Multimodal Representation Learning using Variational Autoencoders

Naman Gupta

BT/CSE/190527

Supervisor: Prof. Ashutosh Modi

November 25, 2021

Abstract

Learning multimodal representations involves integrating information from multiple heterogeneous sources of data. It is a challenging yet crucial area with numerous real-world applications in multimedia, affective computing, robotics, finance, human-computer interaction, and healthcare. In this project, we explore one such real-world case through 3D Hand Pose Estimation using monocular RGB images. Variational autoencoders provide a principled framework for learning deep latent-variable models and corresponding inference models. Hence, we particularly focus on variational techniques which make use of auxiliary modalities as weak labels during training by aligning their corresponding latent spaces and learning a joint representation while using only a single modality during inference. Our method generalizes across domains, being weakly supervised it is robust to missing modalities during training. We also explore standard modality mixing techniques, propose new methods and benchmark our results against the current state of the art in 3D Handpose Estimation.

Acknowledgments

I am deeply indebted to Prof Ashutosh Modi for allowing me to work with him. I am thankful to him for having regular meetings and helping me throughout the project by providing the much needed insights and encouragement. I would also like to thank Dr. Binod Bhattarai, University College London who let me part of the ongoing research work with Prof. Modi, this project is a part of. The regular interactions with Dr. Bhattarai and Prof. Modi helped me to channelize my efforts in a fruitful direction. I also take this moment to thank Mr. Abhinav Joshi and Mr. Jinang Shah, who were part of this research and actively contributed in the same. Last but not the least, I thank my family for their support and love during this pandemic.

Contents

Chapter 1

Introduction

Modality refers to how things are experienced in terms of sensory inputs or how information is represented and communicated. Language, vision, audio, physiological signals, physical sensor signals, etc., are different forms of modalities used to understand the world around us. Multimodal data helps us describe the objects or phenomena using different aspects or viewpoints with complementary or supplementary information. Multimodal Machine Learning is the process of learning predictive models from multimodal data. For example, a simple multimodal classifier assigns a label to an input which consists of different modalities.

There are some obvious challenges with multimodal learning though, such as how to fuse multimodal representations to a joint representation. The aim of this project is to answer such questions using a challenging real-world problem – 3D HandPose Estimation.

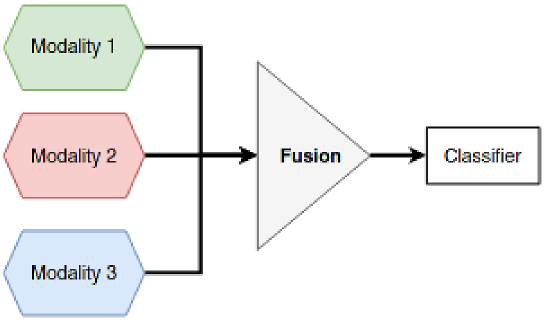


Figure 1.1: A simple multimodal classifier

The higher performance of multimodal models depends on the availability of aligned, noiseless and annotated modalities at training and testing. However, all modalities may not be available at all times; those may be noisy and may be in a limited amount. Hence, we approach the problem as a Weakly Supervised Learning problem where the model is not affected if some of the modalities are missing. The techniques mentioned in this report can easily be applied to other datasets since the core ideas of multimodal fusion of latent representations remain the same.

Chapter 2

Preliminaries

2.1 Variational Autoencoders

One major division in machine learning is generative versus discriminative modeling. While in discriminative modeling one aims to learn a predictor given the observations, in generative modeling one aims to solve the more general problem of learning a joint distribution over all the variables. While the classifier mentioned earlier is a discriminative model where the model learns to predict a label, a variational autoencoder is a generative model where the model learns the distribution of the data itself. We explain this formally below –

Let $\mathbf{x}$ be the vector representing the set of all observed variables whose joint distribution we would like to model. We assume the observed variable $\mathbf{x}$ is a random sample from an *unknown underlying process*, whose true distribution $p^*(\mathbf{x})$ is unknown. We attempt to approximate this underlying process with a chosen model $p_\theta(\mathbf{x})$, with parameters θ. Generative modeling can be seen as the process of searching θ such that $p_\theta(\mathbf{x}) \approx p^*(\mathbf{x})$. A discriminative model on the other hand learns a θ such that $p_\theta(\mathbf{y}|\mathbf{x}) \approx p^*(\mathbf{y}|\mathbf{x})$ where $\mathbf{y}$ is the *label* corresponding to $\mathbf{x}$. The parameters θ can be estimated using Bayesian Inference techniques – *maximum likelihood* or *maximum a posteriori* estimates.

Latent Variables are variables that are part of the model, but not part of the dataset since we don't observe them. Let $\mathbf{z}$ denote the latent variable, in this case the model would represent the joint distribution $p_\theta(\mathbf{x}, \mathbf{z})$ where the marginal distribution $p_\theta(\mathbf{x})$ is given as:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})\, d\mathbf{z}$$

The marginal distribution $p_\theta(\mathbf{x})$ is often quite complex (intractable) due to the inherent nature of the data. This is due to the integral not having an analytical solution. Due to this intractability, we cannot differentiate it with respect to the parameters θ, optimize it to find θ_{MLE}. The posterior

$$p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x}, \mathbf{z})}{p_\theta(\mathbf{x})}$$

is also intractable since it depends on the marginal.

The main idea behind VAEs is approximating the posterior using an encoder $q_\phi(\mathbf{z}|\mathbf{x})$ where ϕ represents the parameters of this encoder model. We optimize ϕ such that $q_\phi(\mathbf{z}|\mathbf{x}) \approx p_\theta(\mathbf{z}|\mathbf{x})$. This approximation helps us to in turn optimize the marginal likelihood $p_\theta(\mathbf{x})$.
The log-likelihood is given as –

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] && (2.1)\\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})}\right]\right] && (2.2)\\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})}\right]\right] && (2.3)\\
&= \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right]\right]}_{\substack{=\mathcal{L}_{\theta,\phi}(\mathbf{x})\\ \text{(ELBO)}}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\left[\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})}\right]\right]}_{=D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))} && (2.4)
\end{aligned}$$

Since we approximate the posterior using another distribution, we enforce this by minimizing the KL Divergence $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$. This is same as maximizing $\mathcal{L}_{\theta,\phi}(\mathbf{x})$. Further, since D_{KL} is non negative, $\mathcal{L}_{\theta,\phi}(\mathbf{x})$ is the lower bound on the log likelihood. Hence, it is known as *Evidence Lower Bound.*

Figure 2.1 represents a typical VAE. The distribution $p(\mathbf{x})$ is complex hence

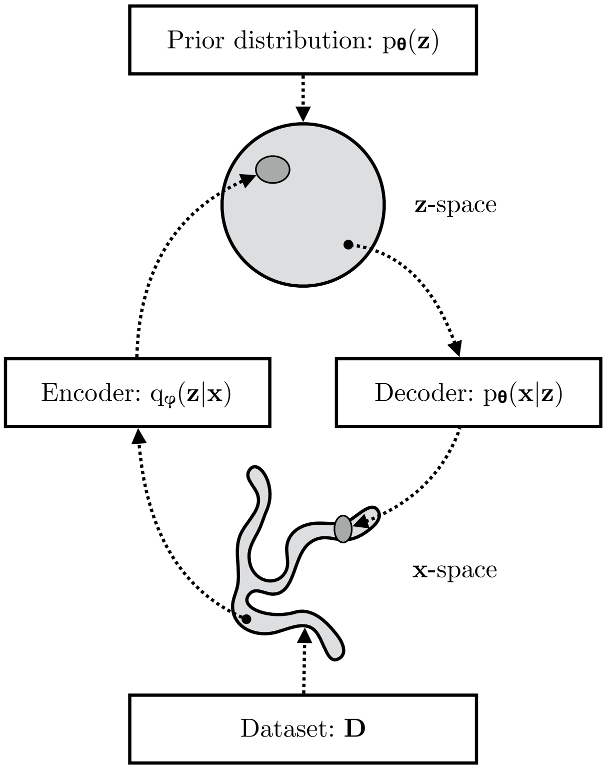

Prior distribution: $p_{\boldsymbol{\theta}}(\mathbf{z})$
$\mathbf{z}$-space
Encoder: $q_{\varphi}(\mathbf{z}|\mathbf{x})$
Decoder: $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$
$\mathbf{x}$-space
Dataset: **D**

Figure 2.1: Representation of a VAE

encoded to $q_{\phi}(\mathbf{z}|\mathbf{x})$ with $p_{\theta}(\mathbf{z})$ as a prior. This prior distribution is tractable, generally assumed as a standard normal distribution. From this encoded representation $\mathbf{z}$, downstream tasks such as reconstruction or prediction can then be performed.

Thus the VAE optimization is to maximize $\mathcal{L}_{\theta,\phi}$, this can be done using standard optimization methods like gradient descent.

The gradients of ELBO with respect to the model parameters θ are –

$$\begin{aligned}
\nabla_{\theta}\mathcal{L}_{\theta,\phi}(\mathbf{x}) &= \nabla_{\theta}\mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\log p_{\theta}(\mathbf{x},\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x})\right] && (2.5)\\
&= \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\nabla_{\theta}(\log p_{\theta}(\mathbf{x},\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x}))\right] && (2.6)\\
&\simeq \nabla_{\theta}(\log p_{\theta}(\mathbf{x},\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x})) && (2.7)\\
&= \nabla_{\theta}(\log p_{\theta}(\mathbf{x},\mathbf{z})) && (2.8)
\end{aligned}$$

Above, $\mathbf{z}$ in the last two lines is a random sample from $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$. The gradients of ELBO with respect to the encoder parameters are –

$$\begin{aligned}\nabla_{\phi}\mathcal{L}_{\theta,\boldsymbol{\phi}}(\mathbf{x}) &= \nabla_{\phi}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\theta}(\mathbf{x},\mathbf{z}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\right] \qquad (2.9)\\ &\neq \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\nabla_{\phi}(\log p_{\theta}(\mathbf{x},\mathbf{z}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}))\right] \qquad (2.10)\end{aligned}$$

Since, backpropagation in case of random sampling would not work directly, a reparameterization trick is employed to convert that random sampling into a determistic function along with a random noise ϵ.

When $p_{\theta}(\mathbf{z})$ is taken to be $\mathcal{N}(0,1)$, the encoder would estimate μ and σ, and $\epsilon \sim \mathcal{N}(0,1)$, $\mathbf{z} = \mu + \sigma\epsilon$. The figure represents the reparameterization trick.

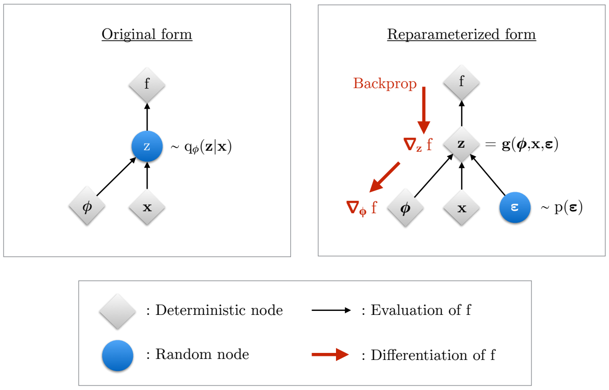


Figure 2.2: The reparameterization trick

2.2 Handpose Estimation

Human pose estimation aims to locate the human body parts and build human body representation (e.g., body skeleton) from input data such as images and videos. It has drawn increasing attention during the past decade and has been utilized in a wide range of applications including human-computer interaction, motion analysis, augmented reality, and virtual reality.

Hand pose estimation aims to locate the joints on a human hand. It plays an important role in areas such as human activity analysis, human computer interaction, and robotics. In 3D hand pose estimation, the aim is to also capture the depth information along with the general 2D location. Unimodal hand pose systems try to directly predict the 3D coordinates using a RGB image, whereas multimodal models make use of additional modalities such as depth maps, pointclouds, segmentation maps etc. However, these approaches are limited by the type of input needed during inference, that is for example, a multimodal handpose estimation model using depth maps requires a depth camera along with a RGB camera to predict the 3D coordinates. A better solution is to use the corresponding depth data which is readily available during training but only using RGB images for inference.

As mentioned earlier, the challenge in hand pose estimation is that simpler techniques (regression from a RGB image) fail to make use of additional information available during training while multimodal techniques have higher data requirements during test. Another challenge with hand pose estimation is the difficulty in annotating the ground truth labels, since it is quite a time taking task.

Chapter 3

Multimodal 3D Handpose Estimation using VAEs

3.1 Methods

In this section, we explore possible VAE based methods for multimodal handpose estimation. The methods themselves can be applied to other problems as well, we treat them in a general fashion for now. We follow [11], propose novel ideas in the upcoming chapters.

3.1.1 Crossmodal VAE

Given data sample $\mathbf{x}$ from some input modality, the cross modal VAE aims to estimate its corresponding target value $\mathbf{y}$ in a target modality by maximizing the evidence lower bound (ELBO) via a latent variable $\mathbf{z}$. The log likelihood is given as –

$$\begin{aligned} \log p(\mathbf{y}) &\geq ELBO_{cVAE}(\mathbf{x};\mathbf{y};\theta,\phi) && (3.1) \\ &= E_{z\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \log p_{\theta}(\mathbf{y}|\mathbf{z}) - \beta D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) && (3.2) \end{aligned}$$

Here β is a scaling parameter, and $p(\mathbf{z}) = \mathcal{N}(0, I)$. Here we do not make use of additional modalities.

Figure 3.1 shows the graphical model of a Crossmodal VAE. The red arrow denotes the encoder $q_{\boldsymbol{\phi}}$, the black arrow denotes the decoder $p_{\theta}(\mathbf{y}|\mathbf{z})$.

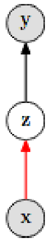


Figure 3.1: Crossmodal VAE

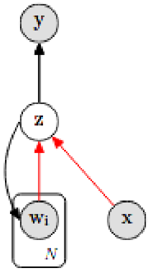
3.1.2 Extended Crossmodal VAE

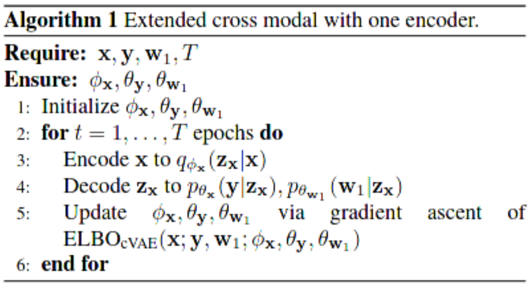
In addition to $\mathbf{x}$ and $\mathbf{y}$, we assume that there are corresponding data from N other modalities $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ and that these modalities are conditionally independent given latent representation $\mathbf{z}$.
The log likelihood in this case is –

$$\begin{aligned}\log p(\mathbf{y}, \mathbf{w}_1) &\geq ELBO_{cVAE}(\mathbf{x}, \mathbf{w}_1; \mathbf{y}, \mathbf{w}_1; \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}, \phi_{\mathbf{x},\mathbf{w}_1}) \qquad (3.3)\\ &= E_{z\sim q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}|\mathbf{x},\mathbf{w}_1)} \log p_{\theta_{\mathbf{y}}}(\mathbf{y}|\mathbf{z})\\ &+ \lambda_{\mathbf{w}_1} E_{z\sim q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}|\mathbf{x},\mathbf{w}_1)} \log p_{\theta_{\mathbf{w}_1}}(\mathbf{w}_1|\mathbf{z})\\ &- \beta D_{KL}(q_{\phi_{\mathbf{x},\mathbf{w}_1}}||p(\mathbf{z})) \qquad (3.4)\end{aligned}$$

Here, we considered $N = 1$, but the same equation can be generalized for any N. $\lambda_{\mathbf{w}_1}$ is a hyperparameter that regulates the reconstruction accuracy between $\mathbf{w}_1$ and $\mathbf{y}$, β is a scaling hyperparameter for KL Divergence.

Essentially, we are encoding both $\mathbf{x}$ and $\mathbf{w}_1$ into a joint representation $\mathbf{z}$, which we are using to decode $\mathbf{y}$ and $\mathbf{w}_1$ again. The graphical model of the extended Crossmodal VAE is shown in the figure below. Again, the red lines represent the encoders for $\mathbf{x}$ and $\mathbf{w}_1$, the black lines represent the decoders for $\mathbf{y}$ and $\mathbf{w}_1$. Also given below, is the algorithm for training an extended Crossmodal VAE.





```
Algorithm 1 Extended cross modal with one encoder.
Require: $\mathbf{x}, \mathbf{y}, \mathbf{w}_1, T$
Ensure: $\phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$
Initialize $\phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$
for $t = 1, \ldots, T$ epochs do
    Encode $\mathbf{x}$ to $q_{\phi_{\mathbf{x}}}(\mathbf{z}_{\mathbf{x}}|\mathbf{x})$
    Decode $\mathbf{z}_{\mathbf{x}}$ to $p_{\theta_{\mathbf{x}}}(\mathbf{y}|\mathbf{z}_{\mathbf{x}}), p_{\theta_{\mathbf{w}_1}}(\mathbf{w}_1|\mathbf{z}_{\mathbf{x}})$
    Update $\phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$ via gradient ascent of
    $\mathrm{ELBO}_{\mathrm{cVAE}}(\mathbf{x}; \mathbf{y}, \mathbf{w}_1; \phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1})$
end for
```

Figure 3.2: Extended Crossmodal VAE

3.1.3 Latent Space Alignment

An alternative solution is to learn $q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}|\mathbf{x}, \mathbf{w}_1)$ and $q_{\phi_{\mathbf{x}}}(\mathbf{z}|\mathbf{x})$ jointly and ensure that they correspond, i.e. are equivalent, by aligning the two distributions together.

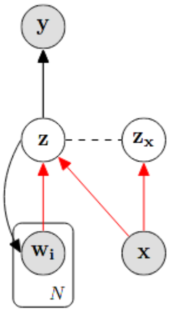
More specifically, we would like to align $\mathbf{z}_{\mathbf{x}}$ (the latent representation learned only from $\mathbf{x}$), with the joint latent representation $\mathbf{z}_{joint}$ learned from both $\mathbf{x}$ and $\mathbf{w}$ to leverage the modalities of $\mathbf{w}$. One can also regard this as bringing together $q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}|\mathbf{x}, \mathbf{w}_1)$ and $q_{\phi_{\mathbf{x}}}(\mathbf{z}|\mathbf{x})$ as close as possible. Now, this alignment can be done in multiple ways –

KL Divergence

An intuitive way of aligning one latent space with another is to incorporate an additional loss term to reduce the divergence between $q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}|\mathbf{x}, \mathbf{w}_1)$ and $q_{\phi_{\mathbf{x}}}(\mathbf{z}|\mathbf{x})$. The log likelihood is then given by –

$$
\begin{aligned}
\mathcal{L}\left(\phi_{\mathbf{x},\mathbf{w}_1}, \phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \\
&= \mathrm{ELBO}_{\mathrm{cVAE}}\left(\mathbf{x}, \mathbf{w}_1; \mathbf{y}, \mathbf{w}_1; \phi_{\mathbf{x},\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \\
&+ \mathrm{ELBO}_{\mathrm{cVAE}}\left(\mathbf{x}; \mathbf{y}, \mathbf{w}_1; \phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \\
&- \beta D_{KL}\left(q_{\phi_{\mathbf{x},\mathbf{w}_1}}\left(\mathbf{z}_{\mathbf{joint}} \mid \mathbf{x}, \mathbf{w}_1\right) \| q_{\phi_{\mathbf{x}}}\left(\mathbf{z}_{\mathbf{x}} \mid \mathbf{x}\right)\right) \qquad (3.5)
\end{aligned}
$$

The graphical model and the algorithm for this approach is given in figure below. The dashed line denotes the operation of embedding crossmodal latent spaces into a joint shared latent space.



```
Require: $\mathbf{x}, \mathbf{y}, \mathbf{w}_1, T$
Ensure: $\phi_{\mathbf{x}}, \phi_{\mathbf{x},\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$
Initialize $\phi_{\mathbf{x}}, \phi_{\mathbf{x},\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$
for $t = 1, \ldots, T$ epochs do
    Encode $\mathbf{x}$ to $q_{\phi_{\mathbf{x}}}(\mathbf{z_x}|\mathbf{x})$
    Encode $\mathbf{x}, \mathbf{w}_1$ to $q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}_{\text{joint}}|\mathbf{x}, \mathbf{w}_1)$
    Decode $\mathbf{z_x}$ to $p_{\theta_{\mathbf{x}}}(\mathbf{y}|\mathbf{z_x}), p_{\theta_{\mathbf{w}_1}}(\mathbf{w}_1|\mathbf{z_x})$
    Decode $\mathbf{z}_{\text{joint}}$ to $p_{\theta_{\mathbf{x}}}(\mathbf{y}|\mathbf{z}_{\text{joint}}), p_{\theta_{\mathbf{w}_1}}(\mathbf{w}_1|\mathbf{z}_{\text{joint}})$
    Construct $D_{KL}(q_{\phi_{\mathbf{x},\mathbf{w}_1}}(\mathbf{z}_{\text{joint}}|\mathbf{x}, \mathbf{w}_1)||q_{\phi_{\mathbf{x}}}(\mathbf{z_x}|\mathbf{x}))$
    Update $\phi_{\mathbf{x}}, \phi_{\mathbf{x},\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$ via gradient ascent of Eq. 3
end for
```

Figure 3.3: Latent Space Alignment: KL Divergence

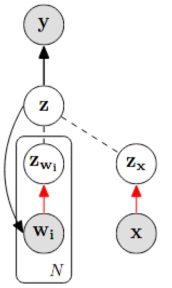
The above formulation suffers from a challenge that as the number of modalities N increases the joint encoder $q_{\phi_{\mathbf{x},\mathbf{w}_1}}$ becomes very difficult to learn. Further, in this approach we are not using the data pairs $(\mathbf{w}_1, \mathbf{y})$ directly. An alternative alignment method is described next.

Modality Mixing

Latent space alignment can be carried out by mixing crossmodal latent spaces, where these crossmodal latent spaces are obtained from individual encoders for each modality. The log likelihood is given as –

$$\begin{aligned}
&\mathcal{L}\left(\phi_{\mathbf{x}}, \phi_{\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \\
&= ELBO_{\text{cVAE}}\left(\mathbf{x}; \mathbf{y}, \mathbf{w}_1; \phi_{\mathbf{x}}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \\
&+ ELBO_{\text{cVAE}}\left(\mathbf{w}_1; \mathbf{y}, \mathbf{w}_1; \phi_{\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \\
&+ ELBO_{\text{cVAE}}\left(\mathbf{x}, \mathbf{w}_1; \mathbf{y}, \mathbf{w}_1; \phi_{\mathbf{x}}, \phi_{\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}\right) \qquad (3.6) \\
&= E_{\mathbf{z_x} \sim q_{\phi_{\mathbf{x}}}} \log p_\theta\left(\mathbf{y}, \mathbf{w}_1 \mid \mathbf{z_x}\right) \\
&+ E_{\mathbf{z}_{\mathbf{w}_1} \sim q_{\phi_{\mathbf{w}_1}}} \log p_\theta\left(\mathbf{y}, \mathbf{w}_1 \mid \mathbf{z}_{\mathbf{w}_1}\right) \\
&+ E_{\mathbf{z_{joint}} \sim \text{Mix}(\mathbf{z_x}, \mathbf{z}_{\mathbf{w}_1})} \log p_\theta\left(\mathbf{y}, \mathbf{w}_1 \mid \mathbf{z_{joint}}\right) \\
&- \beta\left(D_{KL}\left(q_\phi\left(\mathbf{z_x} \mid \mathbf{x}\right) \| p(\mathbf{z})\right) + D_{KL}\left(q_\phi\left(\mathbf{z}_{\mathbf{w}_1} \mid \mathbf{w}_1\right) \| p(\mathbf{z})\right)\right) \qquad (3.7)
\end{aligned}$$

Essentially, here we are encoding all modalities into their individual latent space, enforcing a KL Divergence with the prior, mixing these individual latent spaces into a joint latent space, finally reconstructing back the modalities. **Mix** denotes the mixing of modalities here, which is described in the later chapters. It can be seen as a function which takes as input multiple latent representations (from each modality) and outputs a joint latent representation. The graphical model and the algorithm for the above formulation is shown below.



```
Require: $\mathbf{x}, \mathbf{y}, \mathbf{w}_1, T$
Ensure: $\phi_{\mathbf{x}}, \phi_{\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$
Initialize $\phi_{\mathbf{x}}, \phi_{\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$
for $t = 1, \ldots, T$ epochs do
    Encode $\mathbf{x}$ to $q_{\phi_{\mathbf{x}}}(\mathbf{z_x}|\mathbf{x})$
    Encode $\mathbf{w}_1$ to $q_{\phi_{\mathbf{w}_1}}(\mathbf{z}_{\mathbf{w}_1}|\mathbf{w}_1)$
    Construct $\mathbf{z}_{\text{joint}} = \text{GProd}(\mathbf{z_x}, \mathbf{z}_{\mathbf{w}_1})$
    Decode $\mathbf{z_x}, \mathbf{z}_{\mathbf{w}_1}, \mathbf{z}_{\text{joint}}$ to $p_{\theta_{\mathbf{x}}}(\mathbf{y}|\cdot), p_{\theta_{\mathbf{w}_1}}(\mathbf{w}_1|\cdot)$ respectively
    Update $\phi_{\mathbf{x}}, \phi_{\mathbf{w}_1}, \theta_{\mathbf{y}}, \theta_{\mathbf{w}_1}$ via gradient ascent of Eq. 4
end for
```

Figure 3.4: Latent Space Alignment: Modality Mixing

Chapter 4

Implementation Details

4.1 Dataset

We use the Rendered Handpose Dataset (RHD), which is a synthesized dataset of rendered hand images with 320×320 resolution from 20 characters performing 39 actions. It is composed of 41238 samples for training and 2728 samples for testing. For each RGB image, a corresponding depth map, segmentation mask, and 3D hand pose are provided. The dataset is highly challenging because of the diverse visual scenery, illumination, and noise. A typical example from the dataset is shown in the figure 4.1 below. In this project, we convert the depth maps into corresponding pointclouds since we find that they give better results.

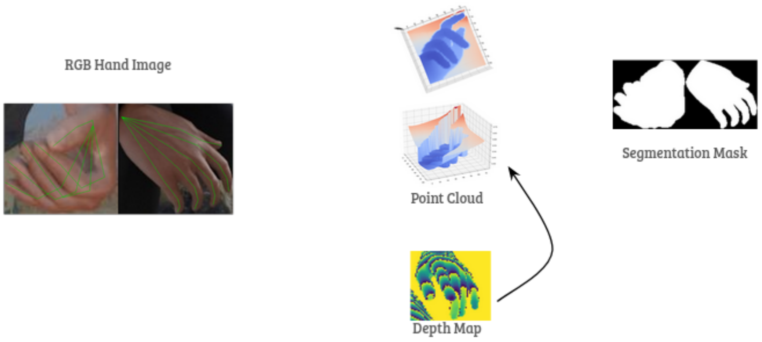


Figure 4.1: Rendered Handpose Dataset

Preprocessing

From the RGB image, the region containing hand is cropped from ground truth masks and resized to 256×256. The corresponding region in the depth image is converted to point clouds using the provided camera intrinsic parameters. For each training step, a different set of 256 points are randomly sampled as training input.

Viewpoint Correction – After cropping the hand from the RGB image, the center of the hand in the image moves from some arbitrary coordinates to the center of the image. As such, the 3D hand pose and associated point cloud must be rotated such that the viewing angle towards the hand aligns with the optical axis. If we assume that the hand's center coordinates are $[u_c, v_c]$ then the rotation matrix $\mathbf{R}_{vc} \in \mathbb{R}^{3\times3}$ can be obtained as follows:

$$\alpha_y = \operatorname{atan2}\left(u_c - o_x, f\right) \tag{4.1}$$

$$\widetilde{\mathbf{c}} = \mathbf{R}_y\left(-\alpha_y\right) \cdot \left[u_c - o_x, v_c - o_y, f\right]^T \tag{4.2}$$

$$\alpha_x = \operatorname{atan2}\left(\widetilde{\mathbf{c}}_2, \widetilde{\mathbf{c}}_3\right) \tag{4.3}$$

$$\mathbf{R}_{vc} = \mathbf{R}_y\left(-\alpha_y\right) \cdot \boldsymbol{R}_x\left(\alpha_x\right) \tag{4.4}$$

Here f is the camera focal length, o_x and o_y are the camera center coordinates.

Augmentation – The images are scaled randomly between [1, 1.2], translated [-20, 20] pixels and rotated by $[-\pi, \pi]$. The pointcloud and 3d pose is accordingly rotated, scaling and translation does not affect them due to normalisation of coordinates. Further we apply a color jitter to the RGB image randomly.

4.2 Model Architecture

The model consists of encoders and decoders for each modality. In particular, we have encoders for RGB and pointcloud, decoders for pose, pointcloud and heatmap. All the encoders output a $\mathbf{d}$ dimensional μ and σ vector, we fix $\mathbf{d} = 64$ in our experiments. For encoding RGB images, we use ResNet-18 [2]. For encoding pointclouds, we use the ResPEL network [5]. The decoder architecture for heatmap is same as that of DC-GAN [6], for decoding pointcloud we use the FoldingNet architecture [13]. Finally, for decoding pose

we use four fully connected layers.
The model architecture is summarised in the figure below.

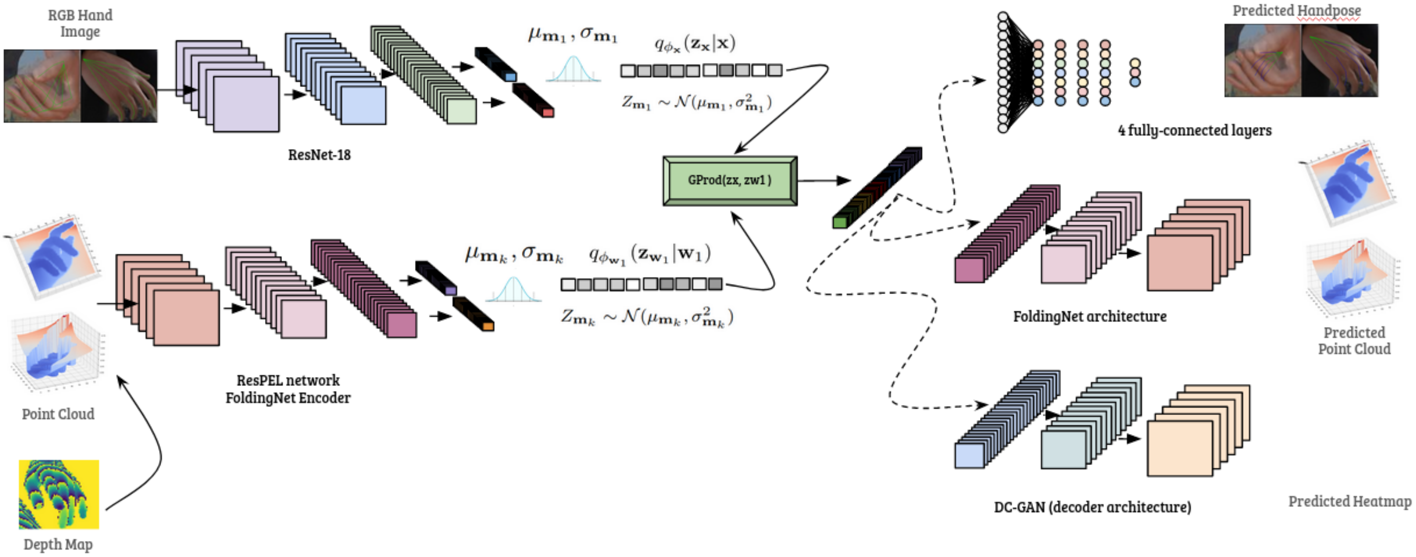

RGB Hand Image
ResNet-18
$\mu_{\mathbf{m}_1}, \sigma_{\mathbf{m}_1}$
$q_{\phi_{\mathbf{x}}}(\mathbf{z}_{\mathbf{x}}|\mathbf{x})$
$Z_{\mathbf{m}_1} \sim \mathcal{N}(\mu_{\mathbf{m}_1}, \sigma^2_{\mathbf{m}_1})$
GProd(zx, zw1)
$\mu_{\mathbf{m}_k}, \sigma_{\mathbf{m}_k}$
$q_{\phi_{\mathbf{w}_1}}(\mathbf{z}_{\mathbf{w}_1}|\mathbf{w}_1)$
$Z_{\mathbf{m}_k} \sim \mathcal{N}(\mu_{\mathbf{m}_k}, \sigma^2_{\mathbf{m}_k})$
Point Cloud
ResPEL network
FoldingNet Encoder
Depth Map
Predicted Handpose
4 fully-connected layers
FoldingNet architecture
Predicted Point Cloud
DC-GAN (decoder architecture)
Predicted Heatmap

Figure 4.2: Model Architecture

4.3 Loss Functions

$$
\begin{aligned}
&\mathcal{L}_{heatmap} = \sum_{j=1}^{J} \left\| \hat{H}_j - H_j \right\| \\
&\mathcal{L}_{chamfer} = \frac{1}{|P|} \sum_{p \in P} \min_{\hat{p} \in \hat{P}} ||\hat{p} - p|| + \frac{1}{|\hat{P}|} \sum_{\hat{p} \in \hat{P}} \min_{p \in P} ||\hat{p} - p\| \\
&\mathcal{L}_{EMD} = \min_{\phi : P \rightarrow P} \frac{1}{|P|} \sum_{p \in P} \|p - \phi(p)\| \\
&\mathcal{L}_{pointcloud} = \mathcal{L}_{chamfer} + \mathcal{L}_{EMD} \\
&\mathcal{L}_{pose} = \|\hat{y} - y\| \\
&\mathcal{L}_{recon} = \mathcal{L}_{pose} + \lambda_{heatmap} \mathcal{L}_{heatmap} + \lambda_{pointcloud} \mathcal{L}_{pointcloud} \\
&\mathcal{L} = \mathcal{L}_{recon} - \beta \left(D_{KL} \left(q_{\phi} \left(\mathbf{z}_{\mathbf{x}} \mid \mathbf{x} \right) \| p(\mathbf{z}) \right) + D_{KL} \left(q_{\phi} \left(\mathbf{z}_{\mathbf{w}_1} \mid \mathbf{w}_1 \right) \| p(\mathbf{z}) \right) \right)
\end{aligned}
$$

Here, J is the number of joints ($= 21$), Chamfer and EMD losses are for pointcloud reconstruction. $\lambda_{heatmap}, \lambda_{pointcloud}$ and β are hyperparameters. We set $\lambda_{heatmap} = 0.01, \lambda_{pointcloud} = 1$ and anneal β from 10^{-5} to 10^{-3} in our experiments.

4.4 Inference

During test, we assume that we do not have any depth information available. Further from the preprocessing perspective, we assume that the hand image is already cropped and centered, the location of the root joint (wrist joint) is known, since the model returns coordinates normalized with respect to this root joint.

Chapter 5

Modality Mixing Techniques

We need to obtain a joint distribution from conditionally independent individual distributions corresponding to each modality. We explore and compare different modality mixing techniques in this section.

5.1 Product of Experts

The joint posterior given N conditionally independent modalities is given by

$$\begin{aligned}
p(\mathbf{z}|\mathbf{x}_1,\ldots,\mathbf{x}_N) &= \frac{p(\mathbf{x}_1,\ldots,\mathbf{x}_N|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x}_1,\ldots,\mathbf{x}_N)} && (5.1)\\
&= \frac{p(\mathbf{z})}{p(\mathbf{x}_1,\ldots,\mathbf{x}_N)}\prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{z}) && (5.2)\\
&= \frac{p(\mathbf{z})}{p(\mathbf{x}_1,\ldots,\mathbf{x}_N)}\prod_{i=1}^{N} \frac{p(\mathbf{z}|\mathbf{x}_i)p(\mathbf{x}_i)}{p(\mathbf{z})} && (5.3)\\
&= \frac{\prod_{i=1}^{N} p(\mathbf{z}|\mathbf{x}_i)}{\prod_{i=1}^{N-1} p(\mathbf{z})}\frac{\prod_{i=1}^{N} p(\mathbf{x}_i)}{p(\mathbf{x}_1,\ldots,\mathbf{x}_N)} && (5.4)\\
&\propto \frac{\prod_{i=1}^{N} p(\mathbf{z}|\mathbf{x}_i)}{\prod_{i=1}^{N-1} p(\mathbf{z})} && (5.5)
\end{aligned}$$

We approximate the true posteriors $p(\mathbf{z}|\mathbf{x}_i)$ by $q(\mathbf{z}|\mathbf{x}_i) \equiv q_\Phi(\mathbf{z}|\mathbf{x}_i)p(z)$ where q_Φ is the underlying inference network. The above equation then becomes

$$p(\mathbf{z}|\mathbf{x}_1, \ldots, \mathbf{x}_N) \propto p(\mathbf{z}) \prod_{i=1}^{N} q_\Phi(\mathbf{z}|\mathbf{x}_i) \tag{5.6}$$

This approximation of the joint posterior distribution using a product of distributions. This is known as Product of Experts (**PoE**) [10]. When the prior $p(\mathbf{z})$ and q_Φ are Gaussian this becomes a product of Gaussian experts, where the mean μ and covariance V of the joint distribution is given as follows

$$T_i = V_i^{-1} \tag{5.7}$$

$$V = \left(\sum_i T_i \right)^{-1} \tag{5.8}$$

$$\mu = \left(\sum_i \mu_i T_i \right) V \tag{5.9}$$

Essentially, the product of experts multiplies the probability density of individual distribution, then normalizes them. One can say that in PoE each distribution has sort of a veto power, where if it's probability density somewhere is low the joint probability at that point is also low. The figure below shows an example of PoE, the red and green curves are probability distributions of individual modalities, the black curve is the joint probability distribution obtained after PoE.

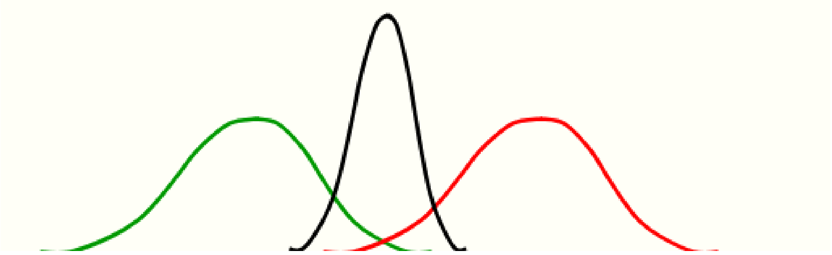


Figure 5.1: Product of Experts

The following figure demonstrates how PoE is useful when one or more modalities are missing during inference. μ_0, σ_0 denote a prior expert which are $0, I$ in our case. μ_i, σ_i are the outputs of individual encoders. When one or more modality is missing we carry out PoE as before with the reduced subset of modalities. This works because we enforce a D_{KL} between each latent space and the prior distribution (which is same as the prior expert). An alternative way is to decode directly from the modality in concern ($\mathbf{z}_{RGB}$) since the loss term also includes crossmodal reconstruction, the latent space $\mathbf{z}_{RGB}$ carries with itself additional information due to alignment during training.

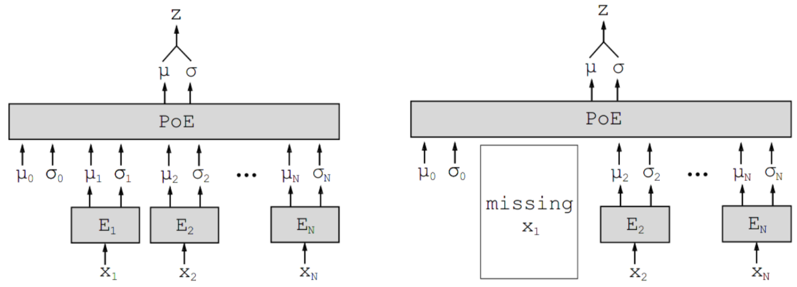


Figure 5.2: Product of Experts with missing modality

5.2 Mixture of Experts

Again approximating the true posterior using q_ϕ, in mixture of experts (**MoE**) the joint posterior is given as

$$p(\mathbf{z}|\mathbf{x}_1,\ldots,\mathbf{x}_N) = sum_{i=1}^{N}\alpha_i q_\phi(\mathbf{z}|\mathbf{x}_i) \tag{5.10}$$

where α_i are weighing parameters. In [7], the authors use $\alpha_i = \frac{1}{N}$ assuming each modality is equally complex. Again assuming independent Gaussian distributions for q_ϕ we can write the joint distribution as

$$\mu = \sum_{i=1}^{N}\alpha_i\mu_i \tag{5.11}$$

$$V = \sum_{i=1}^{N}\alpha_i V_i \tag{5.12}$$

We propose the following modification in mixture of experts – instead of keeping α_i fixed, we also learn these parameters such that the importance of different modalities is automatically reflected. In our experiments with MoE (learnable weights), we find that RGB and depth are assigned a weight in the ratio 3:1 when the training stabilizes.

PoE suffers from overconfident experts, where when one expert has less variance it has more overall influence on the joint distribution leading to a biased mean prediction. MoE on the other hand does not suffer from this problem, since it effectively takes a vote amongst the experts, and spreads its density over all the individual experts. This characteristic makes them better-suited to latent factorisation, being sensitive to information across all the individual modalities.

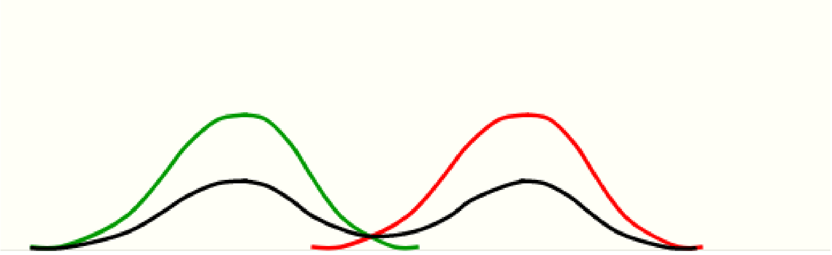


Figure 5.3: Mixture of Experts

5.3 Cross Correlation

To align two latent spaces, we minimize the redundancy between the components of their vectors. That is, it makes the vectors of those spaces similar to each other. We propose to explicitly add a Cross Correlation term in our objective function, where we try to make the cross correlation matrix of two vectors from different latent spaces to be close to the identity matrix [14].

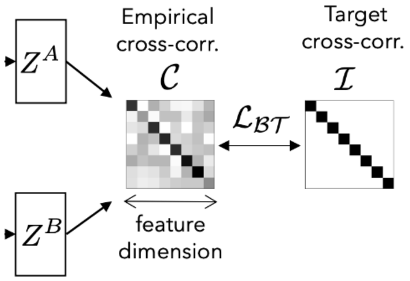


Figure 5.4: Cross Correlation

Here, the loss term can either be an L1 norm or L2 norm, the L1 norm enforces more sparsity.

5.4 Attention

A generalized form of MoE with learnable α_i is using Attention [9] for mixing modalities.

$$Attention(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{5.13}$$

Here we have $Q = \mathbf{z}_{RGB}, K = [\mathbf{z}_{RGB}, \mathbf{z}_{depth}]$ and $V = [\mathbf{z}_{RGB}, \mathbf{z}_{depth}]$. We also experiment with multi-headed attention.

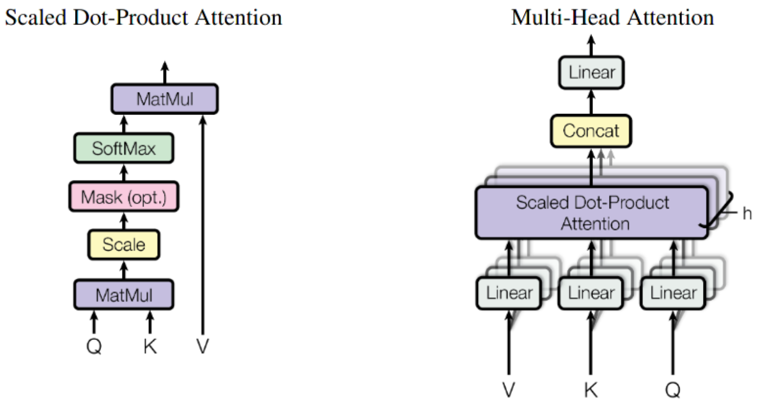


Figure 5.5: Attention

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(h_1, \ldots, h_n)W^o \\ h_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned}$$

Chapter 6

Experiments and Results

6.1 Metrics

Mean EPE

Mean End Point Error (EPE) is defined as the euclidean distance between the prdeicted and the groundtruth 3D pose. We report mean EPE in millimeters.

Area under Curve (PCK)

Percentage Correct Keypoints (PCK) is the percentage of keypoints having euclidean distance from groundtruth below a threshold t. AUC is the area under the curve for PCK vs threshold plot. We take thresholds t from 20mm to 50mm.

6.2 Experiments

We list down our experiments here. We represent RGB by **R**, heatmap by **H**, handpose (keypoints) by **P**, pointcloud by **C** and segmentation map by **S** henceforth. Further, **A2B** implies that the modality A is encoded and B is decoded.

6.2.1 Model Settings

R2P

Crossmodal VAE where the keypoints are directly regressed from RGB images using a crossmodal VAE described earlier.

R2HP

Crossmodal VAE where we decode an additional modality along with handpose. However, we only encode a single modality.

C2P

Crossmodal VAE: Pointcloud to handpose.

RC2CHP

The main multimodal setting, where we encode RGB and pointcloud, decode pointcloud, heatmap and handpose. Here we use different methods for modality mixing.

RCS2CHPS

We include segmentation map in the previous setting during both encoding and decoding. This is done to confirm the hypothesis that PoE suffers when number of modalities are increased due to the higher influence of one modality on overall joint representation.

6.2.2 Modality Mixing

PoE

Product of Experts.

MoE

Mixture of Experts.

MoE-L

Mixture of Experts with learnable weights α_i.

MHTransformerEnc

Multi Headed Transformer Encoder.

PAtt

Attention with primary modality ($\mathbf{z}_{RGB}$) as query.

CrossCorr

One of the above modality mixing methods with cross correlation term included in the loss.

6.3 Results

In this section we report the results we obtained and compare them with the baseline and other prior works.

Model	Modality Mixing	Mean EPE (mm)	AUC
R2P (b)	-	16.61	0.869
R2HP (b)	-	16.10	-
RC2CHP (b)	PoE	13.14	0.943
R2HP (o)	-	14.24	0.93
RC2CHP (o)	PoE	13.65	0.939
RC2CHP (o)	MoE	13.67	0.939
RC2CHP (o)	MoE-L	13.626	0.941
RC2CHP (o)	MHTransformerEnc	13.65	0.94
RC2CHP (o)	PAtt	14.0	0.934

Table 6.1: Comparison with Baseline : b denotes baseline, o denotes ours

Model	**Modality Mixing**	**Cross Correlation**	**Mean EPE (mm)**
RC2CHP (s)	PoE	✗	16.38
RC2CHP (s)	MoE-L	✗	16.29
RC2CHP (s)	MHTransformerEnc	✗	16.13
RC2CHP (s)	PoE	✓	16.26
RC2CHP (s)	MoE-L	✓	16.13
RC2CHP (s)	MHTransformerEnc	✓	16.06

Table 6.2: Cross Correlation, s denotes a 20% subset of data for training

Method	**Mean EPE (mm)**
Spurr *et.al.*[8]	19.73
Yang *et.al.*[12]	19.95
Zimmerman [15]	30.42
Iqbal [3]	13.41
RC2CHP (b)	13.14

Table 6.3: Comparison with other methods

Model	**Modality Mixing**	**EMean EPE (mm)**
RC2CHP (r)	PoE	13.65
RC2CHP (c)	PoE	13.86
RC2CHP (j)	PoE	11.93

Table 6.4: r, c, j denote decoding from $\mathbf{z}_{RGB}, \mathbf{z}_{depth}, \mathbf{z}joint$ respectively

Model	**Modality Mixing**	**EMean EPE (mm)**
RCS2CHPS	PoE	13.96
RCS2CHPS	MoE-L	14.02
RCS2CHPS	MHTransformerEnc	14.03

Table 6.5: Including S

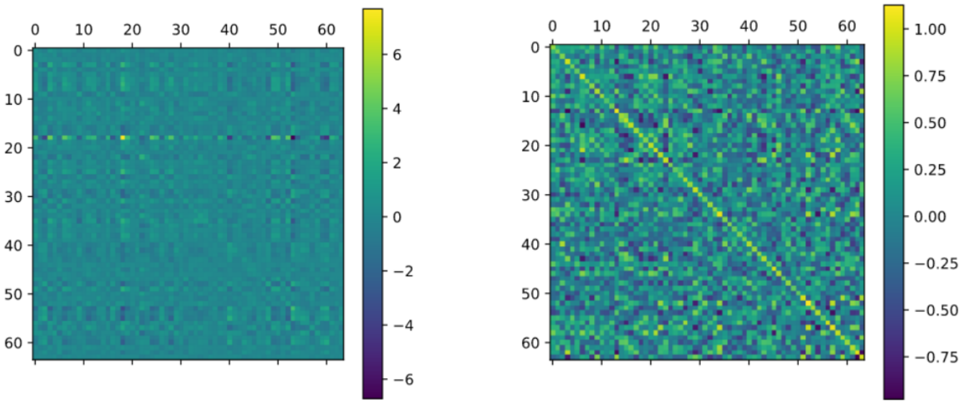


Figure 6.1: Without Cross Correlation

Figure 6.2: With Cross Correlation

Figure 6.3: CrossCorrelation: RC2CHP (MoE-L)

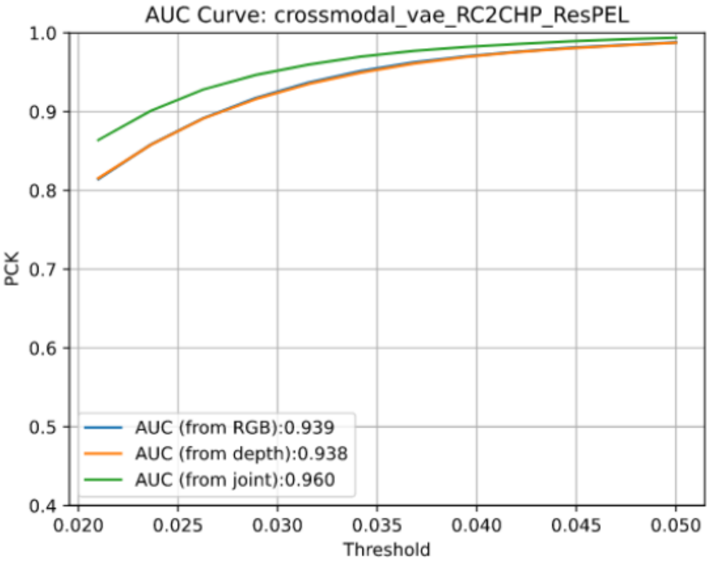


Figure 6.4: AUC: RC2CHP (MoE-L)

Chapter 7

Conclusion and Future Works

From the results mentioned in the previous chapter we can conclude that MoE-L (learnable α) and Transformer with Multi Headed Attention which is essentially a generalized MoE shows better performance than PoE. This difference in performance is more highlighted when we use only 20% of the dataset for training. Thus it hints that, MoE like modality mixing methods are better than PoE. However, the difference with complete dataset is not as significant.
Enforcing cross correlation also has a positive effect when we train on a subset of the dataset. But, this is lessened again with the complete dataset.

Future directions of this work include maximizing the performance of our proposed methods using better training techniques, ablation study on the encoder and decoder model architectures and experimenting with more attention based techniques. Another idea is to use a masked cross correlation loss where we only enforce the regular cross correlation on a subset of dimensions, since each modality also has certain unique features which cannot be aligned with other modalities.

Finally, multimodal fusion has a lot of scope for novel methods since the current methods seem to be heuristic and can be replaced by more flexible methods.

Bibliography

[1] Yujun Cai, Liuhao Ge, Jianfei Cai, and Junsong Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[3] Umar Iqbal, Pavlo Molchanov, Thomas Breuel, Juergen Gall, and Jan Kautz. Hand pose estimation via latent 2.5d heatmap regression, 2018.

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

[5] Shile Li and Dongheui Lee. Point-to-pose voting based hand pose estimation using residual permutation equivariant layer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

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

[7] Yuge Shi, N. Siddharth, Brooks Paige, and Philip H. S. Torr. Variational mixture-of-experts autoencoders for multi-modal deep generative models, 2019.

[8] Adrian Spurr, Jie Song, Seonwook Park, and Otmar Hilliges. Cross-modal deep variational hand pose estimation, 2018.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[10] Mike Wu and Noah Goodman. Multimodal generative models for scalable weakly-supervised learning, 2018.

[11] Linlin Yang, Shile Li, Dongheui Lee, and Angela Yao. Aligning latent spaces for 3d hand pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[12] Linlin Yang and Angela Yao. Disentangling latent hands for image synthesis and pose estimation, 2019.

[13] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation, 2018.

[14] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction, 2021.

[15] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images, 2017.