VII. Variational Autoencoders

Young-geun Kim

Department of Statistics and Probability

STT 997 (SS 2025)

MICHIGAN STATE
UNIVERSITY

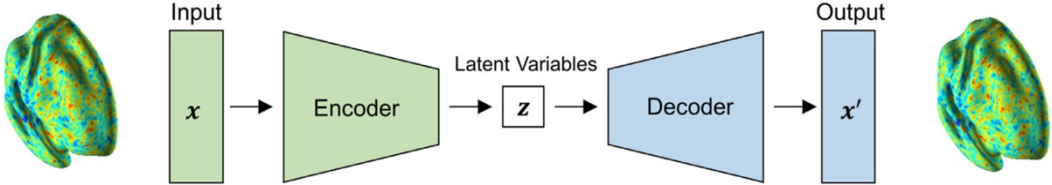
Recap of Previous Section

- In the previous section, we reviewed energy-based models: $p_\theta(\vec{x}) = \int C(\theta)^{-1} \exp(-E_\theta(\vec{z}, \vec{x}))d\vec{z}$.
- The training algorithm, contrastive divergence, requires running a Markov chain (with multi-step Gibbs sampling), which limits the ability to stack multiple layers in their deep learning extensions and extends the training time.
- Furthermore, generation also requires running a Markov chain.
- In this lecture, we will review another generative model, variational autoencoders. By applying variational inference, they do not require running Markov chains during either training or testing times.
- Useful materials include:

  An Introduction to Variational Autoencoders (Kingma et al., 2019).

Outline

Recapping Autoencoder



- We first briefly review autoencoders (Bengio et al., 2006). Autoencoders (AEs) consist of pairs of encoders and decoders that efficiently reduce the dimensionality of data.
- Encoders embed observations into a lower-dimensional space (referred to as *encoding*), while decoders map these encodings back to the original observation space (*decoding* or *reconstruction*).

Images are from Kim et al., 2021.

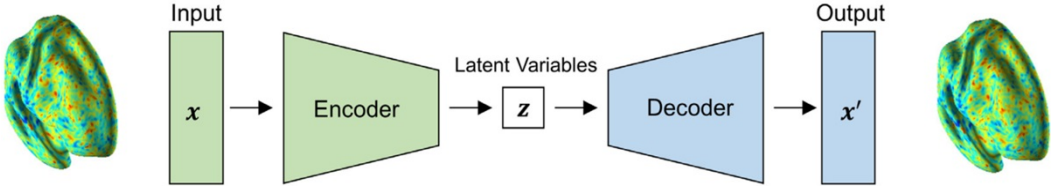
Recapping Autoencoder

- Autoencoders are nonlinear extensions of Principal Component Analysis (Kramer, 1991; Plaut, 2018).
- Assuming the data $(\vec{x}_i)_{i=1}^n$ is centered, for a given dimension r, we define:

$$W^* \in \arg\min_W \left(n^{-1} \sum_{i=1}^n \|\vec{x}_i - WW^T\vec{x}_i\|^2 \right) \text{ subject to } W^TW = I_r. \tag{1}$$

  Here, $W^T\vec{x}_i$ represents the encoding process, and $WW^T\vec{x}_i$ represents the decoding.
- The W^* identifies optimal linear encoder and decoder pairs among symmetric AEs.
- The optimal embeddings $W^{*T}\vec{x}_i$ are the first r principal components up to orthogonal transformations.

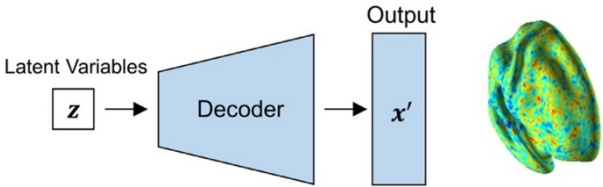
Recapping Autoencoder



- The prefix *auto* is used because they autonomously learn to encode data in an unsupervised manner.
- Autoencoders are trained by minimizing the difference between the original observations and their reconstructions, referred to as the *reconstruction error*.

Images are from Kim et al., 2021.

Motivation: Generation with Nonlinear Decoder



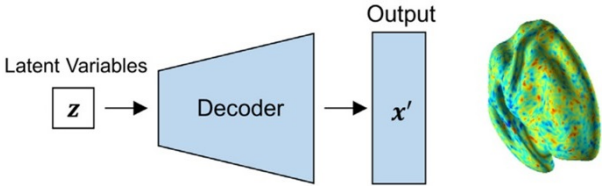
- How about generating data using (nonlinear) decoder networks? That is, modeling the joint density as

$$p_\theta(\vec{z}, \vec{x}) := p(\vec{z}) p_\theta(\vec{x}|\vec{z}) \tag{2}$$

where $p_\theta(\vec{x}|\vec{z}) = p(\vec{x}; \vec{\mu}_{\vec{X}|\vec{Z}}(\vec{z}), \Sigma_{\vec{X}|\vec{Z}}(\vec{z}))$ with $\vec{\mu}_{\vec{X}|\vec{Z}}$ and $\Sigma_{\vec{X}|\vec{Z}}$ being neural networks parameterized by θ. This approach extends the decoder ($\vec{\mu}_{\vec{X}|\vec{Z}}$) to a probabilistic model.

- That is, when we sample $\vec{Z}$ multiple times from $p(\vec{z})$ and sample $\vec{X}$ from this (conditional) Gaussian, the distribution of generated $\vec{X}$ would be: $p_\theta(\vec{x}) = \int p(\vec{z}) p_\theta(\vec{x}|\vec{z}) d\vec{z}$.

Motivation: Generation with Nonlinear Decoder



- The $\vec{\mu}_{\vec{X}|\vec{Z}}$ can be interpreted as a nonlinear mixing function.
- In this context, when $\Sigma_{\vec{X}|\vec{Z}}$ is constant w.r.t. $\vec{Z}$, this formulation can be viewed a nonlinear independent component analysis with observation noise, under certain conditions to guarantee the identifiability (Khemakhem et al., 2020).

Motivation: Generation with Nonlinear Decoder

- Generation with nonlinear decoder networks can enjoy more flexible model classes, but it becomes highly intractable to find optimal θ, e.g., MLEs:

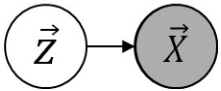
$$\theta^* \in \arg\max_{\theta} \sum_{i=1}^{n} \log p_\theta(\vec{x}_i). \tag{3}$$

Note that $p_\theta(\vec{x}_i) := \int p(\vec{z}) p_\theta(\vec{x}_i|\vec{z}) d\vec{z}$ includes integrating the following nonlinear function:

$$\begin{aligned} & p(\vec{z}) p_\theta(\vec{x}_i|\vec{z}) \\ & \propto \exp(-\vec{z}^T \vec{z}/2) \cdot |\Sigma_{\vec{X}|\vec{Z}}(\vec{z})|^{-1/2} \exp(-(\vec{x}_i - \vec{\mu}_{\vec{X}|\vec{Z}}(\vec{z}))^T \Sigma_{\vec{X}|\vec{Z}}(\vec{z})^{-1} (\vec{x}_i - \vec{\mu}_{\vec{X}|\vec{Z}}(\vec{z}))/2) \end{aligned} \tag{4}$$

- One might consider the Monte Carlo method, i.e., $\log p_\theta(\vec{x}_i) \approx M^{-1} \sum_{m=1}^{M} \log p_\theta(\vec{x}_i|\vec{z}_m)$, but it incurs a large computational cost and may encounter numerical stability issues.
- Furthermore, the inference with $p_\theta(\vec{z}|\vec{x}) := p_\theta(\vec{z}, \vec{x})/p_\theta(\vec{x})$ is also less straightforward.

Variational Autoencoder: Generation

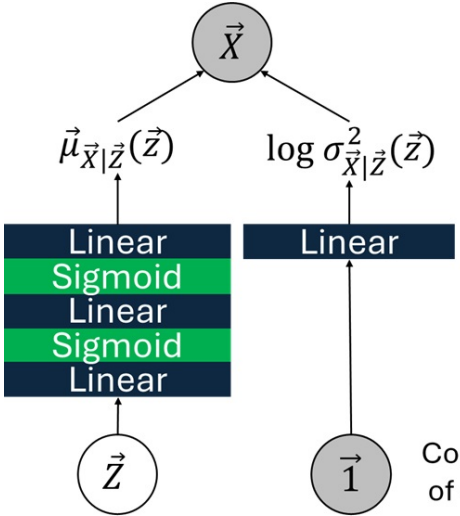


- Variational Autoencoders (VAEs, Kingma and Welling, 2014) utilize *variational inference* (Bishop, 2006) to overcome the aforementioned limitations.
- They model the data generation process using decoder networks with a diagonal covariance $D_{\vec{X}|\vec{Z}} := \text{diag}(\sigma^2_{X_1|\vec{Z}}, \ldots, \sigma^2_{X_p|\vec{Z}})$:

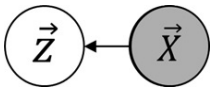
$$p_\theta(\vec{z}, \vec{x}) = \Big(\prod_{j=1}^{r} p(z_j)\Big)\Big(\prod_{j=1}^{p} p_\theta(x_j|\vec{z})\Big)$$

$$= \prod_{j=1}^{r}\left(\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z_j^2}{2}\right)\right)\prod_{j=1}^{p}\left(\frac{1}{\sqrt{2\pi\sigma^2_{X_j|\vec{Z}}(\vec{z})}}\exp\left(-\frac{(x_j - \mu_{X_j|\vec{Z}}(\vec{z}))^2}{2\sigma^2_{X_j|\vec{Z}}(\vec{z})}\right)\right). \quad (5)$$

Variational Autoencoder: Generation



Variational Autoencoder: Inference



- From a variational inference perspective, $p(\vec{z})$ and $p_\theta(\vec{z}|\vec{x})$ are called the *prior* and *posterior* distributions of latent factors $\vec{z}$, respectively. VAEs introduce an encoder, or *inference network*, $q_\phi(\vec{z}|\vec{x})$ to approximate $p_\theta(\vec{z}|\vec{x})$.[1]
- The key idea is that, although $\log p_\theta(\vec{x})$ is intractable,

$$\text{ELBO}(\theta, \phi; \vec{x}) := \log p_\theta(\vec{x}) - \text{KL}(q_\phi(\vec{z}|\vec{x})||p_\theta(\vec{z}|\vec{x})), \tag{6}$$

  which is called *evidence lower bound (ELBO)*, has a computationally tractable form. Under the condition $q_\phi(\vec{z}|\vec{x}) \approx p_\theta(\vec{z}|\vec{x})$, it becomes a tight lower bound of $\log p_\theta(\vec{x})$.
- Furthermore, the $q_\phi(\vec{z}|\vec{x})$ can provide faster computation than using $p_\theta(\vec{z}|\vec{x})$.

[1] VAEs use $q_\phi(\vec{z}|\vec{x})$, a single function (of $\vec{x}$), to approximate the posterior distribution for each new data point, referred to as *amortized variational inference*.

Variational Autoencoder: Inference

- VAEs maximize the average of the ELBO, which is equivalent to minimizing:

$$-\int \text{ELBO}(\theta, \phi; \vec{x}) d\mathbb{P}_n(\vec{x}). \tag{7}$$

- Define $\theta^* \in \arg\min_{\theta} \Big(\min_{\phi} \Big(- \int \text{ELBO}(\theta, \phi; \vec{x}) d\mathbb{P}_n(\vec{x}) \Big) \Big)$. Then,

$$\mathbb{P}_{\theta^*} \in \arg\min_{\mathbb{P}_\theta} \text{KL}(\mathbb{P}_n \| \mathbb{P}_\theta). \tag{8}$$

- We assume that the encoder class $\{q_\phi | \phi \in \Phi\}$ is sufficiently flexible such that for any given θ, there exists a $\phi^*(\theta)$ where: $q_{\phi^*(\theta)}(\vec{z}|\vec{x}) = p_\theta(\vec{z}|\vec{x})$ (a.s. w.r.t. $p(\vec{x})$).

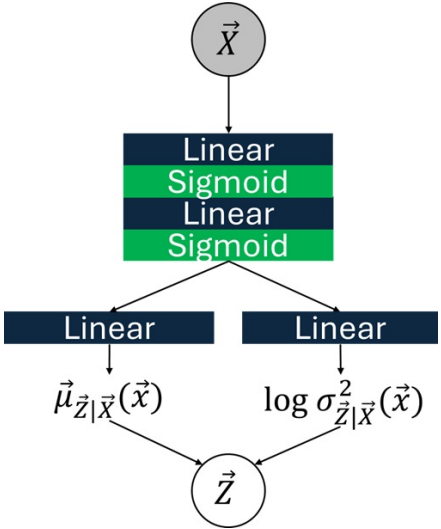
Variational Autoencoder: Inference

Proof: By Equation (9),

$$\begin{aligned}
&\min_{\phi}\Big(- \int \text{ELBO}(\theta, \phi; \vec{x}) d\mathbb{P}_n(\vec{x})\Big) \\
&= \min_{\phi}\Big(- \int \Big(\log p_\theta(\vec{x}) - \text{KL}(q_\phi(\vec{z}|\vec{x}) || p_\theta(\vec{z}|\vec{x}))\Big) d\mathbb{P}_n(\vec{x})\Big) \\
&= \text{KL}(\mathbb{P}_n || \mathbb{P}_\theta) + \min_{\phi} \int \text{KL}(q_\phi(\vec{z}|\vec{x}) || p_\theta(\vec{z}|\vec{x})) d\mathbb{P}_n(\vec{x}) + C.
\end{aligned}$$

Thus, $\min_{\phi}\Big(- \int \text{ELBO}(\theta, \phi; \vec{x}) d\mathbb{P}_n(\vec{x})\Big) = - \int \text{ELBO}(\theta, \phi^*(\theta); \vec{x}) d\mathbb{P}_n(\vec{x}) = \text{KL}(\mathbb{P}_n || \mathbb{P}_\theta)$ up to a constant addition.

Variational Autoencoder: Inference



Variational Autoencoder: Loss Function

- VAEs maximize the ELBO, equivalently, the loss function is the negative ELBO w.r.t. $\log p_\theta(\vec{x})$.
- The ELBO can be expressed as:

$$\begin{aligned} \text{ELBO}(\theta, \phi; \vec{x}) &:= \log p_\theta(\vec{x}) - \text{KL}(q_\phi(\vec{z}|\vec{x}) || p_\theta(\vec{z}|\vec{x})) \\ &= \int \Big(\log p_\theta(\vec{x}|\vec{z}) \Big) q_\phi(\vec{z}|\vec{x}) d\vec{z} - \text{KL}(q_\phi(\vec{z}|\vec{x}) || p(\vec{z})). \end{aligned} \tag{9}$$

Proof: By Bayes' theorem, the relation $p_\theta(\vec{x}) = p_\theta(\vec{x}|\vec{z})p(\vec{z})/p_\theta(\vec{z}|\vec{x})$ holds, implying $\log p_\theta(\vec{x}) - \log \Big(q_\phi(\vec{z}|\vec{x})/p_\theta(\vec{z}|\vec{x}) \Big) = \log p_\theta(\vec{x}|\vec{z}) - \log \Big(q_\phi(\vec{z}|\vec{x})/p(\vec{z}) \Big)$. Taking the expectation over $q_\phi(\vec{z}|\vec{x})$ concludes the proof.

Variational Autoencoder: Loss Function

- The following is another (popular) derivation of the ELBO:

$$
\begin{aligned}
\log p_\theta(\vec{x}) &= \log\left(\int p_\theta(\vec{x}|\vec{z})p(\vec{z})d\vec{z}\right) \\
&= \log\left(\int \frac{p_\theta(\vec{x}|\vec{z})p(\vec{z})}{q_\phi(\vec{z}|\vec{x})} q_\phi(\vec{z}|\vec{x})d\vec{z}\right) \\
&\geq \int \left(\log\left(\frac{p_\theta(\vec{x}|\vec{z})p(\vec{z})}{q_\phi(\vec{z}|\vec{x})}\right)\right) q_\phi(\vec{z}|\vec{x})d\vec{z} \qquad (10) \\
&= \int \left(\log p_\theta(\vec{x}|\vec{z})\right) q_\phi(\vec{z}|\vec{x})d\vec{z} - \int \left(\log \frac{q_\phi(\vec{z}|\vec{x})}{p(\vec{z})}\right) q_\phi(\vec{z}|\vec{x})d\vec{z} \\
&= \int \left(\log p_\theta(\vec{x}|\vec{z})\right) q_\phi(\vec{z}|\vec{x})d\vec{z} - \mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z})).
\end{aligned}
$$

Variational Autoencoder: Loss Function

- In the re-expression of the negative ELBO, $-\int (\log p_\theta(\vec{x}|\vec{z}))\, q_\phi(\vec{z}|\vec{x}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z}))$, the first and second terms are called *reconstruction error* and *KL penalty*, respectively.
- The reconstruction error can be computed via the Monte Carlo method:

$$-\int (\log p_\theta(\vec{x}|\vec{z}))\, q_\phi(\vec{z}|\vec{x}) d\vec{z} \approx -M^{-1} \sum_{m=1}^{M} \log p_\theta(\vec{x}|\vec{z}_m) \tag{11}$$

where $\vec{z}_m \sim q_\phi(\vec{z}|\vec{x})$, and

$$\log p_\theta(\vec{x}|\vec{z}_m) = -\frac{1}{2}\sum_{j=1}^{p} \log(2\pi\sigma^2_{X_j|\vec{Z}}(\vec{z}_m)) - \sum_{j=1}^{p}(x_j - \mu_{X_j|\vec{Z}}(\vec{z}_m))^2/2\sigma^2_{X_j|\vec{Z}}(\vec{z}_m). \tag{12}$$

When we further assume that $\sigma^2_{X_j|\vec{Z}}(\vec{z})$ is constant and $q_\phi(\vec{z}|\vec{x})$ has a negligible uncertainty, this formulation reduces to the reconstruction error of vanilla autoencoders.

Variational Autoencoder: Loss Function

- A typical choice of M in the Monte Carlo method is one.
- In sampling $\vec{z}_m \sim q_\phi(\vec{z}|\vec{x}) = p(\vec{z}; \vec{\mu}_{\vec{Z}|\vec{X}}(\vec{x}), \text{diag}(\sigma^2_{Z_j|\vec{X}}(\vec{x})))$, the following *reparameterization trick* is applied:
  1. Sample $\vec{\epsilon}_m$ from the multivariate standard Gaussian distribution
  2. Compute $\vec{z}_m = \vec{\mu}_{\vec{Z}|\vec{X}}(\vec{x}) + \text{diag}(\sigma_{Z_j|\vec{X}}(\vec{x}))\vec{\epsilon}_m$

  With this approach, we can compute gradients of the reconstruction error, including $\vec{z}_m$, w.r.t. network parameters.

Variational Autoencoder: Loss Function

Q: Recall that, in the motivation slide, we criticized the approximation

$$\log p_\theta(\vec{x}_i) \approx M^{-1} \sum_{m=1}^{M} \log p_\theta(\vec{x}_i|\vec{z}_m) \tag{13}$$

where $\vec{z}_m \sim p(\vec{z})$ for having computational issues. Why is it acceptable now to apply the Monte Carlo method to (in computing reconstruction error part):

$$\log p_\theta(\vec{x}_i) \approx M^{-1} \sum_{m=1}^{M} \log p_\theta(\vec{x}_i|\vec{z}_{i,m}) \tag{14}$$

where $\vec{z}_{i,m} \sim q_\phi(\vec{z}|\vec{x}_i)$?

Variational Autoencoder: Loss Function

- For the second term, the KL penalty term $\mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z}))$, we can use the following formula:

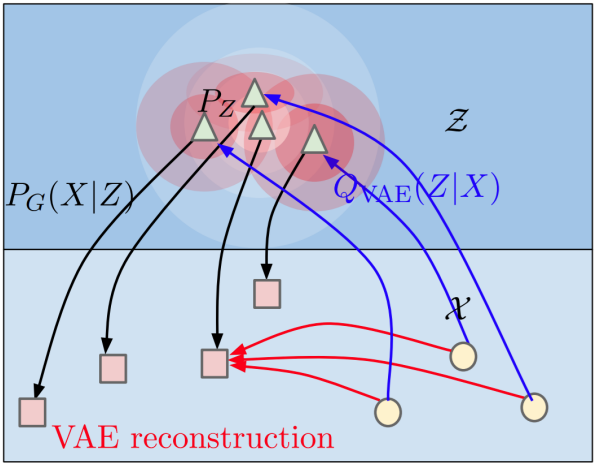
$$\begin{aligned}&\mathrm{KL}(p(\vec{x};\vec{\mu}_1,\Sigma_1)||p(\vec{x};\vec{\mu}_2,\Sigma_2))\\&=\frac{1}{2}\left((\vec{\mu}_2-\vec{\mu}_1)^T\Sigma_2^{-1}(\vec{\mu}_2-\vec{\mu}_1)+\mathrm{tr}(\Sigma_2^{-1}\Sigma_1)-\log\frac{|\Sigma_1|}{|\Sigma_2|}-p\right).\end{aligned} \tag{15}$$

  See `https://statproofbook.github.io/P/mvn-kl.html` for details on the proof.

- The KL penalty term can be expressed as:

$$\frac{1}{2}\left(\sum_{j=1}^{p}\mu^2_{Z_j|\vec{X}}(\vec{x})+\sum_{j=1}^{p}\sigma^2_{Z_j|\vec{X}}(\vec{x})-\sum_{j=1}^{p}\log\sigma^2_{Z_j|\vec{X}}(\vec{x})-p\right). \tag{16}$$

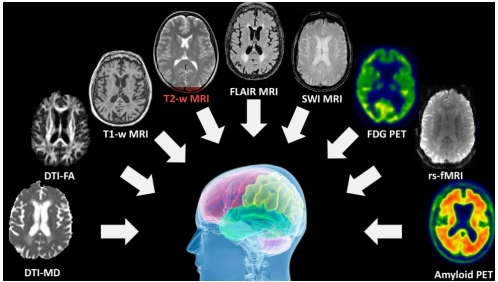
Variational Autoencoder: Loss Function



The figure is from Tolstikhin et al. (2017).

Outline

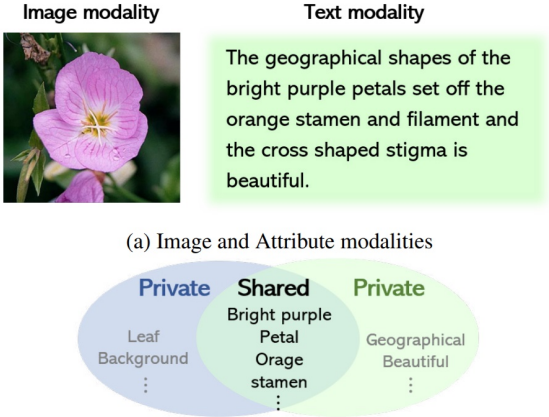
Motivation



- Data often have multi-modal structures.
- Multi-modalities share comprehensive information while each modality also possesses its unique information.
- For example, decomposing the shared and modality-specific information can enhance our understanding of the brain.

The figure is from Gong et al. (2023)

Motivation

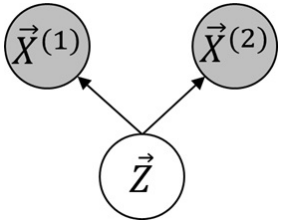


The figure is from Lee and Pavlovic (2021)

Linear Method

- (Traditional) Linear methods are usually based on low-rank matrix compression theory:
  1. Multi-set Canonical Correlation Analysis (Kettenring, 1971)
  2. Joint and Individual Variation Explained (Lock et al., 2013)
  3. Multi-block Partial Least Squares (Chen et al., 2009).
- Multi-modal data may have complicated nonlinear dependencies given latent factors, and we may need to learn mixing mechanisms to analyze new test data.

Joint Multi-modal VAE

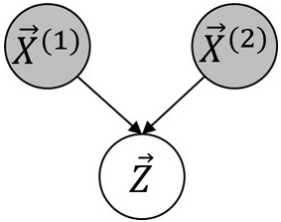


- Joint Multi-modal VAEs (Suzuki et al., 2016) are one of the earliest VAEs-based method, which learn the structure of bi-modal data.
- They assume that two modalities are generated from the same representations:

$$p_\theta(\vec{z}, \vec{x}^{(1)}, \vec{x}^{(2)}) := p(\vec{z}) p_\theta(\vec{x}^{(1)}|\vec{z}) p_\theta(\vec{x}^{(2)}|\vec{z}) \tag{17}$$

where $\vec{x}^{(1)}$ and $\vec{x}^{(2)}$ are realizations of the bi-modal data.

Joint Multi-modal VAE: Inference



- For the inference, they introduce a joint encoder to approximate the posterior distribution of latent variables given the bi-modal data:

$$q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)}). \tag{18}$$

Joint Multi-modal VAE: Loss Function

- Given the generation and inference processes, the loss function, the negative ELBO w.r.t. $\log p_\theta(\vec{x}^{(1)}, \vec{x}^{(2)})$ can be expressed as:

$$
\begin{aligned}
& -\log p_\theta(\vec{x}^{(1)}, \vec{x}^{(2)}) + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})||p_\theta(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})) \\
&= -\int \Big(\log p_\theta(\vec{x}^{(1)}, \vec{x}^{(2)}|\vec{z})\Big) q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})||p(\vec{z})) \\
&= -\int \Big(\log p_\theta(\vec{x}^{(1)}|\vec{z}) + \log p_\theta(\vec{x}^{(2)}|\vec{z})\Big) q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})||p(\vec{z})).
\end{aligned}
\tag{19}
$$

- In the last line, the first term represents the summation of each modality-specific reconstruction errors. This decomposition is based on the property of model classes that $\vec{X}^{(1)}$ and $\vec{X}^{(2)}$ are (conditionally) independent given $\vec{Z}$.

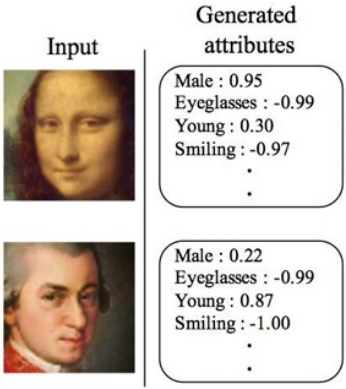
Joint Multi-modal VAE: Addressing Missing Modality

- Block-missing, a situation where some modalities are absent, is a chronic problem in multi-modal data analysis.
- Joint Multi-modal VAEs introduce modality-specific encoders, $q_{\phi^{(1)}}(\vec{z}|\vec{x}^{(1)})$ and $q_{\phi^{(2)}}(\vec{z}|\vec{x}^{(2)})$, to alleviate this issue. The modified objective function can be expressed as:

$$
\begin{aligned}
&-\int \Big(\log p_\theta(\vec{x}^{(1)}|\vec{z}) + \log p_\theta(\vec{x}^{(2)}|\vec{z}) \Big) q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})||p(\vec{z})) \\
&+\alpha \Big(\mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})||q_{\phi^{(1)}}(\vec{z}|\vec{x}^{(1)})) + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})||q_{\phi^{(2)}}(\vec{z}|\vec{x}^{(2)})) \Big).
\end{aligned}
\tag{20}
$$

- The additional penalty terms enforce modality-specific encoders to learn representations from the joint encoder.
- Note that all the posterior distributions, $q_\phi(\vec{z}|\vec{x}^{(1)}, \vec{x}^{(2)})$, $q_{\phi^{(1)}}(\vec{z}|\vec{x}^{(1)})$, and $q_{\phi^{(2)}}(\vec{z}|\vec{x}^{(2)})$, are modeled as Gaussians, the new added terms have closed-form expressions.

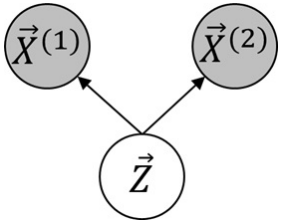
Joint Multi-modal VAE: Result



- Imputation results using $q_{\phi^{(1)}}(\vec{z}|\vec{x}^{(1)})$ and $p_\theta(\vec{x}^{(2)}|\vec{z})$.

The figure is from Suzuki et al. (2016).

Joint Multi-modal VAE: Generation



- (Same data with different methods) What's the difference between the following:
  1. Joint multimodal VAEs,
  2. Two modality-specific VAEs
  3. A single VAE

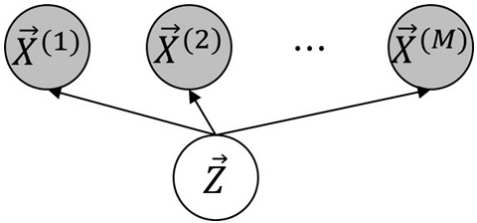
From Bi-modal to Multi-modal

- What happens when the number of modality M increases? When we directly extend the loss function of joint multi-modal VAEs, we have:

$$
\begin{aligned}
&-\log p_\theta(\vec{x}^{(1)}, \ldots, \vec{x}^{(M)}) + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \ldots, \vec{x}^{(2)})) \\
&= -\int \Big(\sum_{m=1}^{M} \log p_\theta(\vec{x}^{(m)}|\vec{z}) \Big) q_\phi(\vec{z}|\vec{x}^{(1)}, \ldots, \vec{x}^{(M)}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1)}, \ldots, \vec{x}^{(M)})||p(\vec{z})).
\end{aligned} \tag{21}
$$

Q: What would be the joint encoder network size for this approach?
Q: Let's consider adding encoders for each observed modality pair. What is the number of possible missing types?

Multi-modal VAE

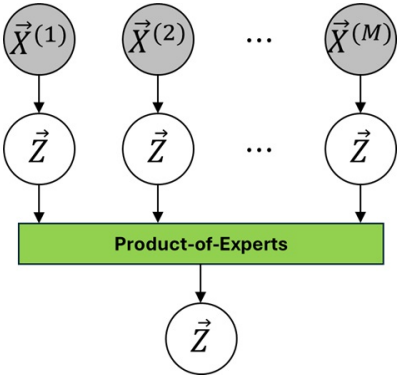


- Multi-model VAEs (Wu and Goodman, 2018) introduce modality-specific VAEs and aggregate information from each modality using product-of-experts techniques (Hinton, 2002).
- Their generation process can be expressed as:

$$p_{\theta^{(1:M)}}(\vec{z}, \vec{x}^{(1:M)}) := p(\vec{z}) \prod_{m=1}^{M} p_{\theta^{(m)}}(\vec{x}^{(m)}|\vec{z}) \tag{22}$$

where $\theta^{(1:M)} := (\theta^{(1)}, \ldots, \theta^{(M)})^T$ and $\vec{x}^{(1:M)} := (\vec{x}^{(1)}, \ldots, \vec{x}^{(M)})^T$.

Multi-modal VAE: Inference



- Product-of-experts techniques enable us to use modality-specific inference networks to build an inference model incorporating all the observed modalities.

Multi-modal VAE: Inference

- When all modalities are observed, the goal of (variational) inference is to approximate:

$$p_{\theta^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)}) \propto \frac{\prod_{m=1}^{M} p_{\theta^{(m)}}(\vec{z}|\vec{x}^{(m)})}{p(\vec{z})^{M-1}} \tag{23}$$

  Hint: Apply Bayes' Theorem twice.

- Here, when each modality-specific encoder perfectly approximates the corresponding posterior, i.e., $q_{\phi^{(m)}}(\vec{z}|\vec{x}^{(m)}) \approx p_{\theta^{(m)}}(\vec{z}|\vec{x}^{(m)})$ for all m, we can utilize them to approximate $p_{\theta^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)})$ using Equation (23).
- In implementation, multi-modal VAEs model the quotient $\tilde{q}_{\phi^{(m)}}(\vec{z}|\vec{x}^{(m)}) := q_{\phi^{(m)}}(\vec{z}|\vec{x}^{(m)})/p(\vec{z})$ instead of $q_{\phi^{(m)}}(\vec{z}|\vec{x}^{(m)})$. With quotient networks, the RHS in Equation (23) can be expressed as $p(z)\prod_{m=1}^{M}\tilde{q}_{\phi^{(m)}}(\vec{z}|\vec{x}^{(m)})$. All the distributions of prior and quotient networks are Gaussian, implying a closed-expression.

Multi-modal VAE: Loss Function

- When all the modalities are observed, the loss function is the summation of three negative ELBOs.

1. $(\vec{X}^{(1:M)})$ The negative ELBO w.r.t. $\log p_{\theta^{(1:M)}}(\vec{x}^{(1:M)})$:

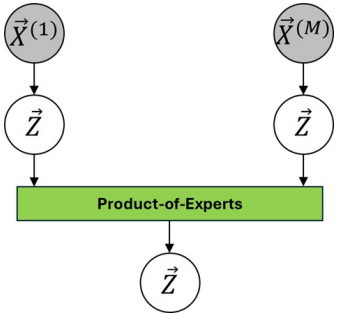
$$
\begin{aligned}
&- \log p_{\theta^{(1:M)}}(\vec{x}^{(1:M)}) + \mathrm{KL}(q_{\phi^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)})||p_{\theta^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)})) \\
&= -\int \Big(\sum_{m=1}^{M} \log p_{\theta^{(m)}}(\vec{x}^{(m)}|\vec{z}) \Big) q_{\phi^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)}) d\vec{z} + \mathrm{KL}(q_{\phi^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)})||p(\vec{z}))
\end{aligned} \tag{24}
$$

   where $q_{\phi^{(1:M)}}(\vec{z}|\vec{x}^{(1:M)}) \propto p(\vec{z}) \prod_{m=1}^{M} \tilde{q}_{\phi^{(m)}}(\vec{z}|\vec{x}^{(m)})$.

2. $(\vec{X}^{(m)})$ The negative ELBO w.r.t. $\log p_{\theta^{(m)}}(\vec{x}^{(m)})$

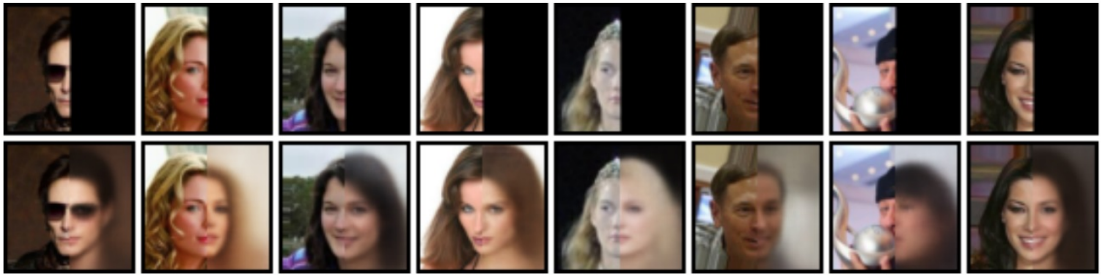
3. $(\vec{X}^{(\pi(1):\pi(m))})$ The negative ELBO w.r.t. $\log p_{\theta^{(\pi(1):\pi(m))}}(\vec{x}^{(\pi(1):\pi(m))})$ where $\pi(1), \ldots, \pi(m)$ are the first m indices from randomly permuted indices.

Multi-modal VAE: Addressing Missing Modalities



- Thanks to the product-of-experts techniques implemented via modality-specific VAEs, the Multimodal VAE can learn networks with arbitrary block-missing types.
- For example, for a sample having only the first and third modalities, $\vec{X}^{(1)}$ and $\vec{X}^{(3)}$, the loss is the summation of the joint ELBO with respect to $(\vec{x}^{(1)}, \vec{x}^{(3)})^T$ and the marginal ELBOs with respect to $\vec{x}^{(1)}$ and $\vec{x}^{(3)}$.

Multi-modal VAE: Result



The figure is from Wu and Goodman (2018).

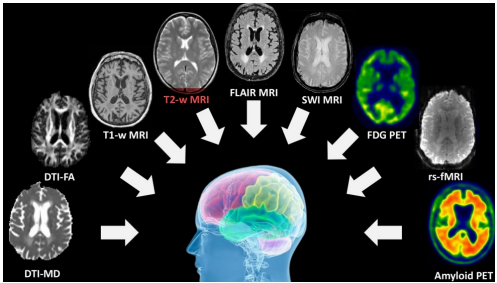
Multi-modal VAE: Result

Type	Sentence
$x_{\text{en}} \sim p_{\text{data}}$	this was one of the highest points in my life.
$x_{\text{vi}} \sim p(x_{\text{vi}}\|z(x_{\text{en}}))$	Đó là một gian tôi vời của cuộc đời tôi.
GOOGLE(x_{vi})	It was a great time of my life.
$x_{\text{en}} \sim p_{\text{data}}$	the project's also made a big difference in the lives of the people .
$x_{\text{vi}} \sim p(x_{\text{vi}}\|z(x_{\text{en}}))$	tôi án này được ra một Điều lớn lao cuộc sống của chúng người sống chữa hưởng .
GOOGLE(x_{vi})	this project is a great thing for the lives of people who live and thrive .
$x_{\text{vi}} \sim p_{\text{data}}$	trước tiên , tại sao chúng lại có ấn tượng xấu như vậy ?
$x_{\text{en}} \sim p(x_{\text{en}}\|z(x_{\text{vi}}))$	first of all, you do not a good job ?
GOOGLE(x_{vi})	First, why are they so bad?
$x_{\text{vi}} \sim p_{\text{data}}$	Ông ngoại của tôi là một người thật đáng <unk> phục vào thời ấy .
$x_{\text{en}} \sim p(x_{\text{en}}\|z(x_{\text{vi}}))$	grandfather is the best experience of me family .
GOOGLE(x_{vi})	My grandfather was a worthy person at the time .

Table 5: Examples of (1) translating English to Vietnamese by sampling from $p(x_{\text{vi}}|z)$ where $z \sim q(z|x_{\text{en}})$, and (2) the inverse. We use Google Translate (GOOGLE) for ground-truth.

The figure is from Wu and Goodman (2018). The training data consist of only 1% completely observed bi-modal data.

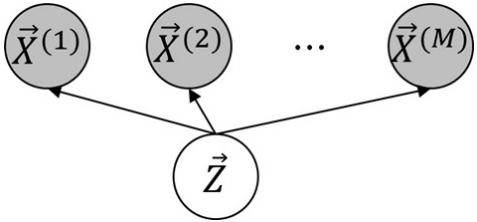
Recapping Motivation



- We reviewed several methods to learn representations of multi-modal data.
- For an advanced analysis, how can we learn representations that interpret comprehensive information across multiple modalities and modality-specific information separately?

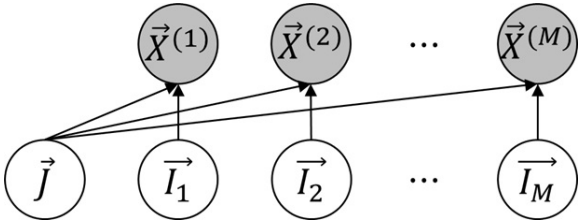
The figure is from Gong et al. (2023)

Recapping Motivation



- Aforementioned methods assume that a single vector of latent factors contribute in generating all M modalities.
- As a consequence, their inference do not distinguish joint information from modality-specific individual one.

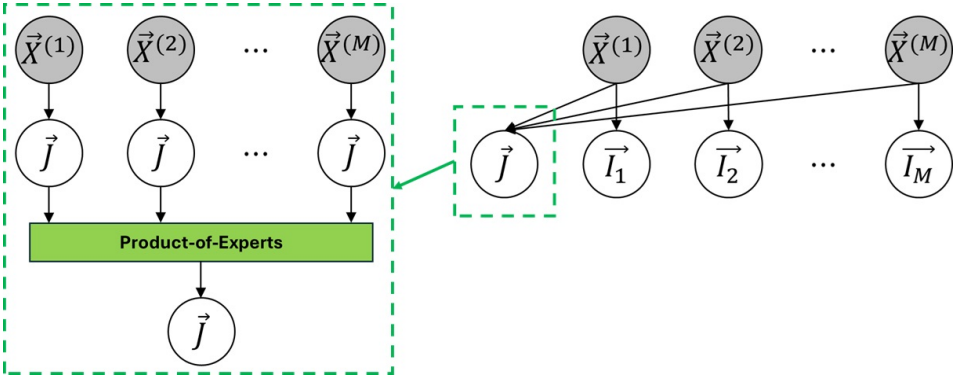
Disentangled Multi-modal VAE



- Disentangled Multi-modal VAE (Lee and Pavlovic, 2021) split the latent factor into joint and individual part.
- Their generation process can be expressed as:

$$p_{\theta^{(1:M)}}(\vec{j}, \vec{i}^{(1:M)}, \vec{x}^{(1:M)}) := p(\vec{j}) \prod_{m=1}^{M} p(\vec{i}) \Big(\prod_{m=1}^{M} p_{\theta^{(m)}}(\vec{x}^{(m)} | \vec{j}, \vec{i}^{(m)}) \Big). \tag{25}$$

Disentangled Multi-modal VAE: Inference



- The inference on the joint part, $\vec{J}$, is based on product-of-experts techniques.
- The inference on the individual part, $\vec{I}^{(m)}$, is based on each modality-specific encoder.

Disentangled Multi-modal VAE: Inference

- Disentangled Multi-modal VAEs consist of:
  1. M modality-specific generators $p_{\theta^{(m)}}(\vec{x}^{(m)}|\vec{j}, \vec{i}^{(m)})$
  2. M pairs of modality-specific encoders $(q_{\phi^{(m)}}(\vec{j}|\vec{x}^{(m)}), q_{\phi^{(m)}}(\vec{i}^{(m)}|\vec{x}^{(m)}))$
- When all modalities are observed, the goal of (variational) inference is to approximate:

$$p_{\theta^{(1:M)}}(\vec{j}, \vec{i}^{(1:M)}|\vec{x}^{(1:M)}) \propto \frac{\prod_{m=1}^{M} p_{\theta^{(m)}}(\vec{j}, \vec{i}^{(m)}|\vec{x}^{(m)})}{p(\vec{j})^{M-1}} \tag{26}$$

  Hint: Apply Bayes' Theorem twice.

  This implies $p_{\theta^{(1:M)}}(\vec{j}|\vec{x}^{(1:M)}) \propto \left(\prod_{m=1}^{M} p_{\theta^{(m)}}(\vec{j}|\vec{x}^{(m)})\right) / \left(p(\vec{j})\right)^{M-1}$. That is, we can apply product-of-experts techniques to $\vec{J}$.
- The inference on the individual part $\vec{I}^{(m)}$ is based on modality-specific encoders for it.

Disentangled Multi-modal VAE: Loss Function

- When all the modalities are observed, the loss function is the summation of two negative ELBOs.

1. $(\vec{X}^{(1:M)})$ The negative ELBO w.r.t. $\log p_{\theta^{(1:M)}}(\vec{x}^{(1:M)})$:

$$
\begin{aligned}
&- \log p_{\theta^{(1:M)}}(\vec{x}^{(1:M)}) + \mathrm{KL}(q_{\phi^{(1:M)}}(\vec{j}, \vec{i}^{(1:M)}|\vec{x}^{(1:M)})||p_{\theta^{(1:M)}}(\vec{j}, \vec{i}^{(1:M)}|\vec{x}^{(1:M)})) \\
&= -\sum_{m=1}^{M} \int \Big(\log p_{\theta^{(m)}}(\vec{x}^{(m)}|\vec{j}, \vec{i}^{(m)}) \Big) q_{\phi^{(1:M)}}(\vec{j}|\vec{x}^{(1:M)}) q_{\phi^{(m)}}(\vec{i}^{(m)}|\vec{x}^{(m)}) d\vec{j} d\vec{i}^{(m)} \\
&\quad + \mathrm{KL}(q_{\phi^{(1:M)}}(\vec{j}|\vec{x}^{(1:M)})||p(\vec{j})) + \sum_{m=1}^{M} \mathrm{KL}(q_{\phi^{(m)}}(\vec{i}^{(m)}|\vec{x}^{(m)})||p(\vec{i}^{(m)}))
\end{aligned}
\tag{27}
$$

where $q_{\phi^{(1:M)}}(\vec{j}, \vec{i}^{(1:M)}|\vec{x}^{(1:M)}) = q_{\phi^{(1:M)}}(\vec{j}|\vec{x}^{(1:M)}) \prod_{m=1}^{M} q_{\phi^{(m)}}(\vec{i}^{(m)}|\vec{x}^{(m)}) \propto p(\vec{j}) \prod_{m=1}^{M} \tilde{q}_{\phi^{(m)}}(\vec{j}|\vec{x}^{(m)}) q_{\phi^{(m)}}(\vec{i}^{(m)}|\vec{x}^{(m)})$.

Hint: $\mathrm{KL}(q_1(\vec{x})q_2(\vec{y})||p_1(\vec{x})p_2(\vec{y})) = \mathrm{KL}(q_1(\vec{x})||p_1(\vec{x})) + \mathrm{KL}(q_2(\vec{y})||p_2(\vec{y}))$.

2. $(\vec{X}^{(m)})$ The negative ELBO w.r.t. $\log p_{\theta^{(m)}}(\vec{x}^{(m)})$.

Disentangled Multi-modal VAE: Loss Function

- In implementation, disentangled multimodal VAEs introduce hyperparameters for the coefficients of each reconstruction error term, and for reweighting the coefficients of each KL term as well.
- When there are missing modalities, they can infer the joint information, $\vec{J}$, using all observed modalities, and infer the individual parts, $\vec{I}^{(m)}$, for each observed modality using the same.

Disentangled Multi-modal VAE: Result

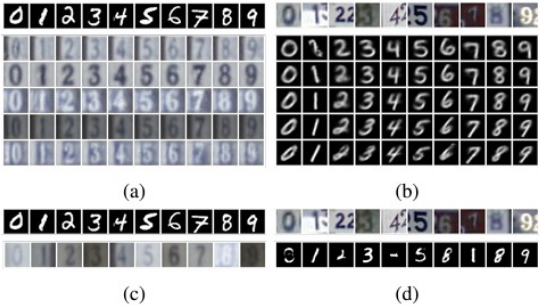


Figure 4: Cross-synthesis images from the opposite modality. (a) and (b) are results with DMVAE, (c) and (d) are results with MVAE. In each image, the first row is the ground truth images from which the share latent code comes. The following rows are the cross-synthesized images.

The figure is from Lee and Pavlovic (2021).

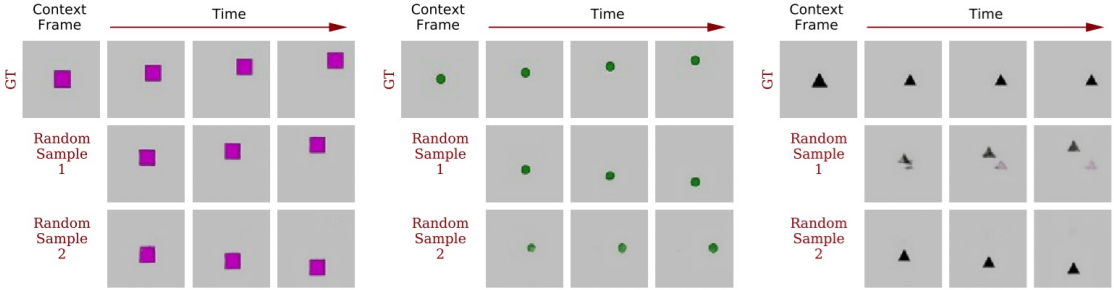
Outline

Motivating Example: Video Generation

- Video frames are popular temporal data, i.e., a time series of image frames at each time step.

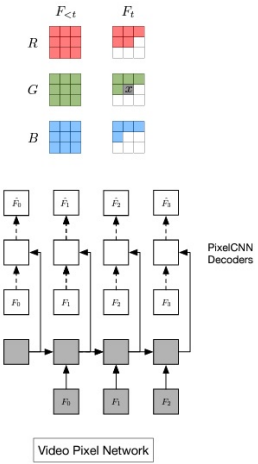
Examples in video generation. Images are from Szeto et al. (2019).

Motivating Example: Uncertainty in Video Data



Images are edited from Babaeizadeh et al. (2017).

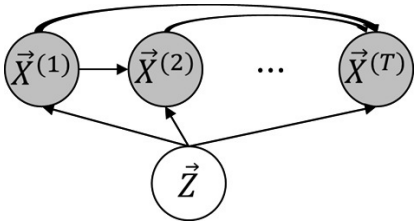
Auto-Regressive Model



- Lotter et al. (2016) proposed a deterministic method that minimizes the MSE between true and predicted future frames.
- This approach can be viewed as an auto-regressive model (under the assumption of additive noise).
- Video Pixel Network (Kalchbrenner et al., 2017) extended image auto-regressive models to videos.

The figure is from Kalchbrenner et al. (2017).

Stochastic Variational Video Prediction

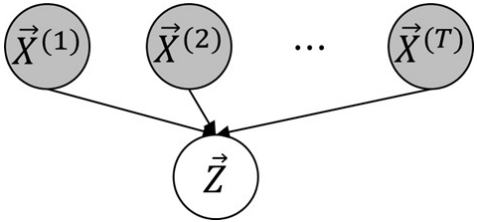


- Stochastic Variational Video Prediction (SV2P) (Babaeizadeh et al., 2017) is one of the earliest VAE-based video generation methods. It proposes two variational approaches: (i) using time-invariant $\vec{Z}$ and (ii) using time-variant $\vec{Z}^{(t)}$.
- We denote the image frame at time t by $\vec{X}^{(t)}$. For (i), the generative model can be expressed as:[2]

$$p_\theta(\vec{z}, \vec{x}^{(1:T)}) = p(\vec{z})p_\theta(\vec{x}^{(1)}|\vec{z})p_\theta(\vec{x}^{(2)}|\vec{z}, \vec{x}^{(1)}) \cdots p_\theta(\vec{x}^{(T)}|\vec{z}, \vec{x}^{(1:(T-1))}). \tag{28}$$

[2]The Stochastic Variational Video Prediction assumes that first few image frames are available at test time. This formulation is a generalized version with a generative model for the initial time step, $p_\theta(\vec{x}^{(1)}|\vec{z})$.

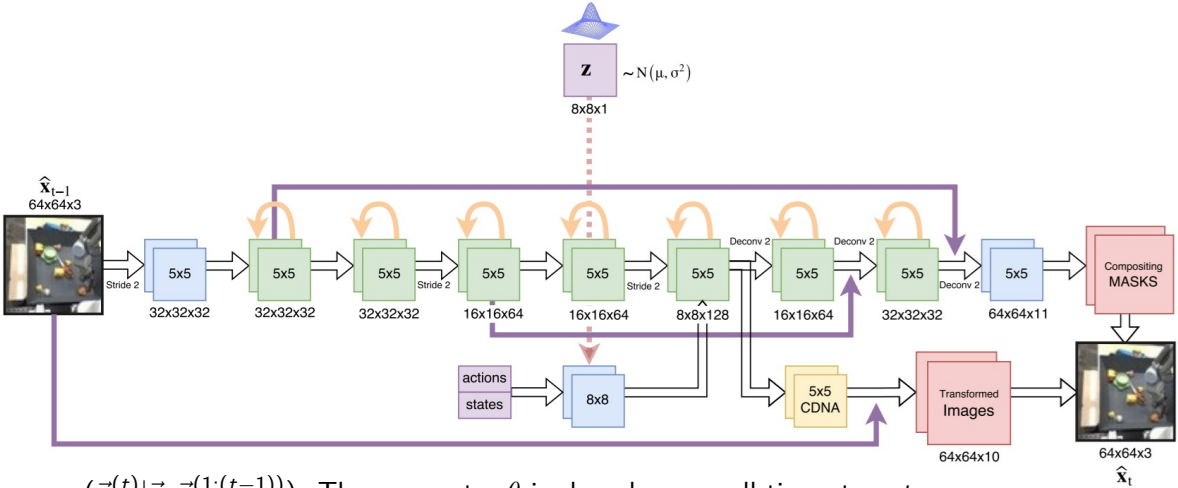
SV2P with a Time-invariant Latent Variable: Inference



- The time-invariant $\vec{Z}$ can be inferred using $q_\phi(\vec{z}|\vec{x}^{(1:T)})$ using all the image frames $\vec{X}^{(1:T)}$.
- The loss function is the negative ELBO w.r.t. $\log p_\theta(\vec{x}^{(1:T)})$:

$$
\begin{aligned}
& -\log p_\theta(\vec{x}^{(1:T)}) + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1:T)})||p_\theta(\vec{z}|\vec{x}^{(1:T)})) \\
&= -\int \left(\log p_\theta(\vec{x}^{(1:T)}|\vec{z})\right) q_\phi(\vec{z}|\vec{x}^{(1:T)}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1:T)})||p(\vec{z})) \\
&= -\sum_{t=1}^{T} \int \left(\log p_\theta(\vec{x}^{(t)}|\vec{z})\right) q_\phi(\vec{z}|\vec{x}^{(1:T)}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x}^{(1:T)})||p(\vec{z})).
\end{aligned}
\tag{29}
$$

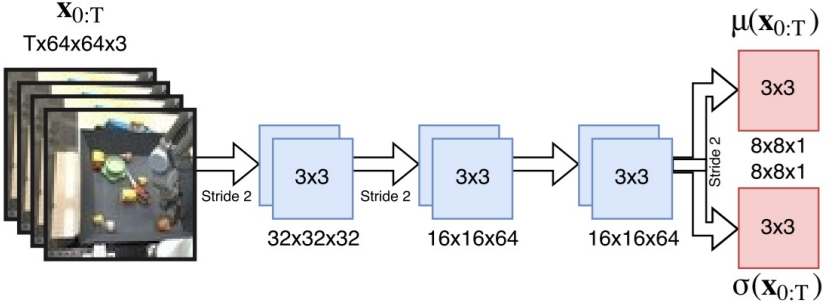
SV2P with a Time-Invariant Latent Variable: Generator Network



- $p_\theta(\vec{x}^{(t)}|\vec{z}, \vec{x}^{(1:(t-1))})$: The parameter θ is shared across all time steps t.

The figure is edited from Babaeizadeh et al. (2017).

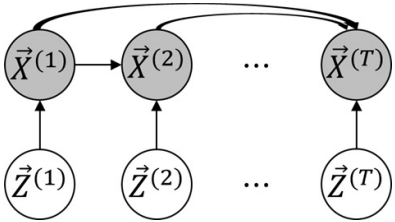
SV2P with a Time-invariant Latent Variable: Inference Network



- $q_\phi(\vec{z}|\vec{x}^{(1:T)})$: The inference network uses all the image frames to infer the time-invariant latent variable.

The figure is from Babaeizadeh et al. (2017).

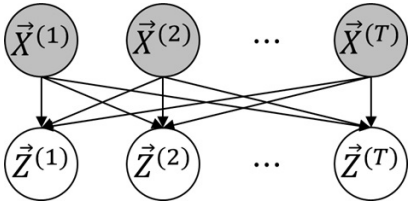
SV2P with a Time-variant Latent Variable: Generation



- For (ii), using time-variant $\vec{Z}^{(t)}$, the generative model can be expressed as:

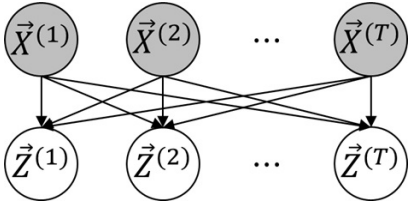
$$p_\theta(\vec{z}^{(1:T)}, \vec{x}^{(1:T)}) = \prod_{t=1}^{T} p(\vec{z}^{(t)}) \prod_{t=1}^{T} p_\theta(\vec{x}^{(t)} | \vec{z}^{(t)}, \vec{x}^{(1:(t-1))}). \tag{30}$$

SV2P with a Time-variant Latent Variable: Inference



- The time-variant $\vec{Z}^{(t)}$ is inferred using all the image frames $\vec{X}^{(1:T)}$.
- The inference network can be formulated as $q_\phi(\vec{z}^{(1:T)}|\vec{x}^{(1:T)}) = \prod_{t=1}^{T} q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:T)})$. Here q_ϕ is assumed to be time-invariant. That is, given videos $\vec{X}^{(1:T)}$, $\vec{Z}^{(1)}, \ldots, \vec{Z}^{(T)}$ are i.i.d. samples.

SV2P with a Time-variant Latent Variable: Loss Function

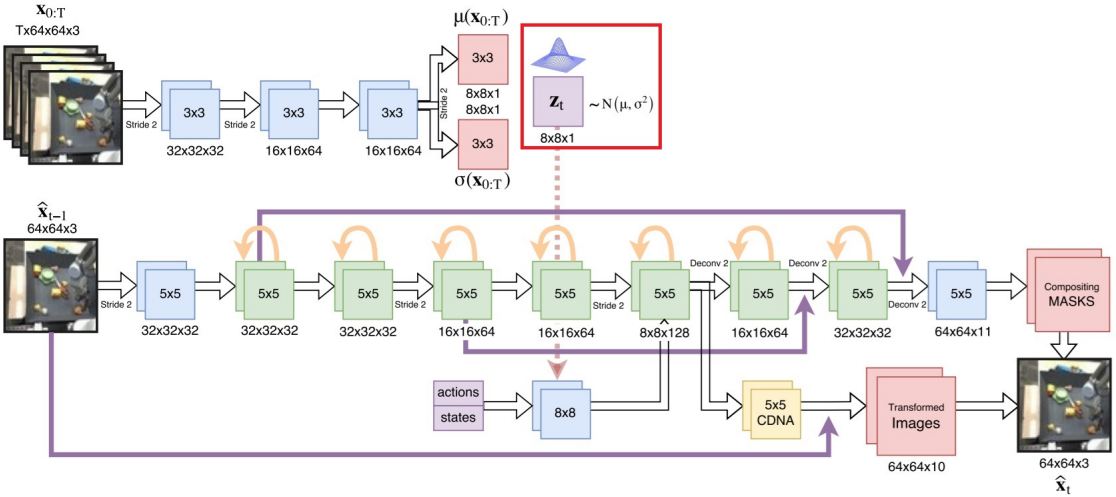


- The loss function is the negative ELBO w.r.t. $\log p_\theta(\vec{x}^{(1:T)})$, which can be expressed as:

$$
\begin{aligned}
& -\log p_\theta(\vec{x}^{(1:T)}) + \mathrm{KL}(q_\phi(\vec{z}^{(1:T)}|\vec{x}^{(1:T)})||p_\theta(\vec{z}^{(1:T)}|\vec{x}^{(1:T)})) \\
&= \sum_{t=1}^{T} \Bigg(-\int \Big(\log p_\theta(\vec{x}^{(t)}|\vec{z}^{(t)}, \vec{x}^{(1:(t-1))}) \Big) q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:T)}) d\vec{z}^{(t)} \\
&\qquad + \mathrm{KL}(q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:T)})||p(\vec{z}^{(t)})) \Bigg)
\end{aligned}
\tag{31}
$$

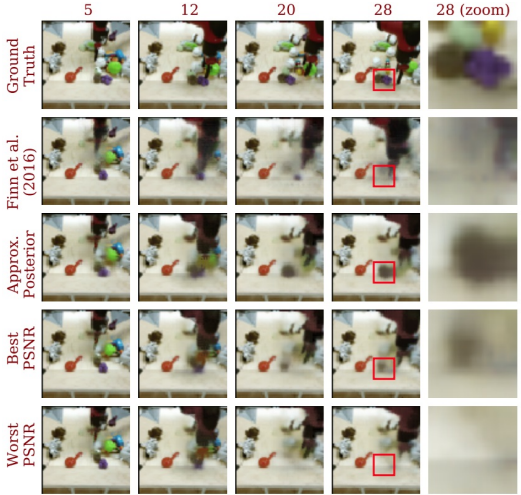
where $q_\phi(\vec{z}^{(1:T)}|\vec{x}^{(1:T)}) = \prod_{t=1}^{T} q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:T)})$.

SV2P with a Time-variant Latent Variable: Architecture



The figure is from Babaeizadeh et al. (2017).

SV2P with a Time-variant Latent Variable: Result



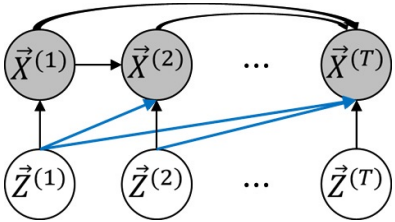
Images are edited from Babaeizadeh et al. (2017). PSNR stands for Peak Signal-to-Noise Ratio.

Stochastic Variational Generation

- In SV2P, the time-variant latent factors $\vec{Z}^{(t)}$ influence the generation of image frames at corresponding time steps $\vec{X}^{(t)}$.
- Due to significant information loss in propagating information from $\vec{X}^{(1:(t-1))}$ to $\vec{X}^{(t)}$, incorporating information from previous latent factors $\vec{Z}^{(1:(t-1))}$ directly may be beneficial to generate $\vec{X}^{(t)}$.
- Stochastic Video Generation (SVG) (Denton and Fergus, 2018) further models dependencies within time-variant latent factors. The authors proposed two approaches: (i) using a fixed prior and (ii) using a learned prior.[3]

[3] Here, 'learned' means that past images are used in designing prior distributions for the current time step.

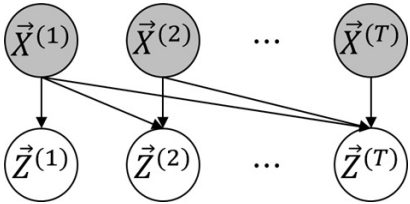
SVG with a Fixed Prior: Generation



- For (i) SVG with a fixed prior, the generative model can be expressed as:

$$p_\theta(\vec{z}^{(1:T)}, \vec{x}^{(1:T)}) = \prod_{t=1}^{T} p(\vec{z}^{(t)}) \prod_{t=1}^{T} p_\theta(\vec{x}^{(t)} | \vec{z}^{(1:t)}, \vec{x}^{(1:(t-1))}). \tag{32}$$

SVG with a Fixed Prior: Inference



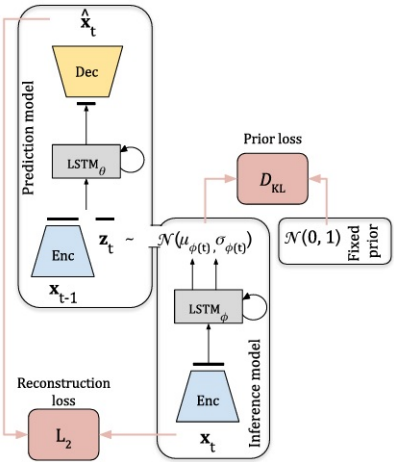
- The time-variant $\vec{Z}^{(t)}$ is inferred using all the image frames up to the time step t, $\vec{X}^{(1:t)}$.
- The inference network can be formulated as $q_\phi(\vec{z}^{(1:T)}|\vec{x}^{(1:T)}) = \prod_{t=1}^{T} q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:t)})$. Similar to p_θ, the parameter ϕ is shared across different time steps, though the inputs vary.

SVG with a Fixed Prior: Loss Function

- The loss function is the negative ELBO w.r.t. $\log p_\theta(\vec{x}^{(1:T)})$:

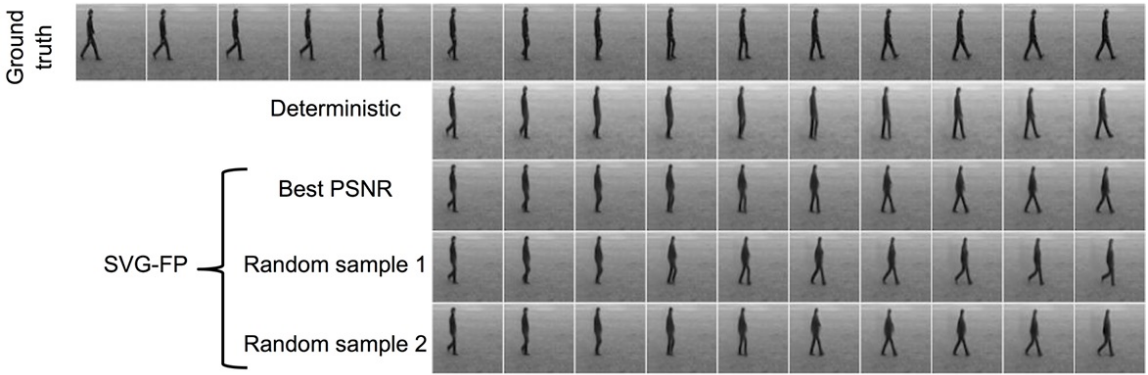
$$
\begin{aligned}
& -\log p_\theta(\vec{x}^{(1:T)}) + \mathrm{KL}(q_\phi(\vec{z}^{(1:T)}|\vec{x}^{(1:T)})||p_\theta(\vec{z}^{(1:T)}|\vec{x}^{(1:T)})) \\
&= \sum_{t=1}^{T} \Bigg(- \int \Big(\log p_\theta(\vec{x}^{(t)}|\vec{z}^{(1:t)}, \vec{x}^{(1:(t-1))}) \Big) \prod_{t'=1}^{t} q_\phi(\vec{z}^{(t')}|\vec{x}^{(1:t')}) d\vec{z}^{(t')} \\
&\qquad + \mathrm{KL}(q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:t)})||p(\vec{z}^{(t)})) \Bigg).
\end{aligned}
\tag{33}
$$

SVG with a Fixed Prior: Architecture



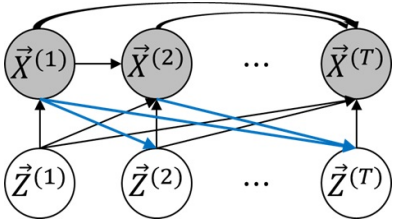
- $p(\vec{z}^{(t)})$: $\vec{z}^{(t)}$ is sampled from multivariate Gaussian.
- $p_\theta(\vec{x}^{(t)}|\vec{z}^{(1:t)}, \vec{x}^{(1:(t-1))})$: LSTM$_\theta$ sequentially inputs $\vec{x}^{(t-1)}$ and $\vec{z}^{(t)}$ to propagate information from $\vec{x}^{(1:(t-1))}$.
- $q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:t)})$: LSTM$_\phi$ sequentially inputs $\vec{x}^{(t)}$ to propagate information from $\vec{x}^{(1:t)}$.

SVG with a Fixed Prior: Result



Images are edited from Denton and Fergus (2018).

SVG with a Learned Prior: Generation



- For (ii), using a learned prior, SVG with a learned prior introduces prior networks $p_\theta(\vec{z}^{(t)}|\vec{x}^{(1:(t-1))})$ to reflect information from past image frames into the distribution of latent variable for future frames.
- Note that $\vec{Z}^{(2)}, \ldots,$ and $\vec{Z}^{(T)}$ are no longer parent nodes. The model distribution can be expressed as:

$$p_\theta(\vec{z}^{(1:T)}, \vec{x}^{(1:T)}) = \prod_{t=1}^{T} p_\theta(\vec{z}^{(t)}|\vec{x}^{(1:(t-1))}) p_\theta(\vec{x}^{(t)}|\vec{z}^{(1:t)}, \vec{x}^{(1:(t-1))}) \tag{34}$$

SVG with a Learned Prior: Loss Function

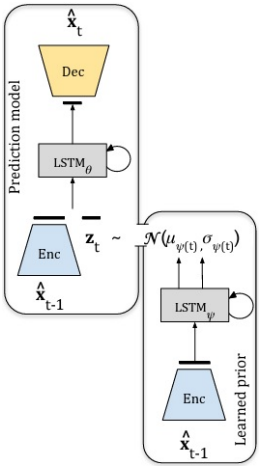
- The inference network is the same as in SVG with a fixed prior: $q_\phi(\vec{z}^{(1:T)}|\vec{x}^{(1:T)}) = \prod_{t=1}^{T} q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:t)})$.
- The loss function was intuitively modified from that of SVG with a fixed prior:

$$\sum_{t=1}^{T}\Bigg(-\int \Big(\log p_\theta(\vec{x}^{(t)}|\vec{z}^{(1:t)}, \vec{x}^{(1:(t-1))})\Big) \prod_{t'=1}^{t} q_\phi(\vec{z}^{(t')}|\vec{x}^{(1:t')}) d\vec{z}^{(t')} + \mathrm{KL}(q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:t)})||p_\theta(\vec{z}^{(t)}|\vec{x}^{(1:(t-1))}))\Bigg). \tag{35}$$

- Kim et al. (2022) showed that this loss represents a negative ELBO with respect to $\log p_\theta(\vec{x}^{(1:T)})$. Furthermore, they showed that minimizing this loss is equivalent to minimizing an upper bound of:

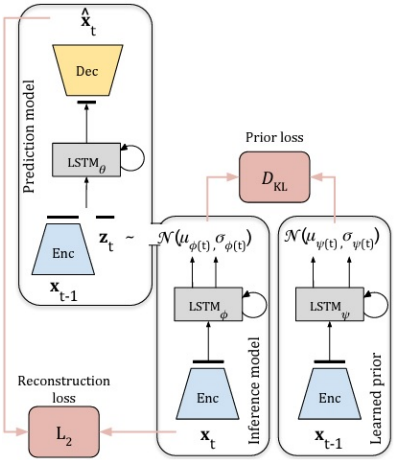
$$\sum_{t=1}^{T} \mathrm{KL}\left(p_\theta(\vec{x}^{(t)}|\vec{x}^{(1:(t-1))}) p(\vec{x}^{(1:(t-1))}) \,||\, p(\vec{x}^{(1:t)})\right). \tag{36}$$

SVG with a Learned Prior: Generator Architecture



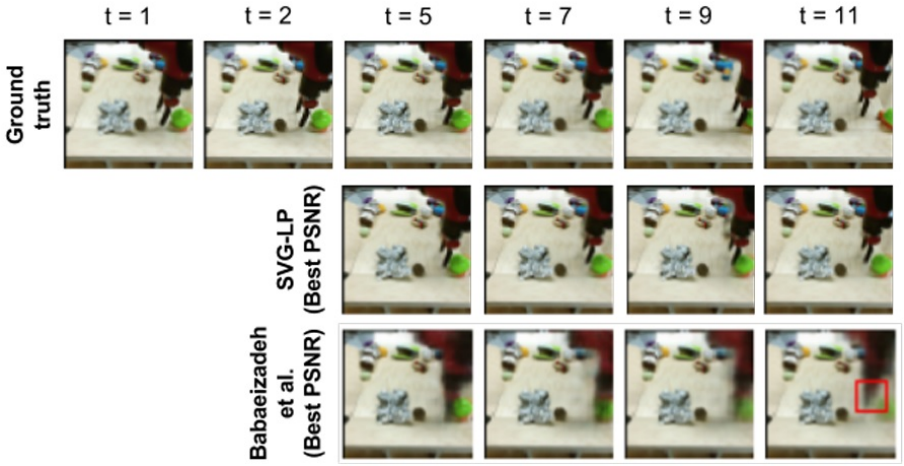
- $p_\theta(\vec{z}^{(t)}|\vec{x}^{(1:(t-1))})$: $\vec{z}^{(t)}$ is sampled from a Gaussian distribution whose means and stds are functions of $\vec{x}^{(1:(t-1))}$.
- $p_\theta(\vec{x}^{(t)}|\vec{z}^{(1:t)}, \vec{x}^{(1:(t-1))})$: LSTM$_\theta$ sequentially inputs $\vec{x}^{(t-1)}$ and $\vec{z}^{(t)}$ to propagate information from $\vec{x}^{(1:(t-1))}$.

SVG with a Learned Prior: Training



- $q_\phi(\vec{z}^{(t)}|\vec{x}^{(1:t)})$: LSTM$_\phi$ sequentially inputs $\vec{x}^{(t)}$ to propagate information from $\vec{x}^{(1:t)}$.

SVG with a Learned Prior: Result



Images are edited from Denton and Fergus (2018).

Outline

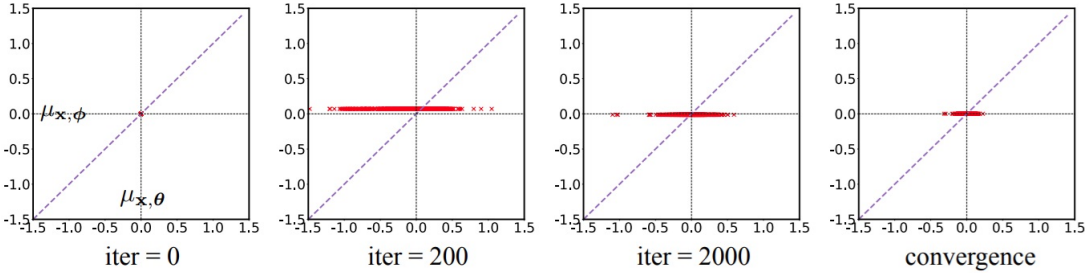
Posterior Collapse

- Again, the loss function of VAEs, the negative ELBO can be expressed as:

$$-\int \Big(\log p_\theta(\vec{x}|\vec{z})\Big) q_\phi(\vec{z}|\vec{x}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z})). \tag{37}$$

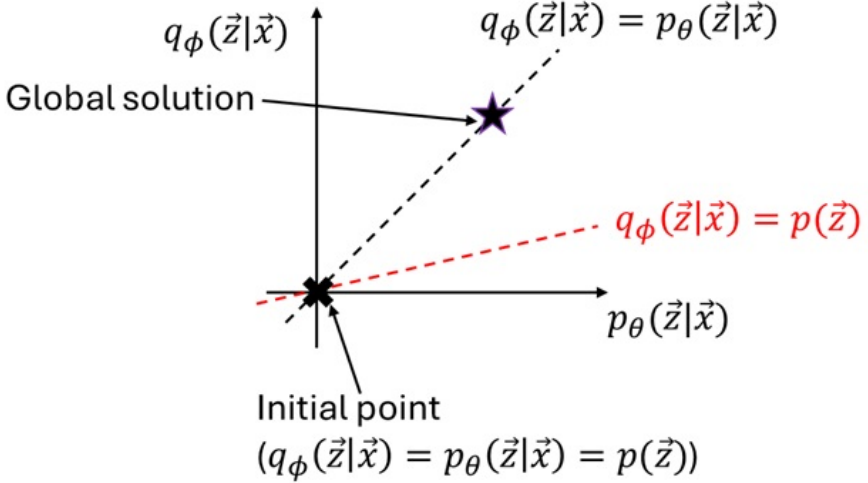
- In the early stages of training, $\mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z})) \approx 0$, with initial random weights, ignores $\vec{x}$. Therefore, it reduces to the multivariate standard Gaussian distribution.
- During the training of VAEs, the KL penalty sometimes dominates the whole loss function. In such cases, q_ϕ remains close to $p(\vec{z})$, continuing to ignore $\vec{x}$ even during the final inference.
- This phenomenon is referred to as *posterior collapse* (He et al., 2019), a chronic optimization issue in VAEs.

Posterior Collapse

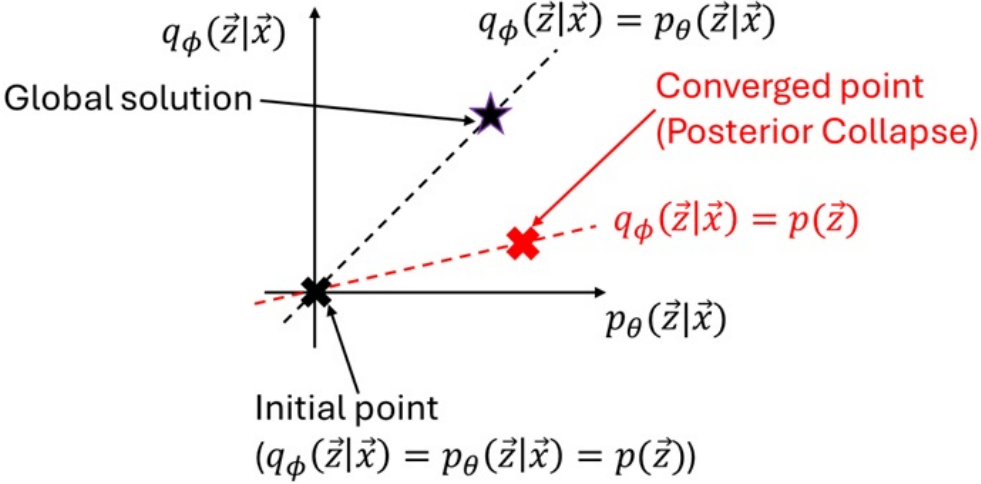


The images are edited from He et al. (2019).

Posterior Collapse



Posterior Collapse



Posterior Collapse

- Dai et al. (2020) showed that the posterior collapse can occur in a common setting where $\vec{X}|\vec{Z} = \vec{z} \sim N(\vec{\mu}_{\vec{X}|\vec{Z}}(\vec{z}), \gamma I)$:

> **Proposition 2** *For any well-behaved VAE with arbitrary non-degenerate decoder* $\boldsymbol{\mu}_x(\boldsymbol{z}; \theta = \tilde{\theta})$*, there will always exist a* $\gamma' < \infty$ *such that the trivial solution* $\boldsymbol{\mu}_x(\boldsymbol{z}; \theta \neq \tilde{\theta}) = \bar{\boldsymbol{x}}$ *and* $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = p(\boldsymbol{z})$ *will have lower cost.*

Remedies for Posterior Collapse: Aggressively Updating Inference Network

Algorithm 1 VAE training with controlled aggressive inference network optimization.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
$aggressive \leftarrow$ TRUE
repeat
  if $aggressive$ **then**
    repeat ▷ [aggressive updates]
      $\mathbf{X} \leftarrow$ Random data minibatch
      Compute gradients $\boldsymbol{g}_{\boldsymbol{\phi}} \leftarrow \nabla_{\boldsymbol{\phi}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi})$
      Update $\boldsymbol{\phi}$ using gradients $\boldsymbol{g}_{\boldsymbol{\phi}}$
    until convergence
    $\mathbf{X} \leftarrow$ Random data minibatch
    Compute gradients $\boldsymbol{g}_{\boldsymbol{\theta}} \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi})$
    Update $\boldsymbol{\theta}$ using gradients $\boldsymbol{g}_{\boldsymbol{\theta}}$
  else ▷ [basic VAE training]
    $\mathbf{X} \leftarrow$ Random data minibatch
    Compute gradients $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\phi}} \leftarrow \nabla_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi})$
    Update $\boldsymbol{\theta}, \boldsymbol{\phi}$ using $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\phi}}$
  end if
  Update $aggressive$ as discussed in Section 4.2
until convergence

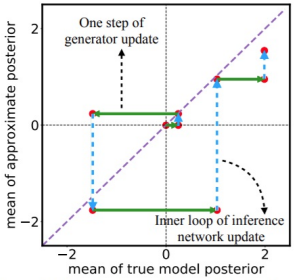
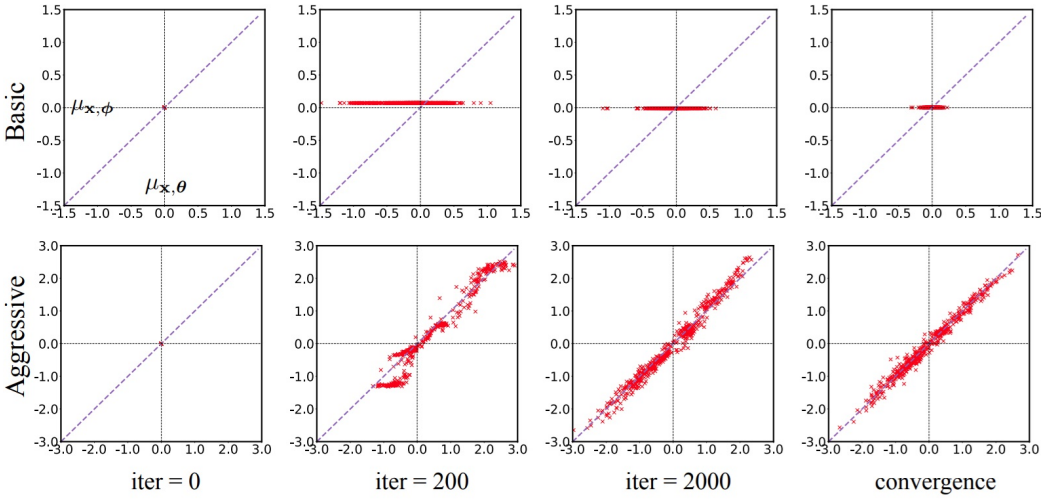


Figure 3: Trajectory of one data instance on the posterior mean space with our aggressive training procedure. Horizontal arrow denotes one step of generator update, and vertical arrow denotes the inner loop of inference network update. We note that the approximate posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ takes an aggressive step to catch up to the model posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$.

The images are edited from He et al. (2019).

Remedies for Posterior Collapse: Aggressively Updating Inference Network



The images are from He et al. (2019).

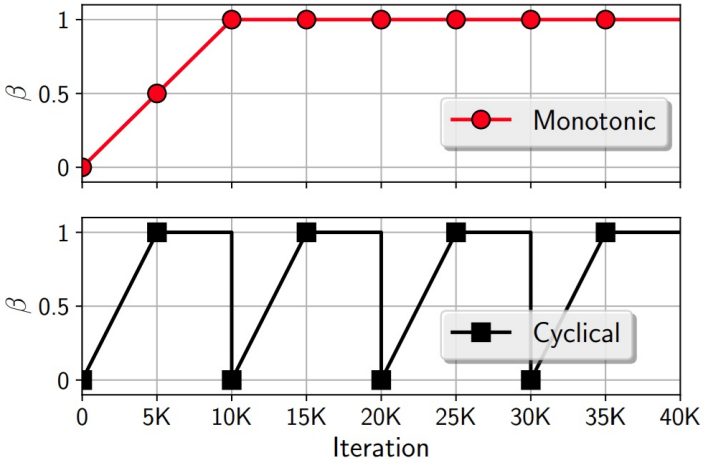
Remedies for Posterior Collapse: KL Annealing

- The KL annealing (Bowman et al., 2015) is a technique that gradually increases the coefficient of the KL penalty term from zero.
- The loss function can be expressed as:

$$-\int \Big(\log p_\theta(\vec{x}|\vec{z})\Big) q_\phi(\vec{z}|\vec{x}) d\vec{z} + \beta^{(t)}\, \mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z})). \tag{38}$$

  where $\beta^{(t)}$ represents a dynamic coefficient. For example, we can set $\beta^{(0)} = 0$ and linearly increase the beta to $\beta^{(t)} = 1$ for $t \geq T/10$, where T is the total number of iterations.
- When $\beta^{(t)}$ is a time-invariant constant, it reduces to a β-VAE (Higgins et al., 2017).

Remedies for Posterior Collapse: Cyclical Annealing Schedule



The images are from Fu et al. (2019).

Outline

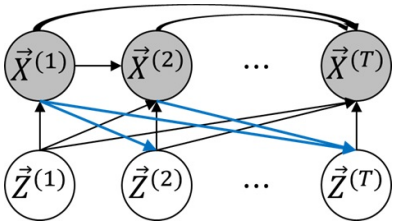
More Flexible Prior

- In the loss function of VAEs,

$$-\int \Big(\log p_\theta(\vec{x}|\vec{z}) \Big) q_\phi(\vec{z}|\vec{x}) d\vec{z} + \mathrm{KL}(q_\phi(\vec{z}|\vec{x})||p(\vec{z})), \tag{39}$$

  the fast computation of the KL penalty term is based on the closed-form expression of KL divergence between two multivariate Gaussian distributions.
- Although nonlinear transformations of Gaussian distributions can express arbitrary distributions of continuous random vectors, considering other prior distributions such as a Gaussian mixture may enhance the interpretability of the inference results.

More Flexible Prior: Learned Prior



- The learned prior in SVG is an example of flexible prior distribution.
- The $p_\theta(\vec{z}^{(t)}|\vec{x}^{(1:(t-1))})$ is still Gaussian, but its mean and covariance can gain flexibility by reflecting information from past frames $\vec{x}^{(1:(t-1))}$.
- Similarly, conditional VAEs (Khemakhem et al., 2020) that use an additional covariate to model the prior distribution are also examples.

More Flexible Prior: Wasserstein Penalty

- Gu et al. (2018) proposed to use the Wasserstein distance instead of the KL-divergence:

$$-\int \Big(\log p_\theta(\vec{x}|\vec{z})\Big) q_\phi(\vec{z}|\vec{x}) d\vec{z} + W(q_\phi(\vec{z}|\vec{x})||p_\theta(\vec{z})), \tag{40}$$

  where $W(\cdot,\cdot)$ represents Wasserstein distances.
- There are several prominent numerical algorithms to approximate the Wasserstein distance between arbitrary distributions, allowing us to consider more flexible prior distributions.
- Although this intuitive modification makes the loss is no longer the negative ELBO. However, this approach has shown remarkable performance.

More Flexible Prior: Variational Mixture of Posterior Prior

- When $q_\phi(\vec{z}|\vec{x}) \approx p_\theta(\vec{z}|\vec{x})$ and $p_\theta(\vec{x}) \approx p(\vec{x})$,

$$\begin{aligned} q_\phi(\vec{z}) &:= \int q_\phi(\vec{z}|\vec{x}) p(\vec{x}) d\vec{x} \\ &\approx \int p_\theta(\vec{z}|\vec{x}) p(\vec{x}) d\vec{x} \\ &\approx \int p_\theta(\vec{z}|\vec{x}) p_\theta(\vec{x}) d\vec{x} \approx p(\vec{z}). \end{aligned} \tag{41}$$

  Thus, the empirical estimate of $q_\phi(\vec{z})$, $n^{-1}\sum_{i=1}^n q_\phi(\vec{z}|\vec{x}_i)$, can be viewed as an implicit optimal prior. This prior, termed an *aggregated posterior*, is a mixture of Gaussians with each component modeled by $q_\phi(\vec{z}|\vec{x})$. The corresponding KL penalty term can be calculated via the Monte Carlo method.

- See Tomczak and Welling (2018) and Takahashi et al. (2019) for further details.

Outline

ICA vs. Auto-regressive Model vs. Energy-based Model vs. VAE

- Independent Component Analysis: $\mathbb{P}_\theta(\vec{x}) = \int \delta_{\vec{x}}(W\vec{z})d\mathbb{P}(\vec{z})$
- Auto-regressive Model: $p_\theta(\vec{x}) = p_\theta(x_1)p_\theta(x_2|x_1)\cdots p_\theta(x_p|x_1,\ldots,x_{p-1})$
- Energy-based Model: $p_\theta(\vec{x}) = \int C(\theta)^{-1}\exp(-E_\theta(\vec{z},\vec{x}))d\vec{z}$.
- Variational Autoencoder: $p_\theta(\vec{x}) = \int p_\theta(\vec{x}|\vec{z})p(\vec{z})d\vec{z}$ where $p_\theta(\vec{x}|\vec{z}) = p(\vec{x};\vec{\mu}_{\vec{X}|\vec{Z}}(\vec{z}),\vec{\Sigma}_{\vec{X}|\vec{Z}}(\vec{z}))$

  Q: What would be their strong and weak points?

References I

Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., and Levine, S. (2017). Stochastic variational video prediction. *arXiv preprint arXiv:1710.11252*.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2006). Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19.

Bishop, C. M. (2006). Pattern recognition and machine learning. *Springer google schola*, 2:1122–1128.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. (2015). Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

Chen, K., Reiman, E. M., Huan, Z., Caselli, R. J., Bandy, D., Ayutyanont, N., and Alexander, G. E. (2009). Linking functional and structural brain images with multivariate network analyses: a novel application of the partial least square method. *Neuroimage*, 47(2):602–610.

References II

Dai, B., Wang, Z., and Wipf, D. (2020). The usual suspects? reassessing blame for vae posterior collapse. In *International conference on machine learning*, pages 2313–2322. PMLR.

Denton, E. and Fergus, R. (2018). Stochastic video generation with a learned prior. In *International conference on machine learning*, pages 1174–1183. PMLR.

Fu, H., Li, C., Liu, X., Gao, J., Celikyilmaz, A., and Carin, L. (2019). Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*.

Gong, C., Jing, C., Chen, X., Pun, C. M., Huang, G., Saha, A., Nieuwoudt, M., Li, H.-X., Hu, Y., and Wang, S. (2023). Generative ai for brain image computing and brain network computing: a review. *Frontiers in Neuroscience*, 17:1203104.

Gu, X., Cho, K., Ha, J.-W., and Kim, S. (2018). Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. *arXiv preprint arXiv:1805.12352*.

References III

He, J., Spokoyny, D., Neubig, G., and Berg-Kirkpatrick, T. (2019). Lagging inference networks and posterior collapse in variational autoencoders. *arXiv preprint arXiv:1901.05534*.

Higgins, I., Matthey, L., Pal, A., Burgess, C. P., Glorot, X., Botvinick, M. M., Mohamed, S., and Lerchner, A. (2017). beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.

Kalchbrenner, N., Oord, A., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., and Kavukcuoglu, K. (2017). Video pixel networks. In *International Conference on Machine Learning*, pages 1771–1779. PMLR.

Kettenring, J. R. (1971). Canonical analysis of several sets of variables. *Biometrika*, 58(3):433–451.

References IV

Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. (2020). Variational autoencoders and nonlinear ica: A unifying framework. In *International conference on artificial intelligence and statistics*, pages 2207–2217. PMLR.

Kim, J.-H., Zhang, Y., Han, K., Wen, Z., Choi, M., and Liu, Z. (2021). Representation learning of resting state fmri with variational autoencoder. *NeuroImage*, 241:118423.

Kim, Y.-g., Lee, K., and Paik, M. C. (2022). Conditional wasserstein generator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7208–7219.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Kingma, D. P., Welling, M., et al. (2019). An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243.

References V

Lee, M. and Pavlovic, V. (2021). Private-shared disentangled multimodal vae for learning of latent representations. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 1692–1700.

Lock, E. F., Hoadley, K. A., Marron, J. S., and Nobel, A. B. (2013). Joint and individual variation explained (jive) for integrated analysis of multiple data types. *The annals of applied statistics*, 7(1):523.

Lotter, W., Kreiman, G., and Cox, D. (2016). Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*.

Plaut, E. (2018). From principal subspaces to principal components with linear autoencoders. *arXiv preprint arXiv:1804.10253*.

Suzuki, M., Nakayama, K., and Matsuo, Y. (2016). Joint multimodal learning with deep generative models. *arXiv preprint arXiv:1611.01891*.

References VI

Szeto, R., Sun, X., Lu, K., and Corso, J. J. (2019). A temporally-aware interpolation network for video frame inpainting. *IEEE transactions on pattern analysis and machine intelligence*, 42(5):1053–1068.

Takahashi, H., Iwata, T., Yamanaka, Y., Yamada, M., and Yagi, S. (2019). Variational autoencoder with implicit optimal priors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5066–5073.

Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B. (2017). Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*.

Tomczak, J. and Welling, M. (2018). Vae with a vampprior. In *International conference on artificial intelligence and statistics*, pages 1214–1223. PMLR.

Wu, M. and Goodman, N. (2018). Multimodal generative models for scalable weakly-supervised learning. *Advances in neural information processing systems*, 31.