

A ANALYSIS OF INDEPENDENCE ASSUMPTION AND FACTORIZED OBJECTIVE

Imposing conditional independence assumption to factorize the joint posterior to a number of single-modality posterior trades off the model’s capacity to learn co-representation for flexibility against missing modalities. Here we present the detailed derivation for Subsection 3.4:

$$\begin{aligned}\mathcal{L} &:= ELBO = \mathbb{E}_{z \sim q_\phi(z|x_1, \dots, x_M)} \left[\log p_\theta(x_1, \dots, x_M | z) + \log \frac{p_\theta(z)}{q_\phi(z | x_1, \dots, x_M)} \right] \\ p(x_1, \dots, x_M | z) &= \prod_{i=1}^M p(x_i | z)\end{aligned}\tag{11}$$

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbb{X})} \log \frac{p_\theta(\mathbb{X}, \mathbf{z})}{q_\phi(\mathbf{z} | \mathbb{X})} \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbb{X})} \log \frac{p_\theta(\mathbb{X}, \mathbf{z}) \cdot p_\theta(z) \prod_{i=1}^M p_\theta(x_i | z)}{q_\phi(\mathbf{z} | \mathbb{X}) \cdot p_\theta(z) \prod_{i=1}^M p_\theta(x_i | z)} \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbb{X})} \left[\log \frac{p_\theta(z) \prod_{i=1}^M p_\theta(x_i | z)}{q_\phi(\mathbf{z} | \mathbb{X})} + \log \frac{p_\theta(\mathbb{X}, \mathbf{z})}{p_\theta(z) \prod_{i=1}^M p_\theta(x_i | z)} \right] \\ &= \mathcal{L}_{CI} + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbb{X})} \left[\log \frac{p_\theta(\mathbb{X} | \mathbf{z})}{\prod_{i=1}^M p_\theta(x_i | z)} \right]\end{aligned}\tag{12}$$

$$\begin{aligned}\mathbb{E}_{q(\mathbb{X})} [\mathcal{L}] &= \mathbb{E}_{q(\mathbb{X})} [\mathcal{L}_{CI}] + \mathbb{E}_{q(\mathbb{X})} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbb{X})} \left[\log \frac{p_\theta(\mathbb{X} | \mathbf{z})}{\prod_{i=1}^M p_\theta(x_i | \mathbf{z})} \right] \\ &= \mathbb{E}_{q(\mathbb{X})} [\mathcal{L}_{CI}] + \iint q(\mathbb{X}) q_\phi(\mathbf{z} | \mathbb{X}) \left[\log \frac{p_\theta(\mathbb{X} | \mathbf{z})}{\prod_{i=1}^M p_\theta(x_i | \mathbf{z})} \right] d\mathbf{z} d\mathbb{X} \\ &= \mathbb{E}_{q(\mathbb{X})} [\mathcal{L}_{CI}] + \iint \frac{q(\mathbb{X}) q_\phi(\mathbf{z} | \mathbb{X})}{p_\theta(\mathbb{X} | \mathbf{z})} p_\theta(\mathbb{X} | \mathbf{z}) \left[\log \frac{p_\theta(\mathbb{X} | \mathbf{z})}{\prod_{i=1}^M p_\theta(x_i | \mathbf{z})} \right] d\mathbf{z} d\mathbb{X} \\ &= \mathbb{E}_{q(\mathbb{X})} [\mathcal{L}_{CI}] + \int \frac{q(\mathbb{X}) q_\phi(\mathbf{z} | \mathbb{X})}{p_\theta(\mathbb{X} | \mathbf{z})} \int p_\theta(\mathbb{X} | \mathbf{z}) \left[\log \frac{p_\theta(\mathbb{X} | \mathbf{z})}{\prod_{i=1}^M p_\theta(x_i | \mathbf{z})} \right] d\mathbf{z} d\mathbb{X} \\ &= \mathbb{E}_{q(\mathbb{X})} [\mathcal{L}_{CI}] + \mathbb{E}_{z \sim \frac{q(\mathbb{X}) q_\phi(\mathbf{z} | \mathbb{X})}{p_\theta(\mathbb{X} | \mathbf{z})}} \left[\underbrace{\mathbb{E}_{\mathbb{X} \sim p_\theta(\mathbb{X} | \mathbf{z})} \left[\log \frac{p_\theta(\mathbb{X} | \mathbf{z})}{\prod_{i=1}^M p_\theta(x_i | \mathbf{z})} \right]}_{\text{conditional total correlation}} \right]\end{aligned}\tag{13}$$

, where $\mathbb{X} \equiv (x_1, \dots, x_M)$. In another viewpoint, literatures have pointed out that a larger gap between the objective and the actual log likelihood of the input data will hinder the performance of VAE. Several works thus proposed a tighter ELBO to facilitate the learning process Feng et al. (2019); Burda et al. (2015); Cremer et al. (2017); Li & Turner (2016).

Empirical results from Suzuki et al. (2016) also show that without further constraints, the inferred latent variables from incomplete modalities will be impaired and causes degradation to the generative samples’s quality. Which indicates that z is not completely independent to x_j given only x_i . So it is fair to say that $p(z|x_1, \dots, x_N) \neq p(z|x_i)$. This insight motivate us to drop the conditional independence assumption and find another way to achieve a scalable multimodal model. Different from factorization method, our SMVAE optimize on a tighter lower bound of the data’s log-likelihood while being able to exploit correlations between different modalities.

B DETAILED TRAINING CONFIGURATIONS AND ARCHITECTURES OF ENCODER AND DECODER

For MNIST and FASHION dataset, we consider images as $\mathbf{x} \in \mathbb{R}^{28 \times 28}$ and the corresponding labels as one-hot vectors $\mathbf{y} \in \{0, 1\}^{10}$. The detailed architectures of encoders and decoders for these datasets are listed in Table 3, and Table 4. The size of latent variables is set to 64, i.e., $L = 64$, the weighting coefficient for images is 1.0 while that of the label is set to 10.0, total number of epochs is set to 200, and mini-batch size is set to 100 for all methods. The learning rate is set to $5e^{-4}$ for our SMVAE method while in other comparing methods, it is set to $1e^{-3}$ according to setting reported in their papers. We let β to anneal for 100 out of 200 epochs.

For CelebA dataset, we use crop central face images and resize the image to 64×64 . The resized images are regarded as $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$ while the corresponding attributes are also processed as one-hot vectors $\mathbf{y} \in \{0, 1\}^{18}$. The architectures of encoders and decoders are listed in Table 5. The size of latent variables is set to 100, the weighting coefficient for images is 1.0 and that of the label is set to 10.0, total number of epochs is set to 200, and mini-batch size is set to 100 for all methods. The learning rate is set to $5e^{-4}$ for our SMVAE method while in other comparing methods, it is set to $1e^{-3}$ according to setting reported in their papers. We let β to anneal for 100 out of 200 epochs.

Input $K \in \mathbb{R}^{m \times 512}$	Input $V \in \mathbb{R}^{m \times 512}$	Input $Q \in \mathbb{R}^{1 \times 512}$
FC.512.Swish	FC.512.Swish	FC.512.Swish
Split(512//h)		
Softmax(QK^T / \sqrt{V}) $\times V$		
Skip-Connection, LayerNorm		
FC.L \times 2		

¹ FC.N denotes a fully connected layer with N hidden units and outputs a N dimensional vector.

² h denotes number of attention heads.

³ L denotes the size of latent space

Table 2: Set Encoder architecture.

Encoder-Image	Decoder-Image	Encoder-Label	Decoder-Label
Input $\mathbf{x} \in \mathbb{R}^{1 \times 28 \times 28}$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{y} \in \mathbb{R}^{1 \times 10}$	Input $\mathbf{z} \in \mathbb{R}^L$
FC.512.Swish	FC.512.Swish	FC.512.Swish	FC.512.Swish
-	FC.1 \times 28 \times 28	-	FC.10

¹ FC.N denotes a fully connected layer with N hidden units and outputs a N dimensional vector.

² Swish denotes swish activation function Ramachandran et al. (2017).

³ L denotes the size of latent space

Table 3: Encoder and decoder architecture for MNIST dataset.

For the computer vision application setting, we use a SMVAE model with 250 latent size. The learning rate is set to 10^{-4} . Regular image, obscured image and image with watermarks are regarded as image of $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$ while the other image modalities are regarded as $\mathbf{x} \in \mathbb{R}^{1 \times 64 \times 64}$. Mini-batch size is set to 100, and learning rate is set to $5e^{-5}$. We let β to anneal for 50 epochs out of 100 epochs. Table 6 shows the detailed architectures for encoder and decoder for each modality.

For the robotics application setting, there are a total of 150 trajectories in the Vision&Touch dataset, we use 70% of samples independently drawn from the dataset as the train set and the remaining 30%

Encoder-Image	Decoder-Image	Encoder-Label	Decoder-Label
Input $\mathbf{x} \in \mathbb{R}^{1 \times 28 \times 28}$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{y} \in \mathbb{R}^{1 \times 10}$	Input $\mathbf{z} \in \mathbb{R}^L$
Conv.64.Swish	FC.512.Swish	FC.512.Swish	FC.512.Swish
Conv.128.Swish	FC.128 $\times 7 \times 7$.Swish		FC.10
FC.512.Swish	DeConv.64.Swish	-	-
-	DeConv.1	-	-

¹ Conv. N denotes a convolutional layer with N kernels and N output channels.

² DeConv. N denotes a deconvolutional layer with N kernels and N output channels.

³ BN denotes a batch normalization layer Ioffe & Szegedy (2015).

⁴ Swish denotes swish activation function Ramachandran et al. (2017).

⁵ All the convolutional and deconvolutional layers has strides set to 4 and padding set to 2.

⁶ Other abbreviations can refer to Table 3.

Table 4: Encoder and decoder architecture for experiments on Fashion dataset.

Encoder-Image	Decoder-Image	Encoder-Attributes	Decoder-Attributes
Input $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{y} \in \mathbb{R}^{1 \times 18}$	Input $\mathbf{z} \in \mathbb{R}^L$
Conv.32	FC.256 $\times 5 \times 5$.Swish	FC.512.BN.Swish	FC.512.BN.Swish
Conv.64.BN.Swish	DeConv.128.Swish	FC.512.BN.Swish	FC.512.BN.Swish
Conv.128.BN.Swish	DeConv.64.Swish	FC.512.BN	FC.512.BN.Swish
Conv.256.BN.Swish	DeConv.32.Swish	-	FC.18.BN.Swish
FC.512.Swish	DeConv.3.Swish	-	-

¹ abbreviations can refer to Table 3 and Table 4

Table 5: Encoder and decoder architecture for experiments on CelebA dataset.

Encoder-Img/Obs/Wm	Decoder-Img/Obs/Wm	Encode-Gray/Edg/Mask	Decoder-Gray/Edge/Mask
Input $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$	Input $\mathbf{z} \in \mathbb{R}^L$
Conv.32	FC.256 $\times 5 \times 5$.Swish	Conv.32	FC.256 $\times 5 \times 5$.Swish
Conv.64.BN.Swish	DeConv.128.Swish	Conv.64.BN.Swish	DeConv.128.Swish
Conv.128.BN.Swish	DeConv.64.Swish	Conv.128.BN.Swish	DeConv.64.Swish
Conv.256.BN.Swish	DeConv.32.Swish	Conv.256.BN.Swish	DeConv.32.Swish
FC.512.Swish	DeConv.3.Swish	FC.512.Swish	DeConv.3.Swish

¹ Obs denotes obscured image, Img denotes regular RGB image, Wm denotes image with watermarks.

² abbreviations can refer to Table 3 and Table 4 .

Table 6: Encoder and decoder architecture for experiments on computer vision transformations.

as the test set. The rgb images are regarded as $\mathbf{x}_1 \in \mathbb{R}^{3 \times 64 \times 64}$, the depth images are regarded as $\mathbf{x}_2 \in \mathbb{R}^{1 \times 64 \times 64}$, action commands are regarded as $\mathbf{x}_3 \in \mathbb{R}^4$, and force sensor inputs are regarded as $\mathbf{x}_4 \in \mathbb{R}^{6 \times 32}$. We train a SMVAE model with 64 latent size, and set the learning rate to $1e^{-3}$. We allow β to anneal for 150 epochs out of 200 epochs. The architecture for the encoder and decoder are listed in Table 8 and Table 9

Encoder-Img/Obs/Wm	Decoder-Img/Obs/Wm	Encoder-Gray/Edg/Mask	Decoder-Gray/Edge/Mask
Input $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$	Input $\mathbf{z} \in \mathbb{R}^L$
Conv.32	FC.256 \times 5 \times 5.Swish	Conv.32	FC.256 \times 5 \times 5.Swish
Conv.64.BN.Swish	DeConv.128.Swish	Conv.64.BN.Swish	DeConv.128.Swish
Conv.128.BN.Swish	DeConv.64.Swish	Conv.128.BN.Swish	DeConv.64.Swish
Conv.256.BN.Swish	DeConv.32.Swish	Conv.256.BN.Swish	DeConv.32.Swish
FC.512.Swish	DeConv.3.Swish	FC.512.Swish	DeConv.3.Swish

¹ Obs denotes obscured image, Img denotes regular RGB image, Wm denotes image with watermarks.

² Abbreviations can refer to Table 3 and Table 4.

Table 7: Encoder and decoder architecture for experiments on computer vision transformations.

Encoder-Image	Encoder-Depth	Encoder-Action	Encoder-Force
Input $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$	Input $\mathbf{x} \in \mathbb{R}^{1 \times 64 \times 64}$	Input $\mathbf{x}_3 \in \mathbb{R}^4$	Input $\mathbf{x}_4 \in \mathbb{R}^{6 \times 32}$
Conv.32	Conv.32	FC.512.Swish	CausalConv1D.16
Conv.64.BN.Swish	Conv.64.BN.Swish	FC.512	CausalConv1D.32
Conv.128.BN.Swish	Conv.128.BN.Swish	-	CausalConv1D.64
Conv.256.BN.Swish	Conv.256.BN.Swish	-	CausalConv1D.128
FC.512.Swish	FC.512.Swish	-	FC.512

¹ Other abbreviations can refer to Table 3 and Table 4.

Table 8: Encoder architecture for experiments on robotics dataset.

Decoder-Image	Decoder-Depth	Decoder-Action	Decoder-Force
Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{z} \in \mathbb{R}^L$	Input $\mathbf{z} \in \mathbb{R}^L$
FC.256 \times 5 \times 5.Swish	FC.256 \times 5 \times 5.Swish	FC.512.Swish	FC.128 \times 2.Swish
DeConv.128.Swish	DeConv.128.Swish	FC.4	DeConv1D.64
DeConv.64.Swish	DeConv.64.Swish	-	DeConv1D.32
DeConv.32.Swish	DeConv.32.Swish	-	DeConv1D.16
DeConv.3.Swish	DeConv.3.Swish	-	DeConv1D.6

¹ Abbreviations can refer to Table 3 and Table 4.

Table 9: Decoder architecture for experiments on robotics dataset.

C APPENDIX 3. PROBABILISTIC SYMMETRY

Recent work on probabilistic symmetries has established a correspondence between functional symmetries with probabilistic symmetries. Here we quote some contents from Bloem-Reddy & Teh (2020). As one of their main results, for an exchangeable input \mathbf{X}_n , the conditional distribution of an output \mathbf{Y} is invariant to permutations of \mathbf{X}_n if and only if there exists a function f such that:

$$(\mathbf{X}_n, \mathbf{Y}) \stackrel{a.s.}{=} (x_{set}, f(\eta, \mathbb{M}_{\mathbf{X}_n})) \quad (14)$$

, where $\mathbb{M}_{\mathbf{X}_n}$ is some aggregation function for \mathbf{X}_n in summation form, η is a noise variable that acts as a generic source of stochasticity and $\stackrel{a.s.}{=}$ denotes equal almost surely. An example of this is the noise-outsourced functional representation which is a general version of the reparameterization trick. The equation holds as long as $\mathbb{M}_{\mathbf{X}_n}$ is a function that removes the order of elements in \mathbf{X}_n . We adopt

the above process as support for designing the set representation in our probabilistic model. More detailed explanations can refer to its original paper Bloem-Reddy & Teh (2020).