoder are more realistic than $p(p_{prior}(z))$ using the prior, they serve as better adversarial examples for the discriminator. The $\mathcal{L}_{\text{GAN}}$ loss to be used to leverage this benefit is:

$$\mathcal{L}_{\text{GAN}} = \log(Dis(x)) + \log(1 - Dis(Dec(z))) + \log(1 - Dis(Dec(Enc(x)))) \quad (2.13)$$

2.2 Research Area Introduction

In this section, the details of various approaches and the SOTA in 3D HPE are presented along with some works in the sibling tasks of hand pose estimation. This section aims to only give an overview of how the problem of 3D HPE is tackled in the literature, approaches that are directly related to the method proposed in this thesis are presented later in section (2.3), Related Works. The different categories of the approaches mentioned below are not exclusive but are the main aspects of the

described approaches.

2.2.1 Cascading Approach

Numerous works try to estimate 3D human poses from 2D RGB images or 2D joint confidence heatmaps [4, 6, 30, 34, 44]. Most of these methods follow a cascading approach, where an explicit intermediate representation of 2D heatmaps or 2D poses is predicted. This typically involves training a convolutional neural network to find humans in the image or just extract features that correspond to the joints. These features are forwarded to another neural network that learns to estimate the corresponding 3D pose, usually by predicting the depth offsets of the 2D joint or sometimes by predicting the shape and the view parameters of the 3D pose.

For example, [30] proposes a general framework with 3 networks as depicted in Fig 2.2.1. Human detection Network, RootNet, PoseNet. Where the human detection network predicts the region the human is in an image. The RootNet localizes the human's root in the global 3D world. And, the PoseNet predicts the 3D pose of a single person relative to the root. Where the root is a fixed reference point of the human body says, the pelvis.

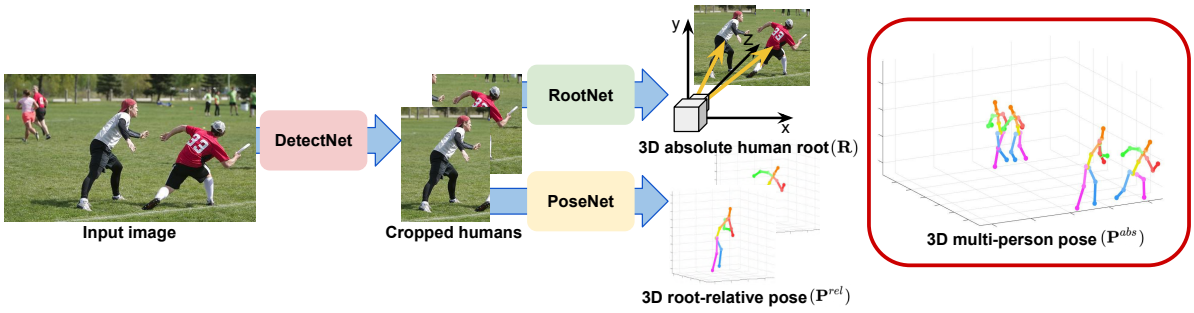


Figure 2.2.1: Pipeline of an absolute 3D HPE network following a cascading approach [30], showing different neural networks such as DetectNet, RootNet and PoseNet that trained on specific subtasks to estimate absolute 3D human pose.

The advantage of such top-down frameworks is the possibility to divide the task of RGB to 3D into smaller, well-studied sub-tasks. This enables explicit supervision of know intermediate states that could be of interest for understanding the representations learned by the network or to use the intermediate output for other auxiliary tasks. Moreover, in this case, it makes scaling single-person pose estimation algorithms for multi-person pose estimation easy, as the majority of the data available mostly consists

of a single person per frame. Most importantly certain modules can be replaced or improvised without affecting or having to re-train the entire pipeline.

2.2.2 Pose Lifting

In contrast to the estimating pose from an image, Pose Lifting works such as [4, 5, 22, 31, 43], focus on estimating 3D poses from 2D poses alone while assuming 2D poses from the SOTA methods in 2D HPE. This category follows the idealogy of cascading based approaches but restricts the study and discussion purely to lifting 2D to 3D pose. These methods include simple linear models as first described in [27] with a series of fully connected linear, batch normalization, dropout layers with residual connections as illustrated in Fig 2.2.2 to regress 3D pose effectively. These simple networks have enough capacity to capture the features of the data as the input and output data are much smaller in dimensions as compared to RGB images.

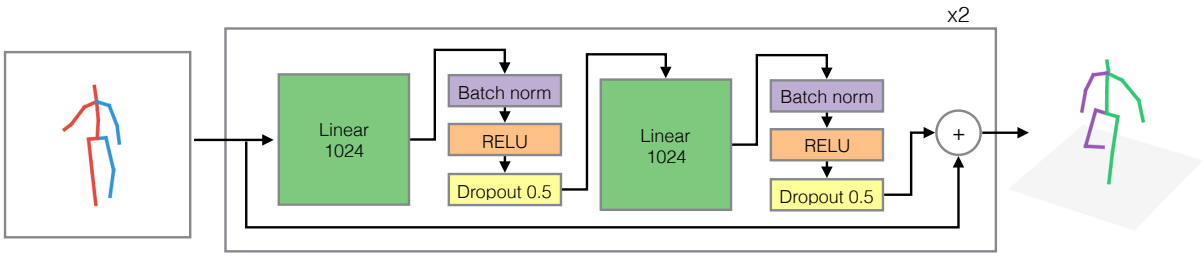


Figure 2.2.2: Architecture of the first deep learning based lifting appraoch proposed by Martinez *et al.* [27]

Non-Rigid Structure from Motion (NRSfM) is another promising lifting method that also leverages images along with 2D annotations. NRSfM deals with the problem of reconstructing 3D shape (pose/point cloud) and cameras of each projection from a sequence of images with corresponding 2D projections (2D keypoints). This approach has been widely used in facial keypoint detection and [21] introduces a deep learning variant for the same. Instead of predicting the 3D coordinates of each keypoint/joint of the 3D pose, [21, 31, 44, 45] predicts the 3D shape and camera pose from 2D pose using this method. The advantage of such an approach is to obtain a disentangled representation of view and 3D pose that is invariant of the view or camera position.

The Lifting approaches facilitate to leverage of the already well established 2D HPE models that are trained on enormous and diverse labeled data. Thus demanding

lesser training data for 3D pose estimation than it would need when learning from images. Since these networks do not have large convolution layers they are computationally inexpensive for both training and inference. Moreover, the 2D and 3D pose data usually can be entirely loaded onto the GPU further accelerating the training procedure. Thus addressing one of the major problems that affect the scalability of 3D HPE models as well as enabling the development of better modular systems by combining the best of Lifting networks with the best of 2D HPE.

2.2.3 Multiple Hypothesis Estimation

As stated in (1.1), 2D-to-3D pose lifting is an ill-posed-inversed problem due to inherent depth ambiguity as multiple plausible 3D poses give the same 2D projection, this shall be further elaborated later in Chapter 3, Data. To address this problem, Jahangiri *et al.* [19] propose a solution a 3D Gaussian Mixture Model (GMM) to learn uniformly sampled 3D poses and conditionally sample to retrieve 3D poses that have the reprojection error within the given limits. Thus estimating multiple 3D poses conditioning on a single input 2D pose, similar to a dictionary-based learning approach.

More deep learning-based approaches such as [24, 25, 34] were later introduced. Out of them, Chen Li *et al.* [24] proposes a variational inference model replacing the GMM with a Mixture Density Network (MDN) which was first introduced by Ye *et al.* [50] to handle occlusions in hand pose estimation. Thus addressing two of the major problems of 3D HPE i.e missing joints from occlusions and variational inference.

Another interesting approach from Sharma *et al.* [34], presents a conditional Variational Auto-Encoder which takes a random sample from a normal distinguish as input and 2D pose as a condition and outputs different 3D pose for different random samples given the same 2D pose. This approach also handles missing joints and provides variational inference. An evaluation technique is also proposed to rank each of the multiple hypotheses by scoring the poses based on joint-ordinal depth relations learned from the images or by oracle score that access 3D ground truth to compute the closest match as illustrated in Fig 2.2.3.

Recent work from Chen Li *et al.* [25] which is an improvised version of their earlier work [24] proposes a weakly supervised approach that is much more similar to that of

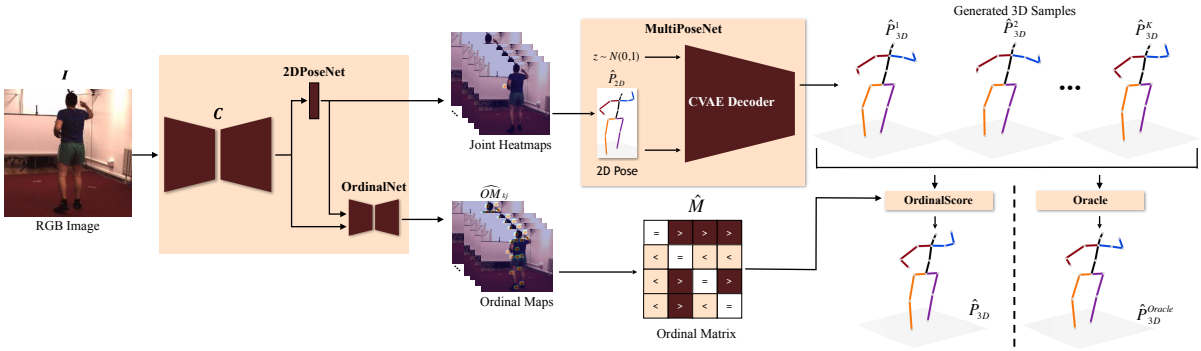


Figure 2.2.3: Illustration of modular framework from [34] that uses a Conditional VAE to estiamte multiple hypothesis for a given 2D pose as condition. The ordinal scoring learnt from joint-depth ordinal relations and the score obtained from the oracle that learns scores higher to poses closest to the ground truth pose are used rank to each of the predicted 3D pose.

this thesis. The details are discussed later in the related works section (2.3). However, all of the mentioned approaches still require 3D poses in one way or other for training and hence do not address the most important bottleneck of obtaining 3D ground truth.

2.2.4 Non-Supervised Learning

The standard way to train 3D/2D HPE is by minimizing the distance between the predicted 3D/2D pose and its corresponding 3D ground truth. The area of 2D HPE is well established and matured with reliable systems deployed in the real world. This was made possible with the high volume of images from diverse settings and the reasonable ease of manual labeling of 2D poses. On the other hand, labeling 3D pose manually is not practical. Though single-person datasets such as Human3.6M [17], Human Eva [36] and, multi-person datasets such as CMU Panoptic [15] provide 3D pose ground truth, they are obtained using Motion Capture (MoCap) systems Fig[2.2.4] which are only limited to indoors or cannot be directly adapted to outdoor environments where the majority of the use cases exist. It is also worth mentioning JTA (Joint Track Auto) dataset [8] that is made using the GTA(Grand Theft Auto) game engine which is technically scalable with its limitations. The datasets from simulations come with the difficulty of domain adaptation to be transferable to the real world.

To overcome this bottleneck, [22] proposes unsupervised training of a generative adversarial network by projecting the predicted 3D pose back to 2D and minimizing its

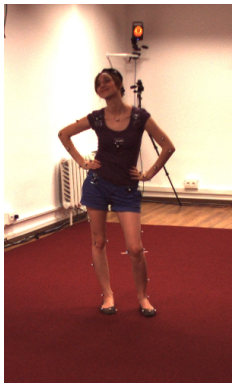


Figure 2.2.4: Image from Human3.6 Dataset [17] of subject wearing MoCap markers

distance with the input 2D pose. And further training a discriminator to distinguish the real 2D pose from the projected poses as illustrated in Fig 2.2.5. Thus removing the need for any explicit 3D annotations besides 2D pose that are either manually labeled or obtained using 2D HPE models. RepNet [43] trains an adversarial network without 2D-3D correspondences in a weakly supervised manner. Moreover, it also does not require camera parameters to project the 3D pose but learns to predict them. Thus enabling better generalization to more diverse data with unknown cameras and poses.

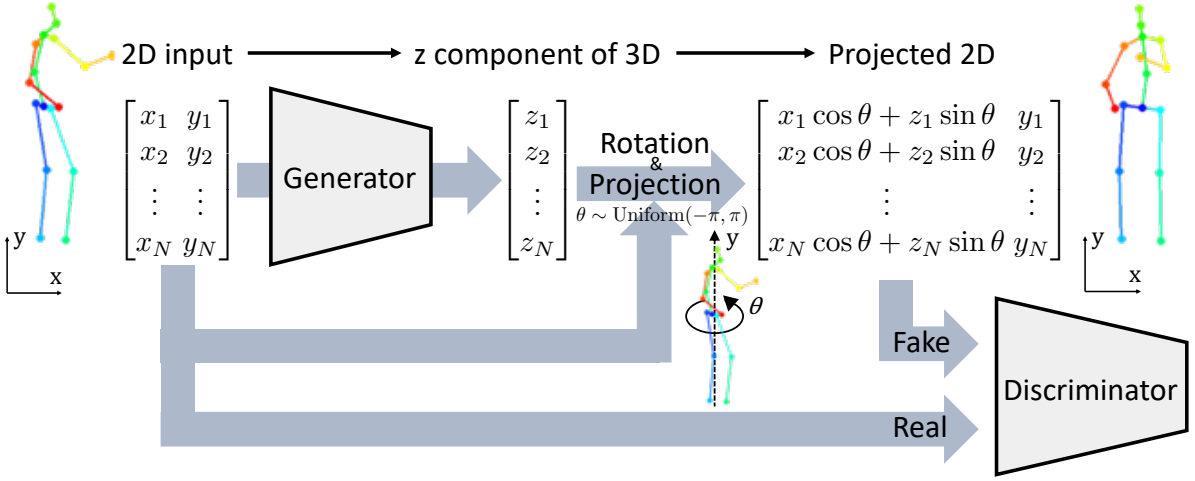


Figure 2.2.5: Architecture of a simple unsupervised adversarial learning architecture proposed by [22]. The network takes 2D coordinates (x, y) of each joint in the pose and predicts the corresponding z-component. The 3D pose achieved by combining the input x, y with the predicted z, is randomly rotated along the y-axis to retrieve a novel view of the 3D pose. This pose is the projected to xy-plane, which would be much different from the input 2D pose. This projected 2D pose should be similar to the real 2D poses in the dataset if the predicted 3D represents the true human pose. Their similarity to real 2D poses is reinforced by a discriminator. The whole training produce is carried out without the need for a 3D pose in any shape or form.

To test the maximum capability of Pose Lifting networks, [5] proposes a combination of unsupervised and adversarial learning that mainly leverages the property of *plane-invariance*. It is the property that 2D projections of a 3D pose from different camera viewpoints, when lifted should produce identical and the original 3D pose. In this method, the predicted 3D pose is rotated in random angles and is reprojected to 2D in a different Point of View (POV). A discriminator is then used to evaluate if this new 2D pose is in the possible pose distribution which is learned from 2D pose datasets alone. These steps are redone in reverse order to obtain the original 2D input. This cycle provides three intermediate representations of the single 2D input that the models learn from. Additionally, this approach exploits the temporal consistency in the datasets as well as integrates a domain adaptation network to learn from different datasets and distributions to achieve comparable results to that of the methods that require more supervision. However, due to the inherent ambiguity in lifting 2D pose to 3D and as the images are not captured with orthogonal cameras, reprojection of 3D pose is not necessarily consistent with the ground truth as the camera intrinsic and extrinsic parameters are not taken into account. Hence it is challenging to match the performance of models trained on 3D ground truth.

2.2.5 Multimodal Representation Learning

Another interesting approach is training VAEs using multiple modalities like images, poses, depth maps [12, 35, 38, 42]. Multimodal Variational Auto-Enocder (MVAE)s learn representation from different modalities in the same latent space. True multimodal learning needs to fulfill 4 criteria as follows: i) *Latent Factorization* - Implicit factorization of latent space into private, shared subspaces based on modality as illustrated in the figure[2.2.6]. ii) *Coherent Joint Generation* - Coherence in generations of different modalities from the same latent value with respect to the shared aspects of the latent. iii) *Coherent Cross Generation* - Generation of one modality conditioned on data from different modality while preserving the similarity between them. iv) *Synergy* - Enhancement in generation quality of one modality as a result of learning representations of different modalities.

Mixture-of-Experts Multimodal Variational Auto-Enocder (MMVAE) proposed by [35] fulfills all 4 of the above-mentioned criteria learning representations of image and text data, while other approaches focus on leveraging specific advantages of multimodal

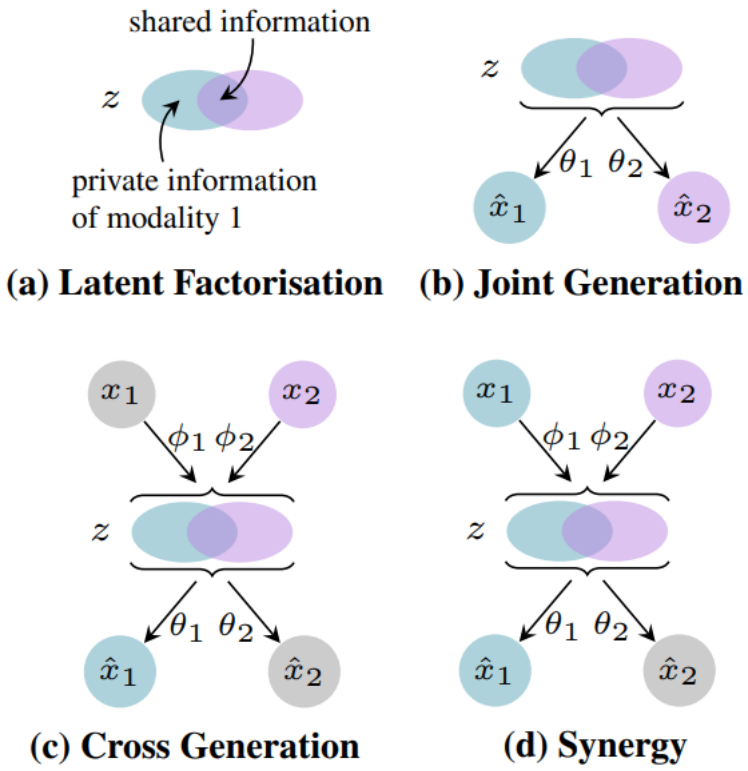


Figure 2.2.6: Criteria for Multimodal Generation [35]

learning. Consider the cross-modal learning for 3D Hand Pose Estimation proposed by [38]. It involves training an encoder-decoder pair to learn image representation, and another such pair to learn 3D hand pose representations in the same latent space. This training procedure focuses on cross-generation and synergy. That is, using the shared latent space of the image and pose representations, the RGB image encoder combined with the pose decoder can generate 3D poses and vice versa while preserving the commonality between the conditioned and the generated data. With this approach, it is possible to train a VAE for 3D HPE from RGB images without explicit intermediate stages like the earlier mentioned cascading approaches. Making it more efficient and fast for both training and inference without compromising the modularity offered by cascading approaches.

2.3 Related Work

In this section, works that are directly related to the thesis are discussed in more detail. Some are the best examples of their kind and have already been discussed thoroughly.

The basic idea of the thesis is to learn 3D HPE just from 2D pose data without using 3D ground truth in any shape or form. Thus developing a method that can exploit the huge amounts of 2D pose data that can be generated using state of the art 2D pose networks on diverse images from the real world. The following approaches use weakly supervised or unsupervised approaches to accomplish the same. These serve as the inspiration for many of the choices taken in this thesis and also help understand the possibilities of reducing the need for explicit 3D supervision.

To the best of knowledge acquired during the period of the thesis, [5, 7, 22, 31] are the main approaches that do not use 3D supervision in any way. While [25, 43] are among the main approaches that use 3D supervision to train the discriminator alone. The approaches that are not mentioned are either the approaches the above mentioned are built up or have been missed during the literature study or most likely published after finishing this thesis report.

[5, 7, 22] can be viewed as a series of approaches that are built on one another in the same order. They take 2D poses as the input and learn to predict the depth offset for each joint to reconstruct 3D. Out of the three Ching *et al.* [5], using the plane invariance, geometric self-supervision, and adversarial learning as discussed earlier in (2.2.4), achieves the SOTA results compared to fully supervised methods and also present ways to use domain adaptation network, temporal consistency to further integrate more datasets and improve the performance. Thus directly address the hurdles of scaling the 3D HPE network to the real world. However, they also acknowledge the fact that most of the predictions made by SOTA 2D HPE model on real-world images have missing joints. Since the proposed approaches only predict the depth of every joint, the error from the 2D input pose is directly propagated to the 3D prediction. More importantly, it is not possible to use most of the data that is generated from 2D pose models. Hence it is very crucial to handle the problem of ***missing joints*** to truly unlock the potential of unsupervised learning.

Wandt *et al.* [43] also mentioned in (2.2) proposes an architecture that learns to predict the whole 3D pose, while also learning the camera parameters that are used to project the predicted 3D to 2D. The idea behind the camera parameter network is to learn the view angle given pose to generalize to unknown cameras. The pose network learns to converge the predicted 2D reprojections while using a GAN trained on ***3D ground truth labels*** to supervise the predicted 3D pose. Though there is no direct

error propagation from 2D input to 3D, it is important to note the problem of missing joints is not yet addressed. However, there another fundamental problem of ***depth ambiguity*** persists.

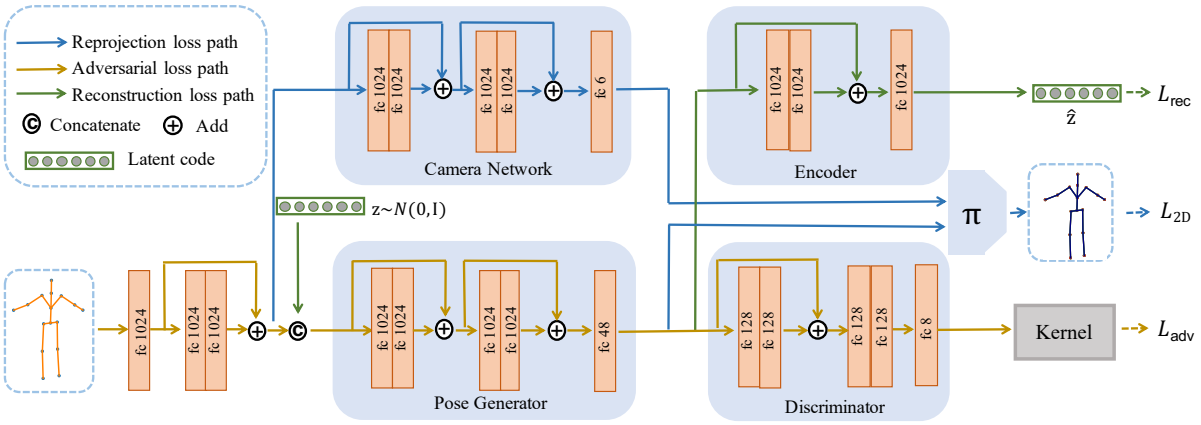


Figure 2.3.1: Illustraion of the weakly supervised multiple hypothesis generation architecture proposed by [25]

Chen Li *et al.* [25], building upon their previous work [24] that is explained earlier in (2.2.3), proposes a weakly supervised variational inference model as the proposed method in this thesis was developed. The new approach [25] was inspired by the architecture of Wandt *et al.* [43] that does not require direct 3D supervision. This model first encodes the input 2D pose to a latent representation. This representation is then concatenated with a *latent code* to produce a 3D pose as well as used by the camera network to predict the camera pose. Instead of directly supervising the predicted 3D pose with the ground truth, it is projected back to 2D using the predicted camera view which is self-supervised as it should be the same as the input 2D pose.

This self-supervision trains the network to only output 3D poses that are close to input 2D in a particular view but unconstrained from another view. To ensure, the predicted 3D is feasible and close to ground truth a WGAN is used as illustrated in Fig 2.3.1. The predicted 3D pose is passed to a critic network that scores the poses on how realistic they are. This encourages the pose network to generate poses that are realistic, in other words, indistinguishable from the 3D ground-truth poses. This results in a realistic 3D pose that is close to the 2D input pose which is most likely close to its corresponding 3D ground truth. In addition to this, an encoder is trained to reconstruct the latent code to ensure diversity and prevent mode collapse of the WGAN.

This variational inference of 3D pose addresses both the problems of depth ambiguity and missing joints. However, this weakly supervised still requires 3D ground truth

poses to train the discriminator and does not address the problem completely. The method proposed in this thesis is similar to this approach but does not require 3D in any shape or form, and uses much smaller architecture without the camera and latent vector encoder networks while addressing all the major problems.

Canonical 3D Pose Networks for Non-Rigid Structure From Motion (C3DPO) [31], which is also discussed in (2.2.4) handles missing joint and is fully unsupervised but does not have the capability of predicting multiple hypotheses for a given 2D pose. This approach that uses NRSfM in an unsupervised way, does not yield good results compared to other approaches. In this thesis, we present a method that has the merits of the above approaches and addresses all the aforementioned problems.

Chapter 3

Data

This chapter discusses the datasets used in the thesis, as well as the pre and post-processing steps that are performed to facilitate the training and validation of the proposed method. In addition to this, a couple of concepts that motivate the choices or that are directly used in the process of the data are explained with the help of illustrations.

The main dataset used in the thesis is *Human3.6M*: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments [17]. Most of the related works benchmark their methods on Human3.6M and it also is freely accessible to academics on request. For further evaluation of model performance in the wild, outdoor datasets that do not have 3D ground truth such as *3DPW*: 3D Poses in the Wild [26] would be used.

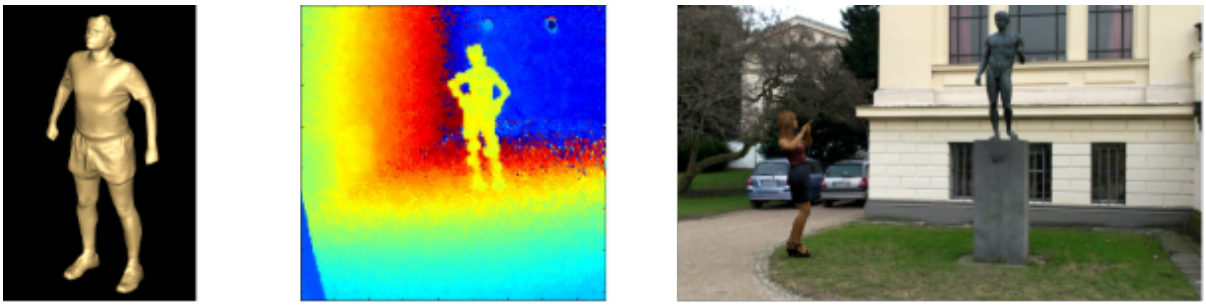
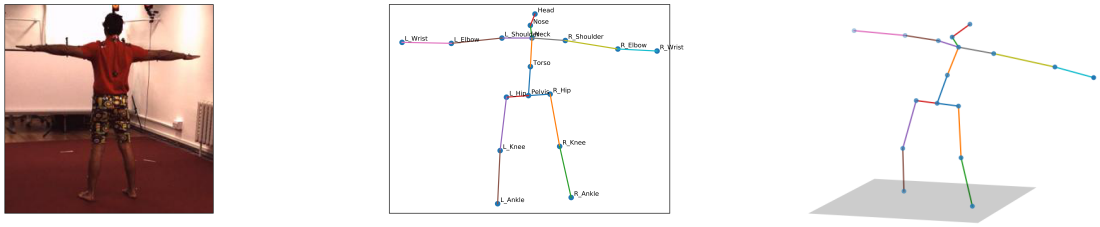


Figure 3.0.1: Additional modalities such as (from left to right) Full body model, depth from time of flight and mixed reality data are availble in Human3.6M dataset [17]. These datasets can be used for human body estimation, depth map estiamtion or inferring absolute 2D or 3D pose directly from full image i.e without cropping the region with a single person.



(a) (b) (c)

Figure 3.1.1: (a) RGB image from one of the 4 cameras. (b) Corresponding 2D pose obtained from the 3D pose to the right with joint labels. (c) Corresponding 3D pose rotated for a better view of the spatial distribution of the joints

3.1 Human3.6M

Human3.6M is a large scale indoor dataset with 3.6 million human poses collected with 4 cameras at different angles using a highly accurate maker-based MoCap system. The dataset constitutes 15 diverse motion and actions in various everyday scenarios namely, Directions, Discussion, Eating, Greeting, Phoning, Photo, Posing, Purchases, Sitting, Sitting Down, Smoking, Waiting, WalkDog, Walking, and WalkTogether. These actions are performed by 11 professional actors wearing a variety of realistic clothing. The datasets provide synchronized 2D and 3D data including full-body scans as shown in Fig 3.0.1. It also includes mixed-reality test data created using animated human models to cover huge variations of background, clothing, illumination, occlusion, and camera angles.

The data of interest is mainly the 2D pose for training, 3D poses for evaluation, and images for qualitative analysis as it is sometimes challenging even for the human eye to estimate 3D pose just from the 2D skeleton. Both the poses are composed of 17 joints or keypoint annotations namely, Pelvis (also referred to as Root), Torso, Neck, Nose, Head, right and left - Hip, Knee, Ankle, Shoulder, Elbow, Wrist. An image sample from the dataset with its corresponding 2D and 3D pose is illustrated in Fig 3.1.1.

As illustrated in Fig 3.1.2, the data is collected from a indoor environment with multiple cameras and MoCap sensors. The obtained coordinates of the 3D pose from MoCap data are global and are relative to a fixed origin in the recording room. In addition to the global 3D pose, camera relative, i.e local 3D pose is also obtained using the camera's location with respect to the global origin. The global poses are useful for methods that try to exploit the multiview information and local poses that are relative to the camera

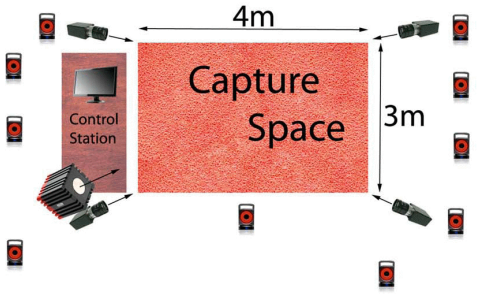
can be used for absolute pose estimation. The 2D pose from each view is obtained by projecting the 3D pose using the parameters of the respective camera. This 2D pose is relative to that particular camera and is with respect to the full-scale image that captures the entire scene in the camera's field of view. Along with the RGB images, global and local 3D pose, and 2D pose, other metadata is available. This metadata includes intrinsic and extrinsic parameters of each camera, an identifier for each of the samples of a particular action sequence, bounding box coordinates of the human in the full-scale image, subject, action, subaction, and camera ids for each sample. This data is processed to suit the task requirements. These details are later presented in section (3.4).

Type of action	Scenarios	Train	Validation	Test
Upper body movement	Directions	83,856	50,808	114,080
	Discussion	154,392	68,640	140,764
Full body upright variations	Greeting	69,984	33,096	84,980
	Posing	70,948	25,800	85,912
	Purchases	49,096	33,268	48,496
	Taking Photo	67,152	38,216	89,608
	Waiting	98,232	54,928	123,432
Walking variations	Walking	114,468	47,540	93,320
	Walking Dog	77,068	30,648	59,032
	Walking Pair	76,620	36,876	52,724
Variations while seated on a chair	Eating	109,360	39,372	97,192
	Phone Talk	132,612	39,308	92,036
	Sitting	110,228	46,520	89,616
	Smoking	138,028	50,776	85,520
Sitting on the floor	Sitting Down	112,172	50,384	105,396
Various Movements	Miscellaneous	-	-	105,576
Total		1,464,216	646,180	1,467,684

(a) The number of 3D human poses in Human3.6M in training, validation and testing aggregated over each scenario. We used 5 subjects for training (2 female and 3 male), 2 for validation (1 female and 1 male) and 4 subjects for testing (2 female and 2 male). The number of video frames is the same as the number of poses (4 cameras capturing at 50Hz). The number of TOF frames can be obtained by dividing the table entries by 8 (1 sensor capturing at 25Hz).

MoCap System		DV System	
No x Sensor	10 x Vicon T40	No x Sensor	4 x Basler piA1000
Resolution	4 Megapixels	Resolution	1000x1000
Freq.	200Hz	Freq.	50Hz
Sync	hardware	Sync	hardware
TOF System		**Body Scanner**	
No x Sensor	1 x Mesa SR4000	Sensor	Vitus Smart LC3
Resolution	176x144	No. Lasers	3
Freq.	25Hz	Point Density	7dots/cm3
Sync	software	Tolerance	< 1mm

(b) Technical summary of our different sensors.



(c) Floor plan showing the capture region and the placement of the video, MoCap and TOF cameras.

Figure 3.1.2: Human3.6M data collection details. Image source [17]

3.2 Camera projection

The projection of the poses in the thesis is using a simple pinhole camera model as illustrated in Fig 3.2.1. The image of an object, here a person depicted as a 3D pose, is formed on the other side of the camera, or the lens at a distance of f. Where f is the focal length of the camera. This image that is formed on the true image plane behind the camera is upside down. For better understanding and comparison the image plane has been translated before the camera and since it's before the pinhole, it is not inverted. Assume the two lines passing through from the head and root joints

of both the 3D pose and 2D image to the camera make an angle of θ. Also, assume the distance between the 3D pose and the camera as c, and the length of the upper half of the pose i.e, the distance between the head and the root joint as k. Considering the similar triangles formed by the two rays and the line joining the head and root of the pose in the image and world frame respectively, the ratio of lengths of upper halves to their distance from the camera should be the same. Hence the length of the upper half of the 2D pose is kf/c. This relationship is used in the following sections on data preprocessing. The ground truth 2D as explained earlier is obtained by similar projects taking the camera intrinsic and extrinsic parameters into account.

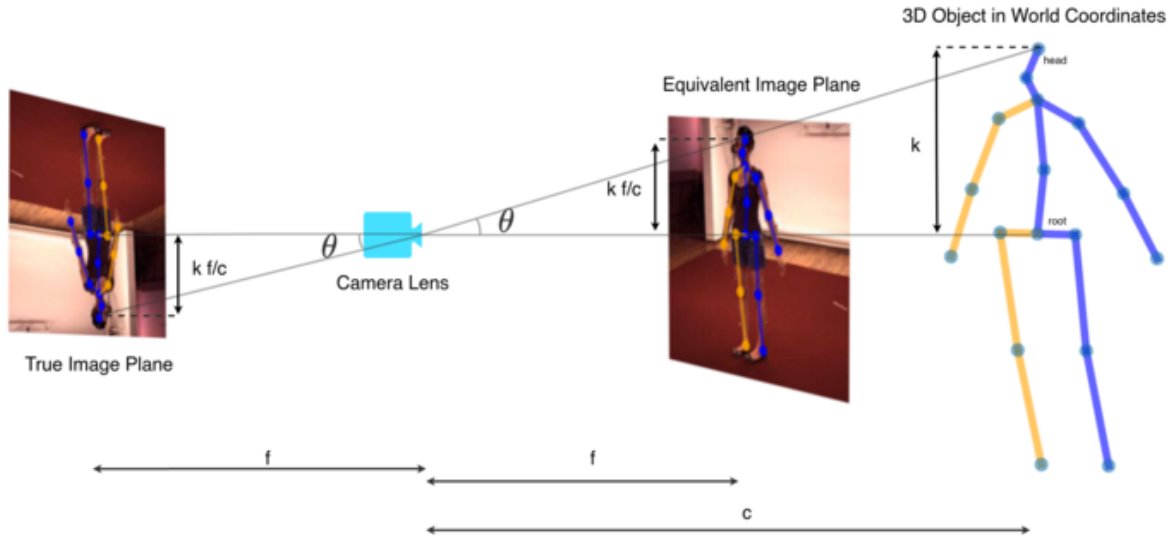


Figure 3.2.1: Illustration of a pinhole camera model

3.3 Depth Ambiguity and Camera Modeling

One of the main problems discussed in (2.3) is the depth ambiguity. The Fig 3.3.1 illustrates a case where a 3D pose gives the same 2D pose when scaled and shifted by the same factor, here 2. The second case shows the same 3D pose giving different projections for different focal lengths i.e the poses are kept the same while scaling the focal length and distance. Hence it is not possible to predict absolute pose from a single view alone. However, works such as [30] try to tackle these challenges by learning the scale of the humans based on the features extracted from images. While works such as [25, 43] that do not use image features learn the camera parameters that are used to project 3D to 2D for learning 3D pose in a self-supervised manner.

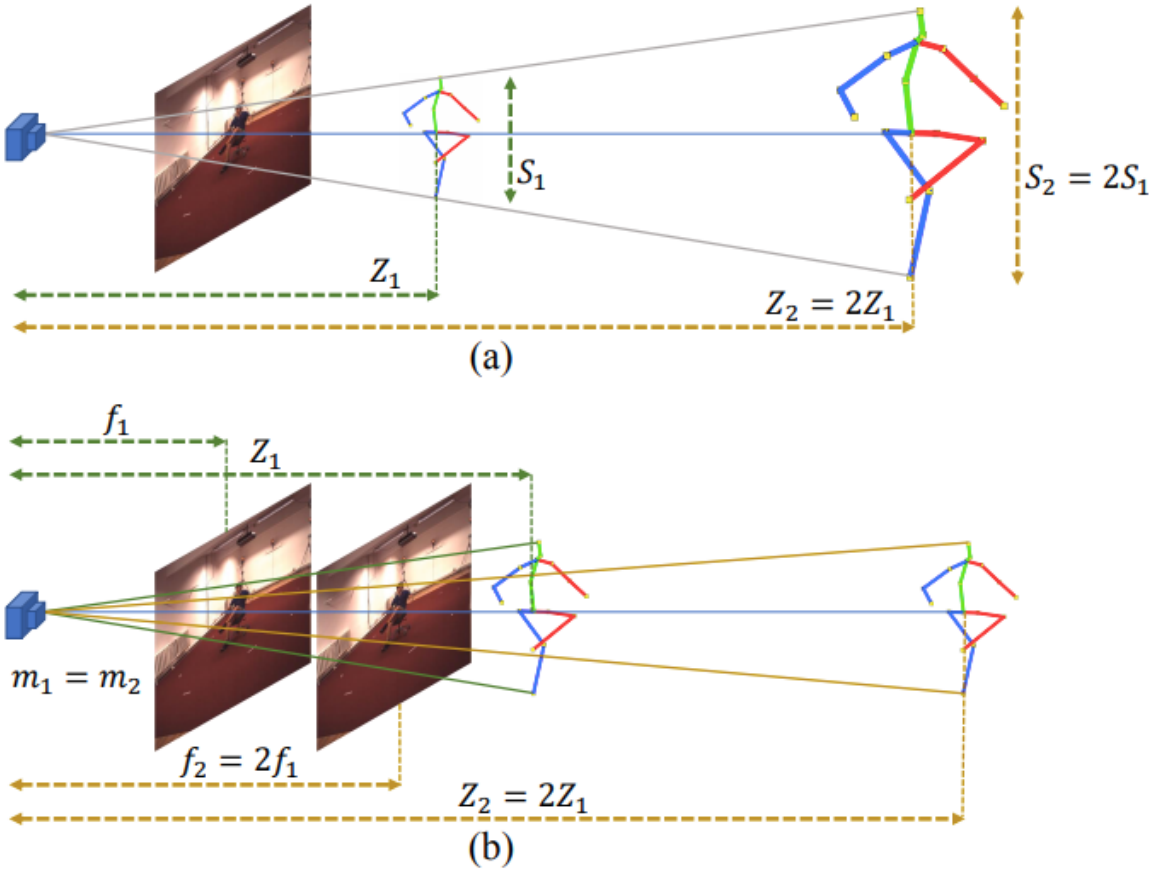


Figure 3.3.1: Cases of depth ambiguity for a given 3D pose (a) Multiple 3D poses that result in the same projection for a camera of a given focal length. (b) Same pose at different distances giving the same projection for different focal lengths. Image Source [4]

Assuming the focal length and distance are known constants, there still exists infinite 3D poses, that give the same 2D projection. However, only a smaller subset of poses are plausible i.e poses that can be articulated by the human body. Fig 3.3.2 illustrates a few 3D poses that when projected from the same camera viewpoint produce the same 2D pose (the first image). The first 3D pose (in red and blue) is the ground truth pose used to obtain the 2D pose where the rest of the 3D poses (in mono-colors) are plausible solutions that can be articulated by the human body.

3.4 Processing

The methods explored as part of this thesis, use images, 2D, and 3D human pose from the dataset besides metadata. Where 2D poses are used as training data, 3D poses

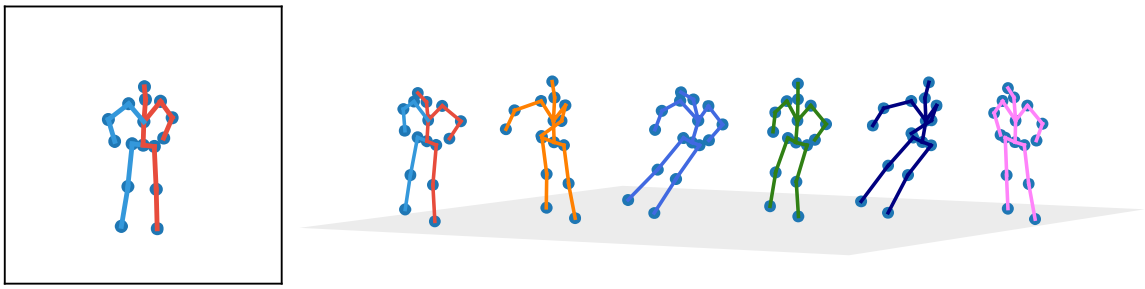


Figure 3.3.2: Cases of depth ambiguity for a given camera and fixed distance from the 3D pose. Multiple 3D poses that result in the same projection for a given focal length and at the same distances from the camera. The variation is only in the depth z component of each joint. Modification of an image from [4]

are only used for evaluation and images are for qualitative analysis. The following describes the modifications done to the raw data for the tasks related to single person detection that are consistently followed in the field. This is followed by the pre-processing steps done to the data that is specific to the presented work.

3.4.1 Standard Pre-processing

As discussed in section (3.1), the 3D pose in the dataset that is obtained from the marker-based MoCap is in a global reference frame. These poses using the camera parameters are transformed into the camera coordinate frame. For the task of predicting a 3D pose from either images or 2D pose, it is unrealistic to directly estimate all the joints of the pose in a global frame. The works that predict absolute 3D pose are presented in Chapter 2 and are fully supervised or exploit multi-view data and is not in the scope of this thesis. Since the focus is on unsupervised estimation and considering the depth ambiguity challenges we confine the scope to estimating pose relative to an origin point. The pelvis joint is generally considered as the origin or the root joint relative to which the coordinates of other joints are described.

The full-scale images for such tasks are cropped using the bounding box annotations as a guide to obtaining images of aspect ratio 1:1. These cropped images are then scaled to 256x256 images. It is important to note that the 2D annotations are relative to the full-scale image and hence the projections are no more synchronized to the images. The translation to these 2D poses is later done exclusively for visualizing the data. Since the image data is not utilized in the training or testing procedures, it is not required to

maintain consistency with 2D or 3D annotations.

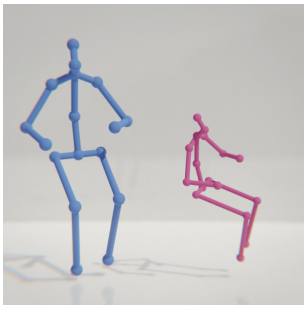
The 2D and 3D poses are translated such that the root joint, the pelvis is at the origin. Since the pelvis is fixed at the origin, we remove it from the pose after translation so that the network is not required to learn a constant. Removing Pelvis, 16 out of the 17 joints or keypoints remain. The poses are generally normalized before feeding them to the neural network. These are the steps that were performed and were sufficient for the initial supervised version of the method. Further steps which are inspired by [5], are performed to make unsupervised adversarial training possible.

3.4.2 Pre-processing for Adversarial training

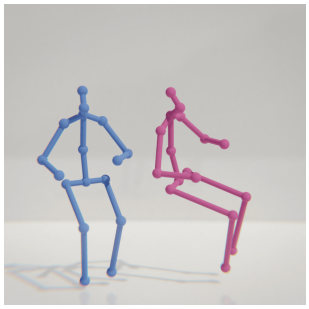
Considering the different cases of depth ambiguity explained in the previous section, a simple unit pinhole camera is assumed for all the data. That is a camera with a unit focal length, whose image plane is at a distance of 1 unit. Hence the 3D pose that forms an image on this image plane is further from the camera as depicted in the pinhole camera model figure 3.2.1. So the estimated 3D poses can not be at the origin, hence they are fixed at a distance of c units. It is worth clarifying that only 2D poses are present and 3D are predicted by the neural network. Hence the 2D poses should be processed to achieve this.

Referring to the pinhole camera figure 3.2.1 again, the scale of the 3D pose that should be ideally predicted by the model is $\frac{c}{f}$ times that of the 2D pose. The predicted 3D is projected to 2D just by dividing the x, y coordinates by the z component as it is a simple pinhole camera. The pose is rotated and project again and passed to a discriminator network following the VAE-GAN network explained in Preliminary Concepts section (2.1). For numerical stability, we process the 2D pose such that the upper half of the desired 3D pose is approximate of unit length. To achieve this the 2D poses are scaled such that the mean distance from the head to the root joint is $\frac{1}{c}$ ($\frac{kf}{c}$ where k and f are 1 unit each). c is set to 10 for all the experiments following [5].

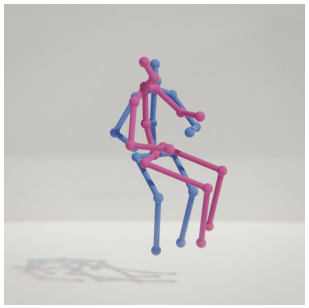
Following the above processing procedure, the predicted 3D pose is not to scale with the ground truth 3D poses and direct comparison is not feasible. Following the related unsupervised works such as [5], we only evaluate the 3D pose by aligning the prediction to the ground truth pose using rigid-body alignment. This is also referred to as



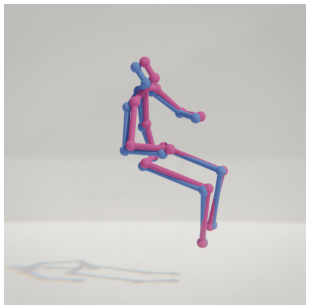
(a)
Prediction and Ground Truth



(b) Scaled to the same size



(c) Translated to the same position



(d) Rotated to the same orientation

Figure 3.4.1: Procrustes alignement of a 3D pose (pose in blue) to minimize the distance to the reference pose (pose in pink)

Procrustes alignment [1] as illustrated in Fig 3.4.1. Since the transformation is taken care of by the algorithm, no other post-processing procedure is required for the proposed unsupervised method. More details about the evaluation are presented in Chapter 4, the Method.

[1] Procrustes Analysis `https://en.wikipedia.org/wiki/Procrustes_analysis`

Chapter 4

Method

This chapter entails the details of the neural network such as the architecture, loss/objective function along with the training and evaluation procedures of the proposed method. Other choices such as activation functions, optimizers, initialization, etc that completes the architecture are presented later in Chapter 5 Experiments, along with the reasoning behind the choices and the compilation of essential components required to reproduce the entire network.

4.1 Architecture

To predict root-relative 3D pose solely using 2D poses, a hybrid network using VAE and GAN as discussed in (2.1.6) is employed. In contrast to the VAE-GAN hybrid proposed by [23], the GAN loss gradient is propagated to both the decoder and the encoder. In other words, the VAE as a whole is considered as the generator network. The overall architecture is composed of 3 models Encoder, Decoder, and Discriminator as illustrated in Fig 4.1.2. This section elaborates on each of the models and how they are interconnected.

4.1.1 Encoder

Adding to the explanation of VAE in (2.1.2), the role of the encoder Q, is to take a 2D pose $\mathbf{x}_i = (x_i, y_i)$ with its root located at the joint and of upper body length of c units as an input. Where $i = 1...J$ and J denote the number of joints of the pose. And output

the corresponding embedding in a d dimensional latent space in the form of mean μ and standard deviation σ of each dimension.

$$Q_{\theta_q}(\mathbf{x}) = \mu, \sigma \tag{4.1}$$

Where θ_q denotes the learned parameters of the encoder during training. This encoder is composed of an upsampling layer that scales the $2 \cdot J$ dimensional input to match the number of hidden neurons h of the encoding module. The encoding module q is made of n residual block composed of 2 Fully Connected (FC) layers following the related works, to allow comparison. This residual block structure is repeated for the decoder and the discriminator. The encoding block is followed by 2 FC (linear) layers that downsample the hidden representation of dimension h to match the latent space dimension d. The output of the two downsampling layers represents the mean and standard deviation respectively. However, in practice, the encoder is designed to predict log-variance instead of the standard deviation to have a better distribution of values and gradient. The layers of the encoder, decoder, and the discriminator models are kept similar for simplicity and as done by the related works. The residual blocks used in all the 3 models are identical and is illustrated in Fig 4.1.1.

Both FC layers of the residual block are followed by batch normalization, activation, and dropout layers of the same dimensionality. While other FC layers are only followed by an activation function. The linear layers in the residual blocks of the VAE contain weights but do not have the bias component.

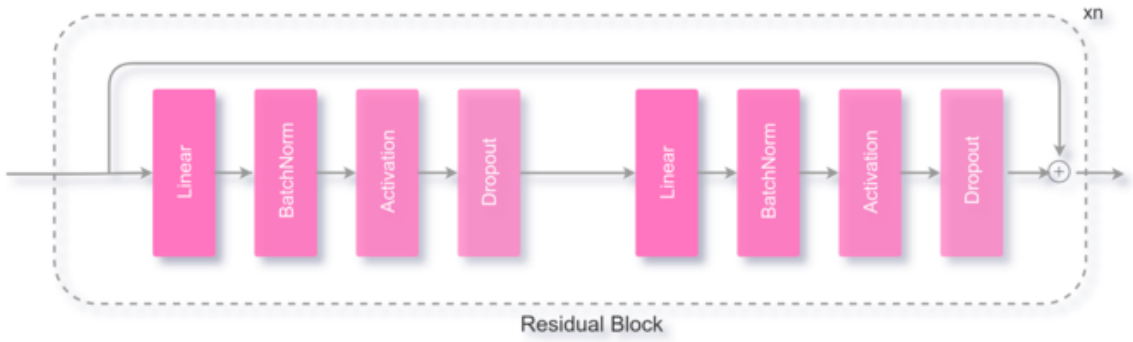


Figure 4.1.1: Illustration of the residual block made of 2 Linear layers of the same input and output dimensionality. The intermediate BatchNorm, activation, and dropout layers maintain the same dimensionality. These blocks are repeated n times based on the configuration of the model.

4.1.2 Decoder

The decoder P takes the 2D pose embedding z derived from the mean and log-variance predicted by the encoder and estimates the corresponding 3D pose $\mathbf{Y}_i = (x'_i, y'_i, z'_i)$. Similar to the encoder, $i = 1...J$ and J denote the number of joints of the pose. The reparametrization trick is used to make the process differentiable and induce variance. This is done by scaling the standard deviation obtained from the log-variance with a random sample ϵ from a unit gaussian distribution $\mathcal{N}(0, 1)$. The sum of the scaled standard deviation and the mean gives the sample z.

$$\begin{aligned} z &= \mu + \sigma \odot \epsilon \\ P_{\theta_p}(z) &= \mathbf{Y} \end{aligned} \tag{4.2}$$

Where θ_p represents the learned parameters of the decoding network. Similar to the encoder, the decoder consists of an upsampling layer that scales the sample z of dimensionality d to match the number of hidden neurons d in the decoding module. The decoding module is identical to the encoding module and consists of n residual blocks similar to that of the encoder. This decoding module is followed by a FC layer to downsample the neurons to predict the 3D pose of dimensions $3 \cdot J$. As the goal is to predict root-relative pose with the root at the origin, the output joints are considered to be relative to the origin (0, 0, 0).

Since the predicted 3D pose is to be projected to back to the 2D pose to form the reconstruction loss, the distance between the head and the pelvis joint should approximately be of unit length as explained in (3.4). To achieve this, a Tanh activation function is used at the downsampling/output layer to obtain the predicted 3D pose (all joints) in the range [-1, 1].

4.1.3 Reprojection

Referring to the camera modeling section (3.2), the 3D that corresponds to the pre-processed 2D pose is located at a fixed distance c from the camera. The 3D pose is considered to be predicted relative to the origin for simplicity and symmetry and is to be translated c units aways from the camera. Another required adjustment is the scale. The range of coordinates predicted is in the range [-1,1]. But the length of the lower

half of the pose can be longer than the upper half and usually is the case. Hence the predicted 3D pose is scaled by a factor of 1.3, which is the ratio of the mean length of the upper and lower halves. This is to enforce that the length of the upper half is 1 unit while covering the true range of the lower half and get the best 2D re-projection.

This scaled 3D pose prediction is directly projected from the same point of view to get a 2D projection that corresponds to the input 2D pose. This similarity is used as the reconstruction loss for constrained optimization of the VAE.

$$\begin{aligned} \mathbf{Y}' &= \mathbf{Y} * 1.3 + (0, 0, c) \\ \mathbf{y} &= PP(\mathbf{Y}') \end{aligned} \tag{4.3}$$

Where PP refers to perspective projection and $\mathbf{y}$ denotes the 2D projection that corresponds to the input 2D pose. The scaled 3D pose is also randomly rotated by uniformly sampling an azimuth angle from the range $[-\pi, \pi]$ and elevation range in the range $[-\pi/9, \pi/9]$. The 2D pose obtained from the projection of this rotated 3D pose gives a different point of view, a ***novel view*** of the 3D pose that is different from the views of the subject in the dataset. This novel view is used for unconstrained optimization of the VAE using the discriminator. The elevation angle has been followed by other works but is not observed to have any visible impact.

$$\begin{aligned} \mathbf{Y}'_{rot} &= \mathbf{R} * (\mathbf{Y} * 1.3) + (0, 0, c) \\ \tilde{\mathbf{y}} &= PP(\mathbf{Y}'_{rot}) \end{aligned} \tag{4.4}$$

Where $\mathbf{R}$ refers to the rotation matrix formed using the randomly sampled angles. $\tilde{\mathbf{y}}$ refers to novel view of the 3D pose.

4.1.4 Discriminator

The discriminator takes the novel view 2D pose of the predicted 3D pose $\tilde{\mathbf{y}}$ along with the real 2D poses $\mathbf{x}$ from the dataset as the input and classifies which 2D pose belongs to which category, real or fake (generated novel view). The real and novel views need not correspond to the same pose, the goal of the discriminator is to learn the general ability to distinguish the 2D poses from the dataset (real) from the predicted poses

(fake).

$$D_{\theta_d}(\tilde{\mathbf{y}} \cup \mathbf{x}) = \left\{p^i_{class} | i \in 1, ..n(\tilde{\mathbf{y}} \cup \mathbf{x})\right\}$$
$$where, 0 \leq p_{class} \leq 1 \tag{4.5}$$

Where θ_d refers to the learned parameters of the discriminator and p_{class} here denotes the probability of the pose's class, with 1 being real and 0 being fake. The discriminator mimicking the encoder and the decoder upsamples the $2 \cdot J$ dimensional novel 2D pose to h dimensions of the main module. The main module just as other models are composed of n residual blocks. However, the batch normalization layer is removed in the discriminator's residual blocks following standard practices. It is important to note that n need not be the same for all the 3 models. The learned features are downsampled to predict the probability of the pose being real or fake. The sigmoid activation function is used at this last layer as the required values are in the range [0, 1].

4.2 Loss Functions

The loss functions used in this VAE-GAN are similar to the ones explained in (2.1.6). The hybrid training proposed in [23] uses the similarity between the prediction and input estimated by the discriminator *replacing* the element-wise loss. While the proposed approach exploits the element-wise loss and the similarity loss by using them on different transformations of the prediction (direct view 2D and novel view 2D). The formulation of each loss function of the hybrid with respect to the human pose data is as follows.

4.2.1 Reconstruction Loss

Since the goal of the proposed method is to learn 3D pose without requiring any of such training data, the approach uses 2D pose as its supervisory data. This self-supervision is achieved by reprojecting the 3D pose in the same view to retrieve the 2D pose that should ideally be the same as the input 2D pose as discussed in the Reprojection (4.1.3) part of the previous section. The objective of the model here is to converge the distance (element-wise) between the predicted 2D pose $\mathbf{y}$ and the input 2D pose $\mathbf{x}$. L1 loss is

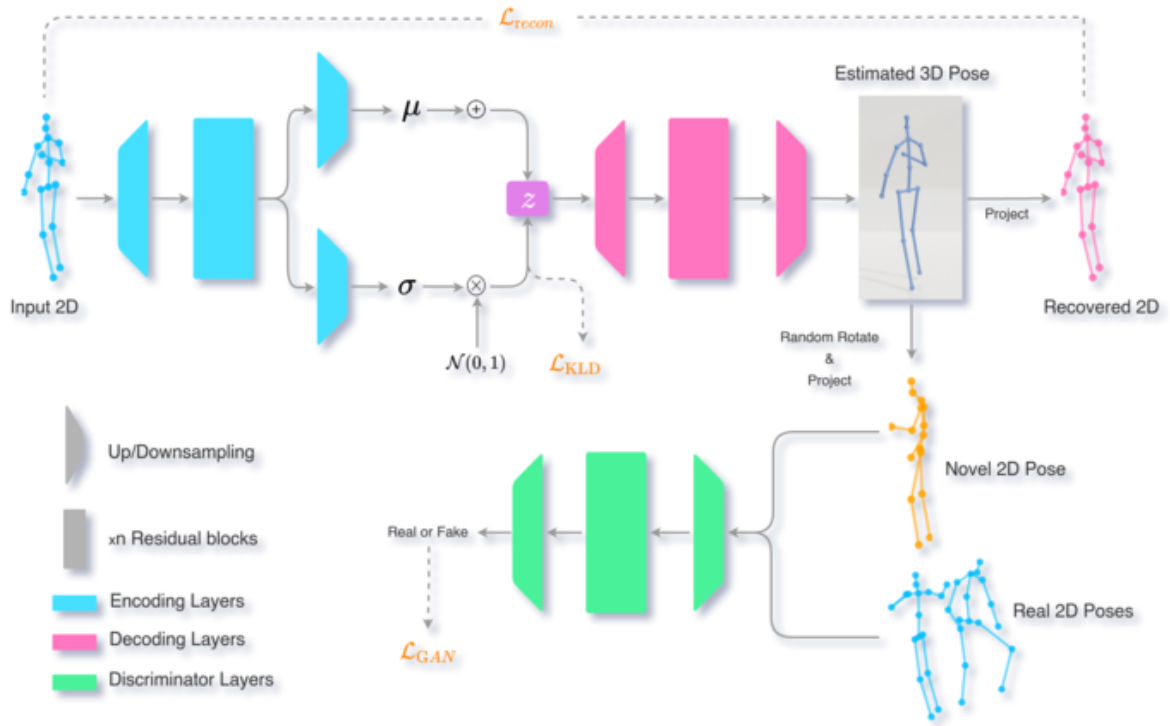


Figure 4.1.2: Illustration of the neural architecture of the proposed method. The network components in blue, encode the 2D pose to a latent representation in terms of mean μ and standard deviation σ of the distribution. While the components in pink, sample z from this latent space and decode the corresponding 3D pose. The 3D pose is projected directly to 2D space for a constrained optimization using the input 2D pose in the camera view and randomly rotated and projected for unconstrained optimization using the discriminator D in a novel angle. The data that contributes to the loss function in orange, is mapped with a dotted line.

used to measure the element-wise reconstruction error.

$$\begin{aligned} \mathcal{L}_{\text{recon}} &= L1\_loss(\mathbf{x}, \mathbf{y}) \\ &= \sum_{i=1}^{J} |\mathbf{x}_i - \mathbf{y}_i| \end{aligned} \tag{4.6}$$

The training of the hybrid network with 3 different losses is not very stable. For the reconstruction loss, there is an absolute target and metric for learning. So this is a constrained optimization and directly corresponds to the error of the 3D pose. Hence it is desired to give high priority and keep this loss relatively stable and low. All the 3 loss terms are weighted to be summed to one final object term that is used to update the network. The weight coefficient λ_{recon} is kept constant and relatively higher than

the other two that are introduced next.

4.2.2 Latent Prior Loss

The proposed hybrid though named VAE-GAN, actually uses a β-VAE, it is not highlighted as the only difference is having a variable coefficient rather than fixing it to 1. Referring to the explanation of the β-VAE in (2.1.3), the higher the weight of the KLD term the better is the disentanglement or clustering of the embeddings are in the latent space. There is no magic β that gives the best reconstruction and representation. This is a trade-off and the value is chosen based on the requirements. Though both are the priorities of this work, more importance is given to the reconstruction as it is quantitative. It is important to note that the weight coefficient λ_{KLD} to balance the loss terms are separated for the β term to have better control when conducting experiments with different scheduling strategies for the β term. A cyclic beta technique similar to [10] is used to handle the vanishing KLD problem, where β is increased from 0 to true value every b epochs and observed to improve training.

$$\begin{aligned} \mathcal{L}_{\text{prior}} = \mathcal{L}_{\text{KLD}} &= -0.5 * \sum (1 + log(\sigma^2) - \mu^2 - \sigma^2) \\ &= -0.5 * \sum (1 + logvar - \mu^2 - e^{logvar}) \end{aligned} \tag{4.7}$$

The $logvar$ term here is the output of one of the downsampling layers of the encoder in practice. The terms denoted in the equation are per sample and in practice, the losses are averaged and normalized based on the dimensionality. The normalization is included in their λ coefficients.

4.2.3 Discriminator Loss

The discriminator takes both the samples from the real dataset $\mathbf{x}$ and predicted novel 2D poses $\tilde{\mathbf{y}}$ and classifies the samples. For making such training data the values of 1s and 0s are given as the labels for the real and generated 2D poses respectively. As the labels are self-generated the discriminator training is self-supervised as well. In contrast to the reconstruction error, the error in discriminator's classification is not expected to be minimized even though the models are learning to minimize this metric.

Rather, the role it plays is to help the generator produce 3D poses that give more realistic 2D novel views. That is 2D projections from random views of the 3D pose that are indistinguishable from the 2D pose from the real dataset. Fig 4.2.1 illustrates the differences in the novel views of realistic and unrealistic 3D poses that this adversarial training aims to resolve. Hence this error is used to optimize both the discriminator and generator models that produce the 3D poses.

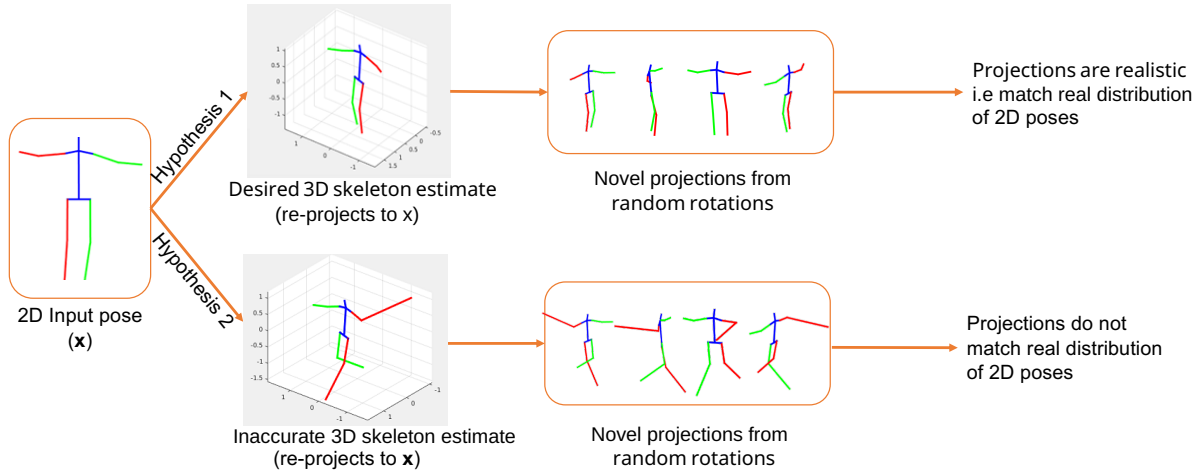


Figure 4.2.1: Illustration of the differences in accurate or realistic 3D pose predicitions from inaccurate predicitions. Modified image from [7]

The real and the fake 2D poses need not be related in any way. In practice, the loss terms of the real and fake data are computed and used to update the models separately, as suggested in [11, 13]. As the goal of the discriminator is to just predict the correct label of the 2D pose, the Binary Cross Entropy (BCE) loss function is used.

$$\begin{aligned} \mathcal{L}_{\text{Disc.}} &= BCE\_loss(y, \hat{y}) \\ &= -1 * (\hat{y} \cdot \log y + (1 - \hat{y}) \cdot \log(1 - y)) \\ &Where, \;\; y \in D_{\theta_d}(\tilde{\mathbf{y}} \cup \mathbf{x}) \;\; and \;\; \hat{y} \in (0s, 1s) \end{aligned} \tag{4.8}$$

The $\tilde{\mathbf{y}}$ here is the transformed output of the decoder. That means, to minimize this loss the decoder model is to be updated along with the discriminator. However, as discussed earlier we consider the encoder to be part of the generator. Hence all the models use this metric to update their weights. Again in contrast to the reconstruction loss, the discriminator loss does not give a direct signal to correct the 3D pose, and thus updating the models to minimize this loss is an unconstrained optimization and is not stable. Hence the weight coefficient $\lambda_{disc.}$ is not as high as that of the reconstruction

loss. As this is a min-max game where the generator tries to fool the discriminator while the discriminator tries to guess which same is from the generator, all the models cannot be updated at the same time. The training procedure to enable the models to learn and improve the generated 3D pose is discussed in the next section.

4.3 Training Procedure

In this min-max game, the generator produces samples and the discriminator verifies them, the generator uses this verification and in our method, it also uses self-supervision loss to improve itself in both aspects. The standard practice in training GANs is to start with generating the samples using the generator. As the discriminator's evaluation is required to teach the generator, the discriminator is first trained to distinguish reals from fakes given by the generator. The samples are again given to the trained discriminator to get the loss to train the generator. That is, the generator is kept constant while updating the discriminator and the discriminator is kept constant while updating the generator. "This turn-wise play is important without which the models would be trying to hit a moving target" [1].

While the procedure seems straightforward, GAN training is finding a Nash equilibrium to a two-player non-cooperative game. There exists no feasible algorithm to find the equilibrium for a problem of such complexity[33]. As the training initiates, the VAE produces 3D poses close to the noise and thus is very easy for the discriminator to learn to classify. This overpowers the discriminator as it predicts fakes with very high confidence and the loss tends to 0. The VAE can not learn properly if the discriminator does not provide proper feedback. This happens when the discriminator does not perform well or if it performs too well where the loss is close to 0 and not helpful [2].

Other common challenges of GAN training and is explained in (2.1.4). The widely practiced tricks to improve the training of GANs are discussed later in Bag of Tricks (5.3). The most challenging part of training any GANs is the lack of a proper metric to evaluate the quality of generations. Since it is difficult to monitor intermediate states in 3D, and since 3D poses have fewer features compared to RGB images, it is even more

[1] GAN training https://developers.google.com/machine-learning/gan/training

[2] From GAN to WGAN https://lilianweng.github.io/lil-log/2017/08/20/from-GAN-to-WGAN.html

challenging to evaluate the quality of the generations. Hence more importance is given to the quantitative aspect in this report.

4.4 Evaluation

4.4.1 Metric

3D HPE and Human3.6M in particular is evaluated by Mean Per Joint Position Estimate (MPJPE) metric. MPJPE as it abbreviates is the mean of the position estimate for all the joints of a pose. Where per-joint position estimate is nothing but the euclidian distance (measured in mm) between the predicted joint to its ground truth. The reported results refer to the average of the MPJPE of all the samples in the evaluation set.

4.4.2 Protocols

Human3.6M has 11 subjects out of which 7 are publically released while the rest are kept private. All the works main use of these 7 subjects for training and test. There are 2 widely used evaluation protocols.

Protocol #1 All the 4 camera views of subjects $S1$, $S5$, $S6$, $S7$, and $S8$ are used for training. Similarly, all camera views of subjects $S9$ and $S11$ are used for validation/testing. But the test subject data is complied by sampling every 64^{th} frame of the raw data while the train subjects data is made by sampling every 5^{th} frame. The evaluation metric MPJPE calculation is done on the predicted 3D pose directly.

Protocol #2 The train/test split is the same as of Protocol #1. But MPJPE is calculated on the predicted 3D after rigid body/ Procrustes alignment.

The design choice of predicting the 3D pose in unit scale makes it impossible to evaluate using protocol #1. Hence, following [5], results are reported using only protocol #2 i.e after aligning the predicted 3D pose with the ground truth.

4.4.3 Best Hypothesis

During the evaluation phase, the discriminator network is discarded and the encoder and decoder pair alone is used. The usual practice during the evaluation phase of a VAE is to compute latent representation z using mean and zeroing out the predicted standard deviation. The 3D pose prediction using the mean only is referred to **ZV**, Zero Variance. As the trained decoder can predict realistic 3D poses from any point in the latent space the variance can be scaled instead of zeroing to retrieve multiple 3D predictions for a 2D pose from which the mean is encoded. To evaluate the representation capability, 10 random hypotheses are produced following [25]. This is done by randomly scaling the predicted variance vector. The best hypothesis, **BH** i.e the pose with the least MPJPE for each 2D pose is selected to compute another variance of the mean MPJPE.

Chapter 5

Experiments

This chapter includes the various techniques and hacks explored to enhance the GAN training and reasoning behind the choices that are crucial for the functioning of the method.

5.1 Supervised Baseline

To the best of the author's knowledge, the closest work using a VAE is a cross-modal training approach in 3D hand pose estimation [38]. As there is no similar approach in HPE using a VAE, a supervised model using just a VAE and 3D ground truth pose data is implemented. This is to verify the feasibility of a VAE and find a set of hyperparameters that worked well for the H3.6M dataset under the supervised setting. This served as a baseline over which an unsupervised GAN was built upon. Hyperparameters such as the number of neurons in the layers, the number of residual blocks n, dimensions of the latent space, etc are taking as reference.

5.2 Unsupervised GAN

Building upon the supervised VAE, the proposed architecture is built. The following choice is kept standard for most of the experiments and could be used to reproduce the results. All the layers are made of 1024 neurons. The encoder and the decoder are made of 1 residual block each ($n = 1$) while the discriminator is made of 2. This is to keep

the capacity of the generator and the discriminator similar. Various forms of scaling the 2D poses are experimented with to make the cycle of lifting and projection smooth. The scaling mentioned in the preprocessing section where the distance between the root joint and head worked the best.

5.3 Bag of Tricks

The following techniques to improve the GAN training and the reasoning behind their importance are elaborated in papers such as [11, 13, 33, 46]. The important hacks or tricks from these papers and other sources are compiled in [37] and are wildly adopted in the field. Since the majority of the work in the field is on images, some of these tricks do not help and sometimes even hurt the training. These ticks along with the pre-processing steps mentioned in (3.4) are critical for proper training of the network. The tricks are as follows.

Normalizing Inputs It is common practice to normalize the inputs before training a neural network. When it comes to GANs, the output of the generator is the input of the discriminator. Hence it is suggested to normalize the images between [-1,1] by using a Tanh activation function at the output layer of the generator. This is another motivation behind predicting the poses in the range [-1,1] and then scaling the make the upper half of the pose to unit length.

Modified Loss function The theory the generator component of the GAN loss 2.5, $log(1 - D(G(z)))$ is minimized. As explained in (2.1.4), it is replaced with maximizing $log(D(G(z)))$ as the former leads to vanishing gradients early on during the training. This is referred to as the $-logD$ trick. In practice, the labels for fakes are flipped as reals while training the generator, since the goal is to make the generator's output real according to the discriminator. And as the BCE loss 4.8 which has a negative magnitude in its formulation is used, maximization is achieved while minimizing the loss during training. Note that the generator variable $G(z)$ in our training procedure is actually $P(Q(\mathbf{x}))$.

Spherical Latent Space Sample z from a spherical distribution instead of uniform distribution. Doing interpolation along the great circle 5.3.1 rather than a straight line from sample A to sample B. This spherical linear interpolation prevents the divergence of samples from the model's prior distribution and produces output with better features.

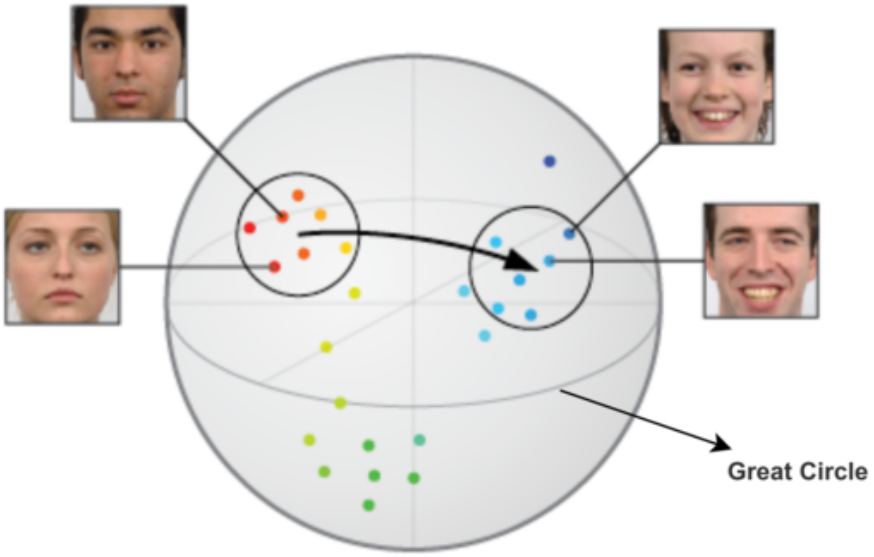


Figure 5.3.1: Illustration of the sampling from the great circle. Image source [47]

BatchNorm Training the GAN using separate batches of real and fake data without mixing them. And perform instance normalization when batch normalization is not possible.

Avoid Sparse Gradients Sparse gradients affect the training stability of the GAN. Hard functions like hard- Rectified Linear Unit (ReLU), Max Pooling that have sparse gradients should be avoided. Instead, Leaky-ReLU work well for both the generator and the discriminator. And average pooling or convolution layers with stride are better for downsampling or upsampling.

As mentioned previously Tanh and Sigmoid activations are used for the output layers of the generator and the discriminator respectively. For all the remaining layers

Leaky-ReLU was tried and good results were observed. Experimenting with other combinations of activation functions, using Mish activation function [29] for the VAE and Leaky-ReLU for the discriminator is found to give small but noticeable improvement to the performance.

Based on the choice of activation, Kaiming He initialization with Leaky-ReLU non-linearity is used for all the layers.

Label Smooting The label smoothing technique mentioned in [37] is by replacing the binary labels for real and fake i.e 1, 0 with random numbers, between 0.7 and 1.2 and 0.0 to 0.3 respectively. However, there are different ways of doing smoothing. These include random/non-random, one-sided/two-sided, only for discriminator/both networks. Only the real labels are smoothed for both the discriminator and the generator in the experiments.

Label Flipping The other way of making the labels noisy for the discriminator to prevent it from being too strong is by flipping labels during the discriminator training. Real labels are changed to fake and vice-versa occasionally to confuse the discriminator.

This had some effect during the initial phase of the training but is not observed to directly affect the quantitative results and hence and kept inactive for simplicity.

GAN Model Choice As training GANs is notoriously difficult, the most explored and studied variant DCGAN (Deep Convolutional GAN) is suggested to be used. The next best approach is to use hybrid models that use KLD or a combination of VAE and GAN. Since it is not desired to use convolutional layers in lifting networks, the proposed method is already following the best alternative.

Optimizer choice The best configuration of optimizers is Adam for the generator and SGD for the discriminator. However, this combination made the models diverge and has severely affected the training of the proposed method. Hence Adam is chosen for both the networks.

Noisy Inputs Adding noise to inputs and decaying over time helps the training. The 2D poses were added random noise in different scales but improvements were not noticeable in the basic experiments.

Training Discriminator More To consistently get good feedback from the discriminator, it is ideal to keep always keep the discriminator at good performance. To achieve this, the discriminator is iterated n times before every iteration of the VAE. However, n is set to 1 as the benefit could not be seen in the initial observations.

Dropout Adding additional noise to the generator using dropout with $p = 0.5$ during training and test time [18]. The discriminator uses dropout with $p = 0.5$. But it is observed that adding dropout layers hurt VAEs and $p = 0.2$ was found to have a good trade-off.

Chapter 6

Results

The results presented here are after training the networks for ∼400 epochs (5.5 hours) on approximately 300,000 2D poses with a batch size of 2560 on an Nvidia Titan X. The input poses are flipped with a probability of 0.5. The model takes 16 joints as the output where the root is added at the origin for validation. The proposed architecture consists of 1024 hidden units per linear layer and 51 latent dimensions. Both the VAE and the discriminator are trained using Adam optimizer with default hyperparameters and with a learning rate of 2e-4. The gradient norms of the discriminator are clipped to 1 when training the discriminator. While training the generator the gradient norms are clipped to 2 for all the models while the gradient values are clipped to 1000.

One of the challenging parts is finding the optimal weights for each of the terms in the triplet loss. The loss coefficients λ_{recon}, λ_{KLD}, $\lambda_{disc.}$ are set to 1, 0.001, 0.001 respectively. The higher weight is motivated by 2 reasons. λ_{recon} refers to the constrained optimization and irrespective of how realistic it is, projection loss is desired to be consistently low to get better MPJPE. That leads to the other reason that the quantitative results are given higher importance.

The values of λKLD and $\lambda disc.$ can be tuned according to the task at hand based on how well the poses are to be clustered or how important it is to reject poses that are not realistic. The β value for the VAE is cycled from 0 to λ_{KLD} every 40 epochs. While keeping it constant at λ_{KLD} for 10 epochs with a 10 epoch warmup at the beginning of the training.

6.1 Quantitative Results

The results obtained by the networks with the above configuration in addition to the choices mentioned in 5 are summarized in Table 6.1.1. The summaries of the models are provided in appendix A to help reproduction.

Supervision	Algorithm	Error (mm)
Full	Martinez *et al.* [27]	37.1
	Chen *et al.* [24] (SH, MH)	42.6
Weak	3D Interpreter *et al.* [48]	88.6
	AIGN *et al.* [39]	79.0
	Wandt *et al.* [43]	38.2
	Drover *et al.* [7]	38.2
	Chen *et al.* [25] (SH)	48.7
	Chen *et al.* [25] (BH)	31.6
Unsupervised	Ching *et al.* [5]	58
	Ching *et al.* [5] (DA)	55
	Ching *et al.* [5] (DA) (TD)	51
	Ours (ZV)	52.74
	Ours (BH)	50.37

Table 6.1.1: Results on Human3.6M in MPJPE under Protocol #2 using ground truth 2D pose as input.

SH refers to the results using 2D Stacked Hourglass detections as input. These detections are noisy and directly affects the predictions of the model. And **DA** refers to using Domain Adaptation network to include additional datasets and **TD** denotes the use of temporal data. It is important to note that the proposed unsupervised method and the one presented in [5] are the only methods that do not predict the scale of the 3D pose due to the processing technique.

R_Hip	R_Knee	R_Ankle	L_Hip	L_Knee	L_Ankle	Torso	Neck
58.01	61.42	81.97	52.45	60.13	92.82	44.78	25.43
Nose	Head	L_Shoulder	L_Elbow	L_Wrist	R_Shoulder	R_Elbow	R_Wrist
33.39	46.29	30.48	55.72	80.84	33.53	59.23	80.05

Table 6.1.2: Average per joint position error (in mm) for each **joint** under Protocol #2 using 2D ground truth.

The best hypothesis, **BH** has improved the results by 2.7 mm which is considerable in comparison to the equivalent gain in [5] using domain adaptation network with more data and temporal information. Since there is no technique to pick the best hypothesis

Directions	Discussion	Eating	Greeting	Phoning	Photo	Posing	Purchases
44.37	45.14	56.64	48.15	56.47	45.06	47.33	71.30
Sitting	SittingDown	Smoking	Waiting	WalkDog	Walking	WalkTogether	
72.08	55.01	52.47	47.37	47.51	49.53	44.61	

Table 6.1.3: Average MPJPE (in mm) for each **action** under Protocol #2 using 2D ground truth.

without having access to the ground truth, the results referred to, are the ones obtained using **ZV** unless specified otherwise.

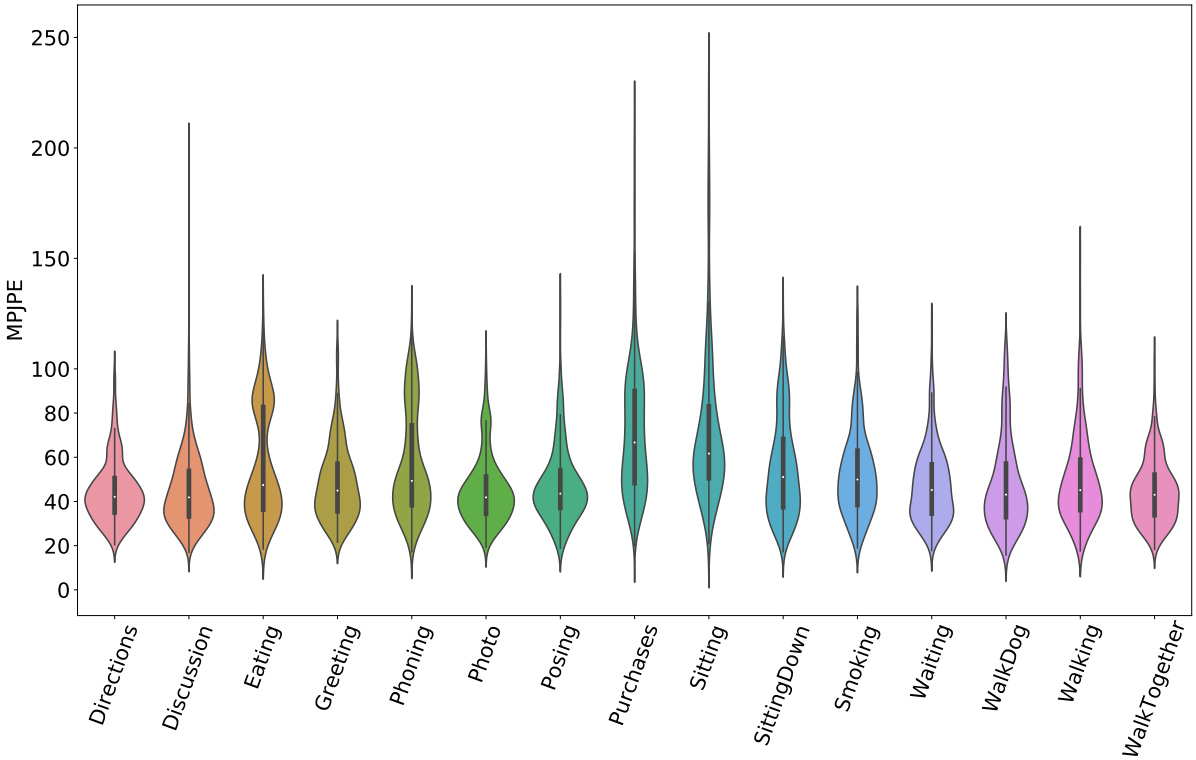


Figure 6.1.1: Visualization of the MPJPE error distribution under Protocol #2 for each action.

The average MPJPE errors for each action is presented in Table 6.1.3, and Fig 6.1.1 illustrates the distribution of these errors. The MPJPE error per joint increases as we move from inner to outer joints. This can be observed in Table 6.1.2 which shows the average per joint position error. The error for the Pelvis joint is not mentioned as it is assumed to be at the origin and is always zero. The larger errors in the limb joints are due to the higher variation in their location throughout the data compared to joints like the neck or the nose. Moreover, these joints are often occluded by the person's body especially when doing actions such as sitting. The 2D projections of a pose from sitting posture is clustered to a smaller region from certain POVs. This makes it challenging

to estimate the scale and depth of each joint and leads to outliers as visualized in the violin plots.

Since the data consists of an equal mix of data from different POVs, the predictions from the POV, where the limbs are in the Field of View (FOV) are more informative than the other POVs. When the limbs are not in the FOV, the model learns to guess their location based on other features such as articulation or posture of the body. The effect of POV is the possible cause of the bimodal error distribution in actions such as purchases, eating, phoning. Since the actions are hands specific, they are restricted to a particular region making the pose more predictable from one POV and less from another. These challenges are inherent to the task of pose lifting and in many cases are challenging even for a human eye.

6.2 Qualitative Results

6.2.1 Average Cases

Since pose lifting is a tough task for humans as well, the errors around and below 50mm are usually not identifiable without 3D ground truth reference. To give a better perspective and sense of depth, 3D models are used to visualize the model predictions. Instead of handpicking predictions, poses that have errors closest to the mean of the respective action are selected and visualized in Fig 6.2.1. The maximum mean MPJPE per action as presented in table 6.1.3, is not more than 75 mm and thus the predictions (in blue) are not very different from the ground truth pose (in pink). The limb joints especially the elbows and the wrist are occluded by the body in the majority of the visualized poses. Consider the 2D poses where the subject is seen in a side view, ignoring the RGB background it is quite ambiguous to say which limb is forward and which is back. The network has guessed the location of these joints to a good extent. The postures in the actions such as sitting down, sitting on a chair, etc have been predicted well.

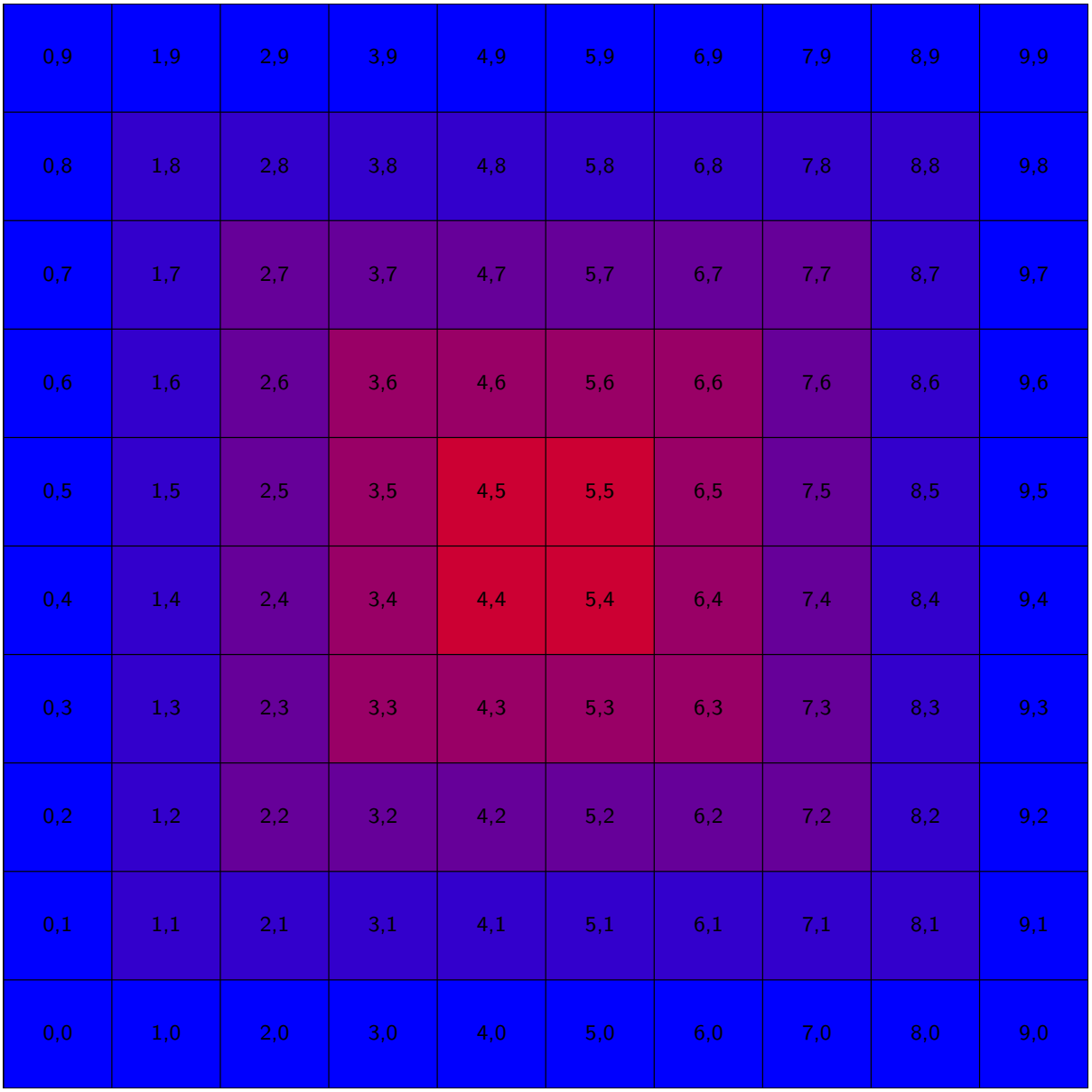


Figure 6.2.1: Visualization of (left to right) the input 2D poses with RGB background for reference, the predicted 3D pose after Procrustes alignment (in blue) and the ground truth 3D pose (in pink). The aligned predictions are the ones closest to the **mean** MPJPE error of each action.

6.2.2 Failure Cases

Similarly, the poses with the maximum errors (outliers) are taken for each action and visualized in Fig 6.2.2. The main reason as mentioned earlier is the failure in estimating the depth or in predicting a good representation of the 2D pose for the decoder. One of the best examples of ambiguity can be seen in the visualizations where the 3D pose is very plausible, with very good 2D reprojection but with the subject facing the wrong direction. It can be observed from some of the visualizations in 6.2.2, that the predicted

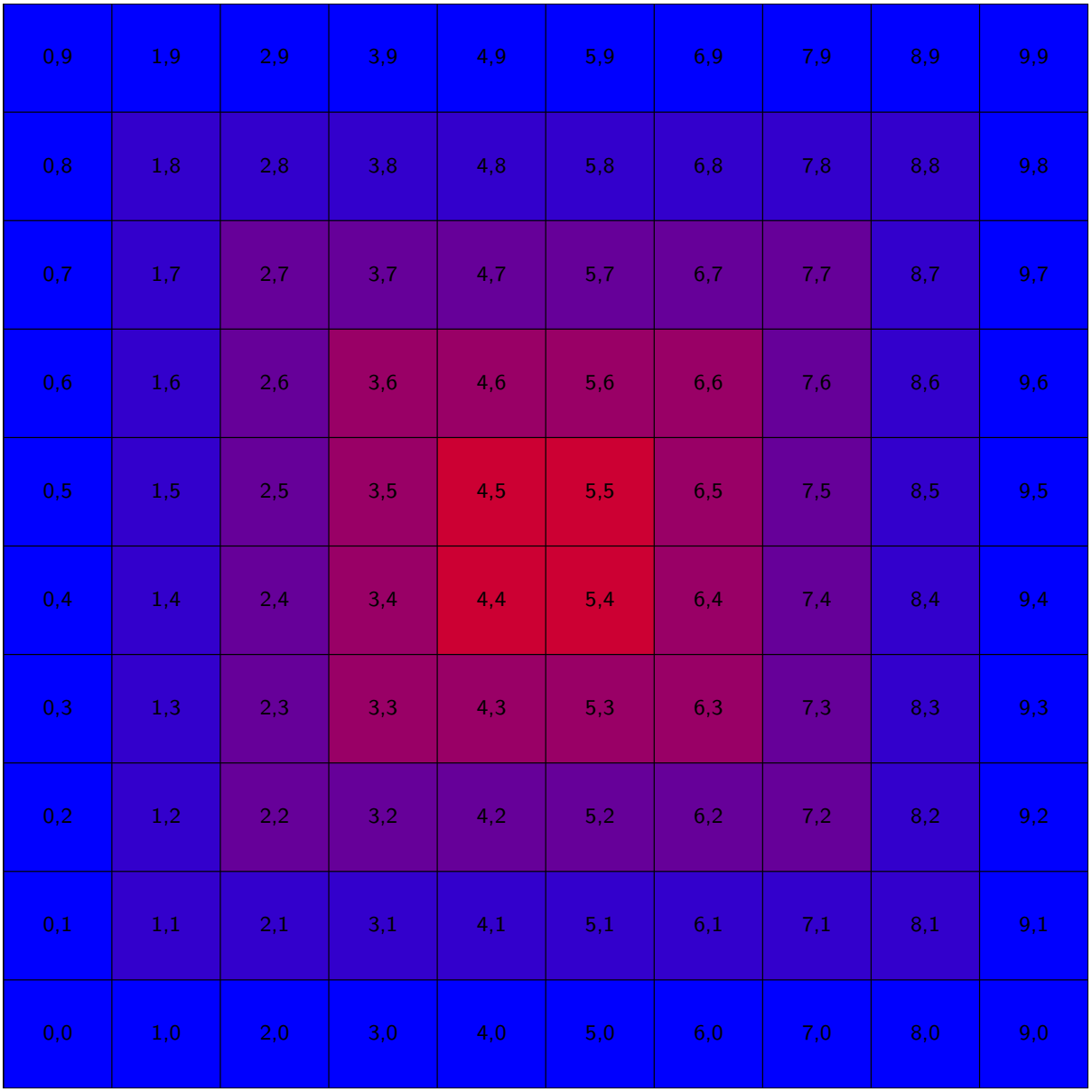


Figure 6.2.2: Visualization of (left to right) the input 2D poses with RGB background for reference, the predicted 3D pose after Procrustes alignment (in blue) and the ground truth 3D pose (in pink). The aligned predictions are the ones closest to the **maximum** MPJPE error of each action.

joints are consistent with respect to each other but are away from the root joint. Note that the root joint is always added to the prediction at the origin. Since the rest of the pose is consistent it could be implied that the model failed to predict the depth of the joints properly. Consider the model predicts a 3D pose whose 2D projection is deviated from the input 2D. This error is magnified in 3D if the joint is farther away from the image. To overcome such errors, higher weight is given to the 2D reconstruction loss.

6.3 Latent Space

A good representation of the 2D poses in the latent space not only helps the decoder to construct better 3D poses but also shows the models understanding of the pose in terms of the articulation/posture, and the POV. Though the ZV model used to report these results is not tuned to have the best clustering, it still should learn an efficient representation to encode the 2D poses in the latent space. In order to evaluate the encoder capabilities, the test dataset consisting of ≈ 8000 2D poses are encoded to 51-dimensional latent space. These embeddings z are further reduced to two dimensions using Uniform Manifold Approximation and Projection (UMAP) [28]. The clustering is computed using 50 neighbors and 0.1 units of minimum Euclidean distance and the reduced embeddings are visualized in Fig 6.3.1.

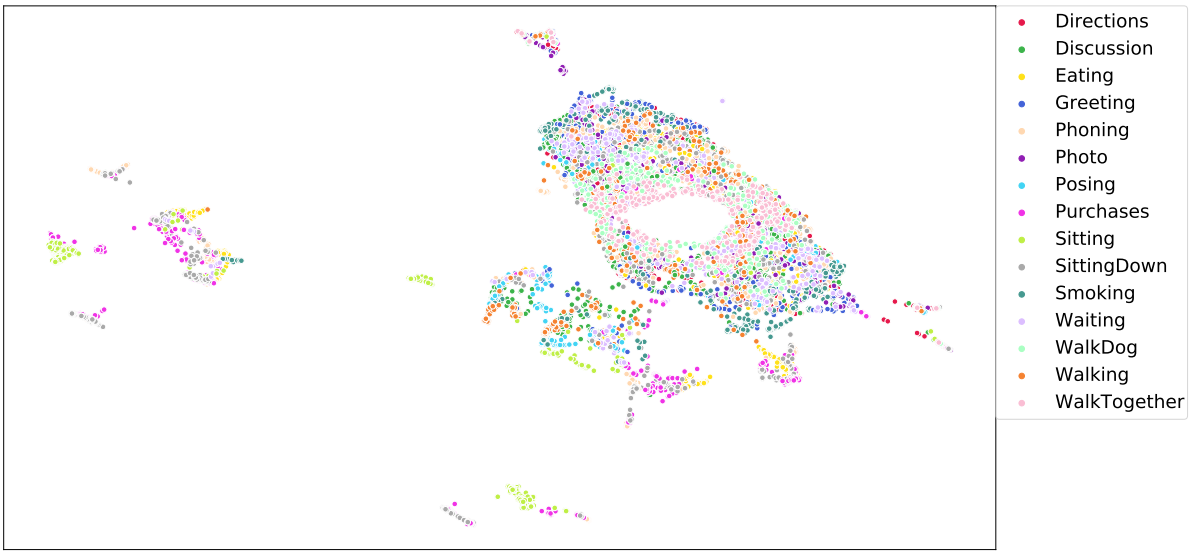


Figure 6.3.1: Visualization of the 2D pose embeddings in the latent space using UMAP dimensionality reduction.

The reduced z are colored with respect to their action. This is not an ideal way to examine the clustering as there is significant overlap in the actions during their 'transition phase'. For example, as the data is collected as the subject performs that action, the subject would be 'standing' before 'sitting down'. Since it is not practical to label each frame, the action labels are used. Despite the overlap, patterns and clusters can be seen indicating a good representation capability of the encoder. To further evaluate the extent of clustering, 5 nearest latent neighbors of the embeddings of a few interesting poses are queried to verify if they represent common distribution. The corresponding 2D poses of these embeddings are visualized in Fig 6.3.2. Strong

similarity can be observed among the nearest neighbors. Similarities include a set of joints having the same articulation, natural transition from one posture to another, or a different POV of the similar pose.

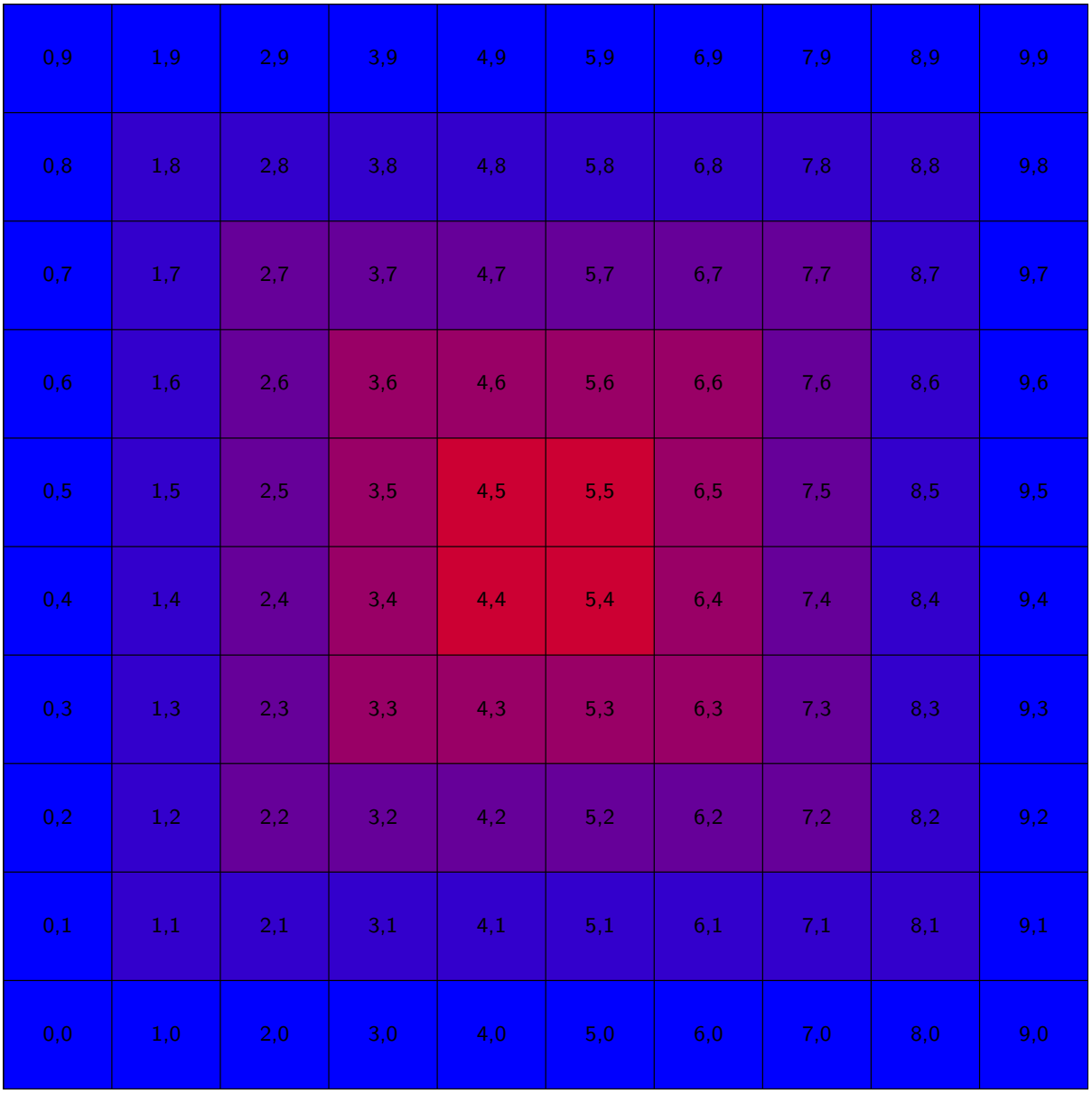


Figure 6.3.2: The first pose in box is the query 2D pose and the following 5 2D poses are its nearest neighbors in the latent space.

6.4 Missing Joints

The other important problem the proposed method addresses is the ability to handle missing points. The intuition is from the ability of the encoder to represent similar poses together in the latent space as seen in 6.3.2. A 2D pose missing a joint or two

should still be able to tell about the posture and the view of the subject. The encoding of such incomplete poses should be closer to the means of similar pose distributions. Given the decoder can produce plausible 3D poses, lifting the incomplete 3D pose should be at least plausible if not similar to the ground truth 3D pose. Hence the network should be able to 'hallucinate' the missing joints. To evaluate this, the model is given incomplete 2D poses obtained by randomly zeroing 1 and 2 joints. The models are not trained explicitly on any incomplete poses in order to verify the above hypothesis. The MPJPE using Zero Variance (ZV) and Best Hypothesis (BH) are reported in table 6.4.1. As expected, there is a significant difference between ZV and BH as there would be many possible locations for the missing joints among which the ground truth pose would be one.

# Missing Joints	Zero Variance (ZV)	Best Hypothesis (BH)
0	52.74	50.3
1	79	53
2	101	65

Table 6.4.1: Results on Human3.6M in MPJPE (in mm) under Protocol #2 using ground truth 2D pose with **missing joints** as input.

Chapter 7

Conclusions

7.1 Discussion

Previous unsupervised learning approaches for pose lifting such as [5] uses deep neural networks and geometric consistency to achieve results comparable to major weakly supervised approaches such as [7]. They address critical problems such as making use of datasets from different distribution using additional domain adaptation networks and exploiting temporal information. These techniques made it possible to use large amounts of 2D data to learn 3D poses. However, as the authors acknowledge the SOTA 2D pose networks often get few joints very wrong or miss them due to occlusions that are quite frequent in the real-world. Hence methods that are independent of the correctness of the input 2D are essential. [25, 31] are such weakly supervised methods that can handle missing joints while [25] can also give multiple hypotheses for a given 2D pose. The proposed method tackles these problems with a unified approach by using VAEs and GAN and is completely unsupervised.

The average MPJPE using ZV, 52.74mm, is equivalent to the settings of [5] without additional data or temporal information, 58 mm. This is a significant improvement considering that the network is independent of the input and can handle errors including missing joints, can produce multiple hypotheses. While [5] uses a self-symmetry technique that requires propagating through the models twice per iteration and requires 2 more additional networks of complexity similar to the lifting network to slightly exceed the performance of the proposed network.

Though the proposed network address all the 3 major challenges of lifting networks, i.e 3D annotations, missing joints, and variational inference, the network only predicts a unit 3D pose. This restriction limits the usability of the 3D pose in some cases or requires additional algorithms to obtain the true scale in the real-world. For example, the scale of the 3D pose is not important if the task is to use the method as a vision-based MoCap solution for gaming where only the articulation of the pose is of interest to capture complex human motion. While tasks in the domain of self-driving cars, robot-human interaction, or AR/VR applications where the pose is used for interaction among the agents, 3D pose alone would not be sufficient.

7.2 Future Work

The source of the major drawback of not predicting to scale is in the processing technique to make the self-supervision simple. This problem could be addressed by extending the method to disentangle the root-relative pose prediction and global pose prediction similar to [30].

VAEs have more to offer than just predict 3D while handling erroneous inputs. Taking the inspiration from [38], the method can be upgraded from a lifting network to end to end network to predict 3D pose from Images just by swapping the encoder model. Techniques like Synergy and Cross-Generation from cross-modal training [35] could be leveraged to train an unsupervised Image to 3D model faster and efficiently with the help of large 2D pose datasets.

In addition to this guaranteed improvised techniques such as using external datasets and temporal information to learn predicting temporally consistent 3D pose as presented in [5] would make the network more practical to use in the real world.

7.3 Final Words

Scaling the task of 3D HPE to the real-world is limited by the need for 3D data. This thesis presented an unsupervised learning method that learns to predict 3D pose without a need for 3D annotations in any shape or form. The method learns to lift 2D poses to 3D strictly using 2D pose data that are obtained from 2D pose estimation

models. The thesis introduced standards VAEs to the task of pose lifting for the first time. This generative model is trained using GAN and self-supervision techniques. The ability of the proposed method to handle incomplete and erroneous data without explicit training is shown. The thesis also shows the capacity of the model to learn a strong representation of the pose that could be used for other applications such as human-centric multi-view video synchronization. This thesis has considerably improved the SOTA in unsupervised 3D HPE and lays the foundation for new ways to tackle the problem.

Bibliography

[1] Arjovsky, Martin and Bottou, Léon. *Towards Principled Methods for Training Generative Adversarial Networks*. 2017. arXiv: 1701.04862 [stat.ML].

[2] Ballard, Dana H. "Modular Learning in Neural Networks." In: *AAAI*. 1987, pp. 279–284.

[3] Blei, David M, Kucukelbir, Alp, and McAuliffe, Jon D. "Variational inference: A review for statisticians". In: *Journal of the American statistical Association* 112.518 (2017), pp. 859–877.

[4] Chang, Ju Yong, Moon, Gyeongsik, and Lee, Kyoung Mu. "AbsPoseLifter: Absolute 3D Human Pose Lifting Network from a Single Noisy 2D Human Pose". In: *arXiv preprint arXiv:1910.12029* (2019).

[5] Chen, Ching-Hang, Tyagi, Ambrish, Agrawal, Amit, Drover, Dylan, Stojanov, Stefan, and Rehg, James M. "Unsupervised 3d pose estimation with geometric self-supervision". In: (2019), pp. 5714–5724.

[6] Cheng, Yu, Yang, Bo, Wang, Bo, Yan, Wending, and Tan, Robby T. "Occlusion-aware networks for 3d human pose estimation in video". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 723–732.

[7] Drover, Dylan, Chen, Ching-Hang, Agrawal, Amit, Tyagi, Ambrish, and Phuoc Huynh, Cong. *Can 3d pose be learned from 2d projections alone?* 2018.

[8] Fabbri, Matteo, Lanzi, Fabio, Calderara, Simone, Palazzi, Andrea, Vezzani, Roberto, and Cucchiara, Rita. "Learning to Detect and Track Visible and Occluded Body Joints in a Virtual World". In: *European Conference on Computer Vision*. 2018.

[9] Fan, Jianqing, Ma, Cong, and Zhong, Yiqiao. *A Selective Overview of Deep Learning*. 2019. arXiv: 1904.05526 [stat.ML].

[10] Fu, Hao. “Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing”. In: *NAACL*. 2019.

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

[12] Gu, Jiajun, Wang, Zhiyong, Ouyang, Wanli, Li, Jiafeng, Zhuo, Li, et al. “3D Hand Pose Estimation with Disentangled Cross-Modal Latent Space”. In: *The IEEE Winter Conference on Applications of Computer Vision*. 2020, pp. 391–400.

[13] Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent, and Courville, Aaron C. *Improved training of wasserstein gans*. 2017.

[14] Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent, and Courville, Aaron C. *Improved training of wasserstein gans*. 2017.

[15] Hanbyul Joo, Hao Liu. “Panoptic Studio: A Massively Multiview System for Social Motion Capture”. In: (2015).

[16] Hinton, Geoffrey E and Salakhutdinov, Ruslan R. “Reducing the dimensionality of data with neural networks”. In: *science* 313.5786 (2006), pp. 504–507.

[17] Ionescu, C., Papava, D., Olaru, V., and Sminchisescu, C. “Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2014), pp. 1325–1339.

[18] Isola, Phillip, Zhu, Jun-Yan, Zhou, Tinghui, and Efros, Alexei A. *Image-to-image translation with conditional adversarial networks*. 2017.

[19] Jahangiri, Ehsan and Yuille, Alan L. *Generating multiple diverse hypotheses for human 3d pose consistent with 2d joint detections*. 2017.

[20] Kingma, Diederik P and Welling, Max. *Auto-Encoding Variational Bayes*. 2013.

[21] Kong, Chen and Lucey, Simon. “Deep non-rigid structure from motion”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 1558–1567.

[22] Kudo, Yasunori, Ogaki, Keisuke, Matsui, Yusuke, and Odagiri, Yuri. “Unsupervised adversarial learning of 3d human pose from 2d joint locations”. In: *arXiv preprint arXiv:1803.08244* (2018).

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

[24] Li, Chen and Lee, Gim Hee. *Generating multiple hypotheses for 3d human pose estimation with mixture density network*. 2019.

[25] Li, Chen and Lee, Gim Hee. *Weakly Supervised Generative Network for Multiple 3D Human Pose Hypotheses*. 2020. arXiv: 2008.05770 [cs.CV].

[26] Marcard, Timo von, Henschel, Roberto, Black, Michael J, Rosenhahn, Bodo, and Pons-Moll, Gerard. "Recovering accurate 3D human pose in the wild using imus and a moving camera". In: *Proceedings of the European Conference on Computer Vision*. 2018, pp. 601–617.

[27] Martinez, Julieta, Hossain, Rayat, Romero, Javier, and Little, James J. "A simple yet effective baseline for 3d human pose estimation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2640–2649.

[28] McInnes, Leland, Healy, John, and Melville, James. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML].

[29] Misra, Diganta. *Mish: A self regularized non-monotonic neural activation function*. 2019.

[30] Moon, Gyeongsik, Chang, Ju Yong, and Lee, Kyoung Mu. "Camera distance-aware top-down approach for 3d multi-person pose estimation from a single rgb image". In: (2019), pp. 10133–10142.

[31] Novotny, David, Ravi, Nikhila, Graham, Benjamin, Neverova, Natalia, and Vedaldi, Andrea. "C3DPO: Canonical 3d pose networks for non-rigid structure from motion". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7688–7697.

[32] Rumelhart, David E, Hinton, Geoffrey E, and Williams, Ronald J. *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[33] Salimans, Tim, Goodfellow, Ian, Zaremba, Wojciech, Cheung, Vicki, Radford, Alec, and Chen, Xi. *Improved techniques for training gans*. 2016.

[34] Sharma, Saurabh, Varigonda, Pavan Teja, Bindal, Prashast, Sharma, Abhishek, and Jain, Arjun. *Monocular 3d human pose estimation by generation and ordinal ranking*. 2019.

[35] Shi, Yuge, Paige, Brooks, Torr, Philip, et al. "Variational mixture-of-experts autoencoders for multi-modal deep generative models". In: *Advances in Neural Information Processing Systems* 32 (2019), pp. 15718–15729.

[36] Sigal, Leonid, Balan, Alexandru O, and Black, Michael J. *Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion*. 2010.

[37] Soumith, Chintala, Denton, Emily, Arjovsky, Martin, and Mathieu, Michael. *How to Train a GAN? Tips and tricks to make GANs work*. 2016.

[38] Spurr, Adrian, Song, Jie, Park, Seonwook, and Hilliges, Otmar. "Cross-modal deep variational hand pose estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 89–98.

[39] Tung, Hsiao-Yu Fish, Harley, Adam W, Seto, William, and Fragkiadaki, Katerina. *Adversarial inverse graphics networks: Learning 2d-to-3d lifting and image-to-image translation from unpaired supervision*. IEEE, 2017.

[40] Ulyanov, Dmitry, Vedaldi, Andrea, and Lempitsky, Victor. "Deep image prior". In: (2018), pp. 9446–9454.

[41] Vincent, Pascal, Larochelle, Hugo, Bengio, Yoshua, and Manzagol, Pierre-Antoine. "Extracting and composing robust features with denoising autoencoders". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 1096–1103.

[42] Wan, Chengde, Probst, Thomas, Van Gool, Luc, and Yao, Angela. "Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 680–689.

[43] Wandt, Bastian and Rosenhahn, Bodo. "Repnet: Weakly supervised training of an adversarial reprojection network for 3d human pose estimation". In: (2019), pp. 7782–7791.

[44] Wang, Chaoyang, Kong, Chen, and Lucey, Simon. “Distill knowledge from nrsfm for weakly supervised 3d pose learning”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 743–752.

[45] Wang, Chaoyang, Lin, Chen-Hsuan, and Lucey, Simon. “Deep nrsfm++: Towards 3d reconstruction in the wild”. In: *arXiv preprint arXiv:2001.10090* (2020).

[46] Weng, Lilian. *From GAN to WGAN*. 2019.

[47] White, Tom. *Sampling Generative Networks*. 2016.

[48] Wu, Jiajun, Xue, Tianfan, Lim, Joseph J, Tian, Yuandong, Tenenbaum, Joshua B, Torralba, Antonio, and Freeman, William T. “Single image 3D interpreter network”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 365–382.

[49] Xie, Junyuan, Xu, Linli, and Chen, Enhong. “Image denoising and inpainting with deep neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 341–349.

[50] Ye, Qi and Kim, Tae-Kyun. *Occlusion-aware hand pose estimation using hierarchical mixture density network*. 2018.

[51] Zhang, Richard, Isola, Phillip, and Efros, Alexei A. “Colorful image colorization”. In: *European conference on computer vision*. Springer. 2016, pp. 649–666.

Appendix - Contents

Appendix A

Model Summaries

A.1 Encoder

	Layers	Parameters
Upsampling Block		
	Linear(in features=32, out features=1024, bias=True)	33792
	BatchNorm1d(1024, eps=1e-05, momentum=0.1)	2048
	Mish()	0
	Dropout(p=0.2, inplace=False)	0
Residual Block		
	Linear(in features=1024, out features=1024, bias=False)	1048576
	BatchNorm1d(1024, eps=1e-05, momentum=0.1)	2048
	Mish()	0
	Dropout(p=0.2, inplace=False)	0
	Linear(in features=1024, out features=1024, bias=False)	1048576
	BatchNorm1d(1024, eps=1e-05, momentum=0.1)	2048
	Mish()	0
	Dropout(p=0.2, inplace=False)	0
Downsampling Block		
	Linear(in features=1024, out features=51, bias=True)	52275
	Linear(in features=1024, out features=51, bias=True)	52275

A.2 Decoder

	Layers	Parameters
Upsampling Block		
	Linear(in features=51, out features=1024, bias=True)	53248
	BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track running stats=True)	2048
	Mish()	0
	Dropout(p=0.2, inplace=False)	0
Residual Block		
	Linear(in features=1024, out features=1024, bias=False)	1048576
	BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track running stats=True)	2048
	Mish()	0
	Dropout(p=0.2, inplace=False)	0
	Linear(in features=1024, out features=1024, bias=False)	1048576
	BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track running stats=True)	2048
	Mish()	0
	Dropout(p=0.2, inplace=False)	0
	Linear(in features=1024, out features=48, bias=True)	49200
Downsampling Block		
	Linear(in features=1024, out features=48, bias=True)	49200
	Tanh()	0

A.3 Discriminator

	Layers	Parameters
Upsampling Block		
	Linear(in features=32, out features=1024, bias=True)	33792
	LeakyReLU(negative slope=0.01)	0
Residual Block		
	Linear(in features=1024, out features=1024, bias=True)	1049600
	LeakyReLU(negative slope=0.01)	0
	Dropout(p=0.5, inplace=False)	0
	Linear(in features=1024, out features=1024, bias=True)	1049600
	LeakyReLU(negative slope=0.01)	0
	Dropout(p=0.5, inplace=False)	0
Residual Block		
	Linear(in features=1024, out features=1024, bias=True)	1049600
	LeakyReLU(negative slope=0.01)	0
	Dropout(p=0.5, inplace=False)	0
	Linear(in features=1024, out features=1024, bias=True)	1049600
	LeakyReLU(negative slope=0.01)	0
	Dropout(p=0.5, inplace=False)	0
Upsampling Block		
	Linear(in features=1024, out features=1, bias=True)	1025
	Sigmoid()	0