Advanced Inference and Representation Learning

Methods in Variational Autoencoders

PhD Thesis

Ignacio Peis
Universidad Carlos III de Madrid
ipeis@tsc.uc3m.es

Advisors:
Antonio Artés Rodríguez
Pablo M. Olmos

22 September 2023

Outline

Contents

Introduction

What are Variational Autoencoders?

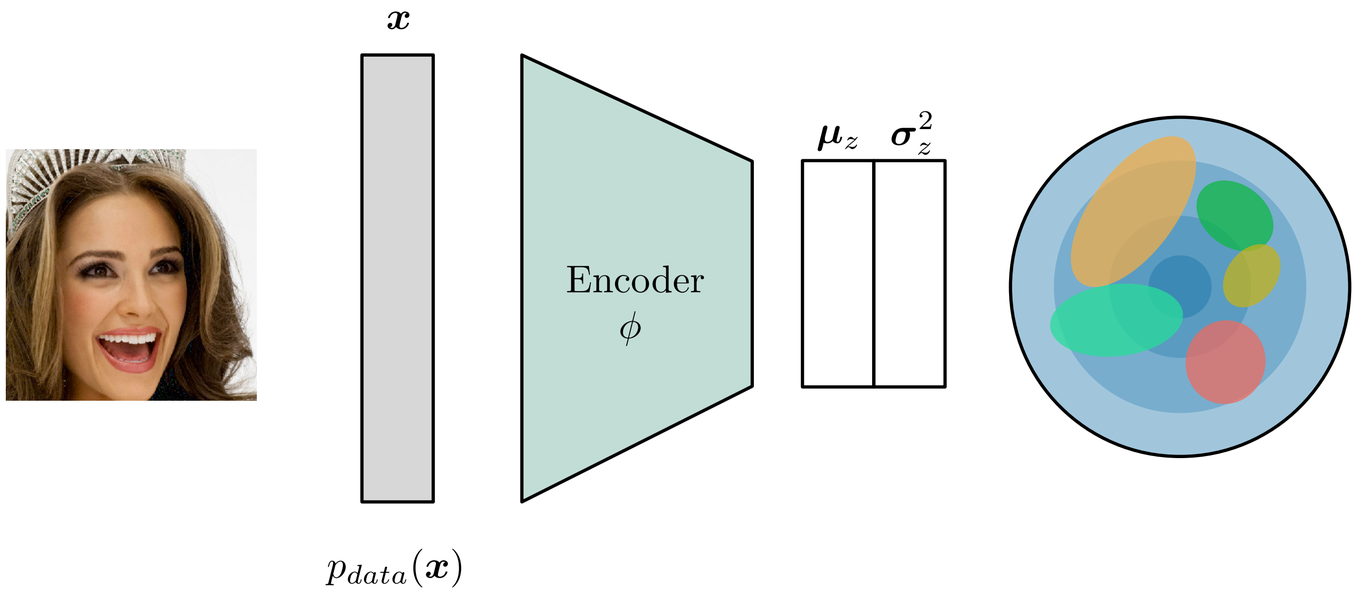
- **VAEs** [1] are a type of **Deep Generative Model** that learns to **encode** and **decode** data for generating similar data.

[1] (Kingma et al., 2013)

Introduction

What are Variational Autoencoders?

- **VAEs** [1] are a type of **Deep Generative Model** that learns to **encode** and **decode** data for generating similar data.

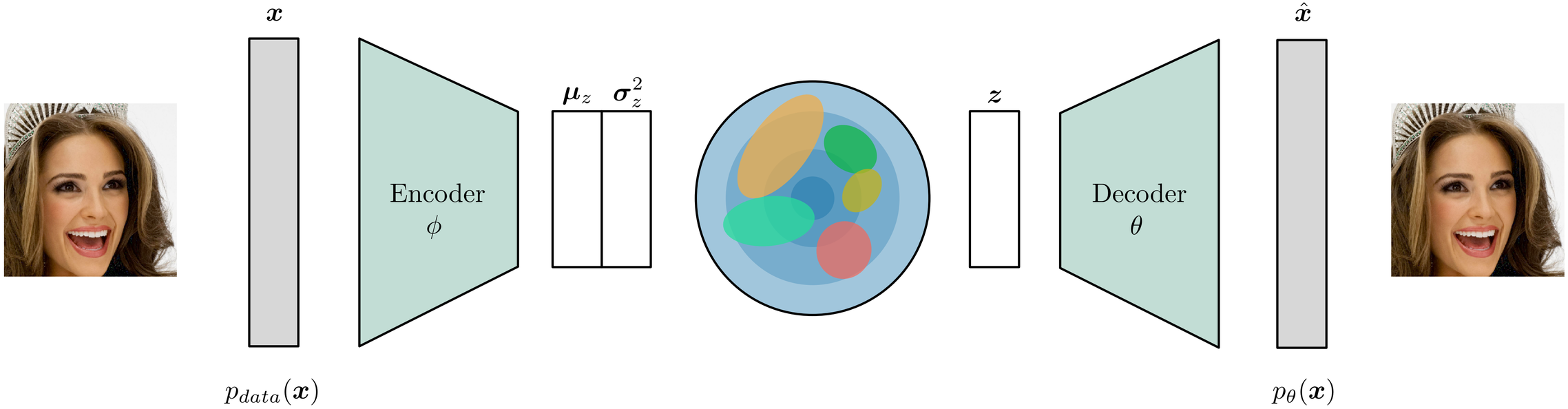


[1] (Kingma et al., 2013)

Introduction

What are Variational Autoencoders?

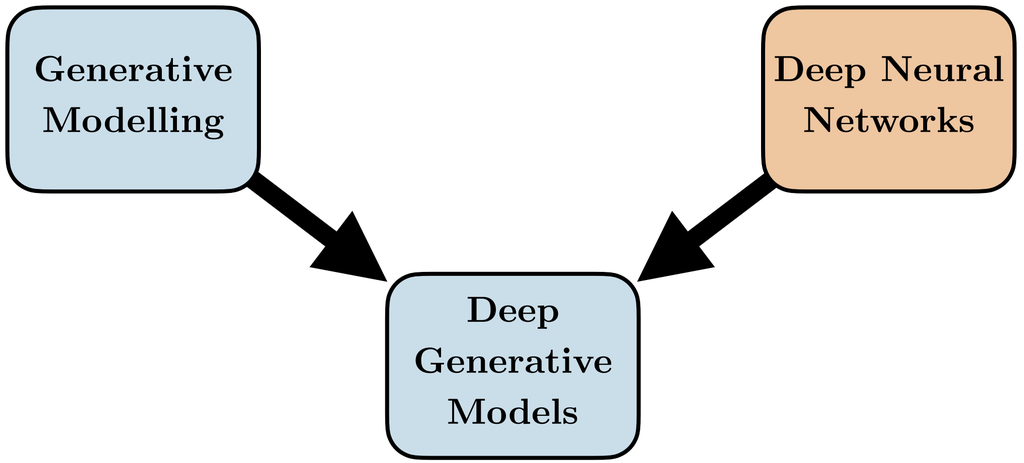
- **VAEs** [1] are a type of **Deep Generative Model** that learns to **encode** and **decode** data for generating similar data.



[1] (Kingma et al., 2013)

Introduction

What are Deep Generative Models?

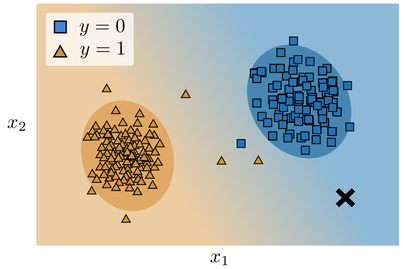


[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

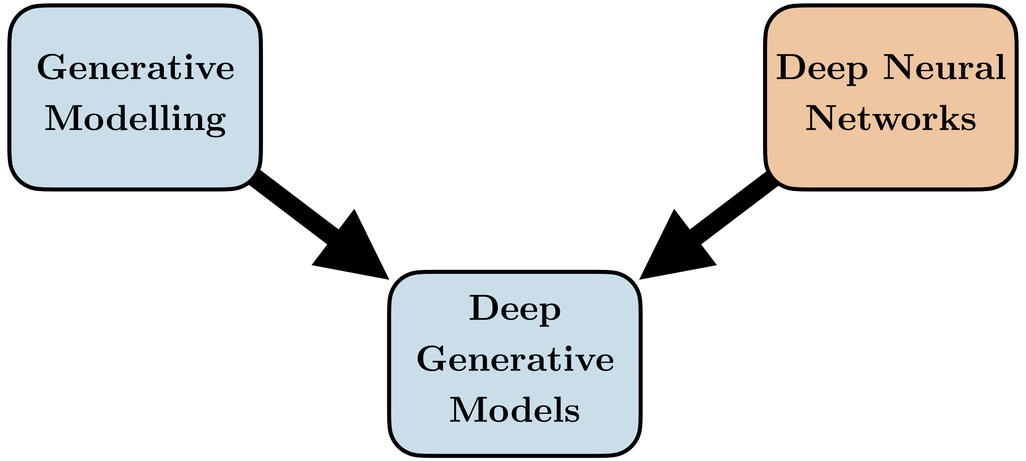
Introduction

What are Deep Generative Models?

$$p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$



- ✓ Capture linear/non-linear relationships between $\boldsymbol{x}$ and y.
- ✓ Quantify uncertainty.
- ✓ Learn data structure.

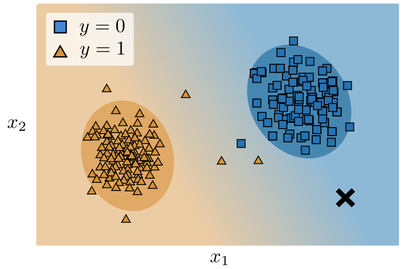


[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

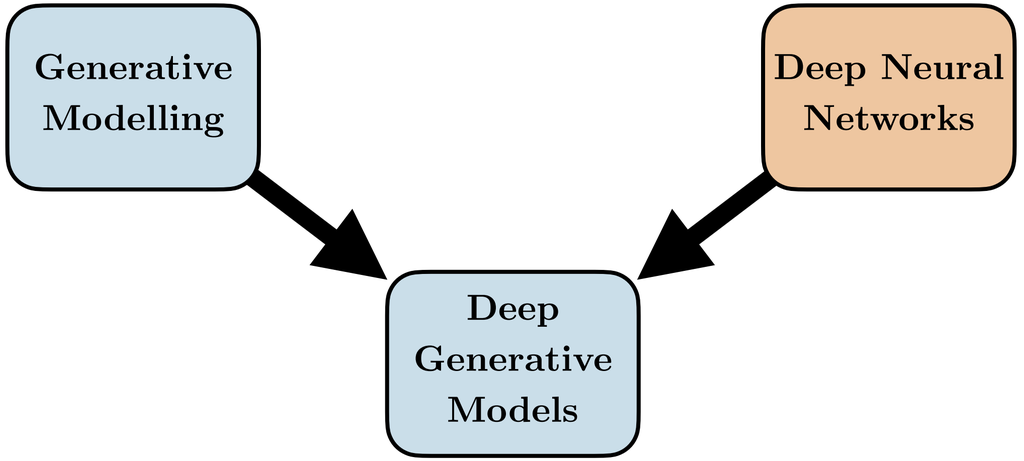
Introduction

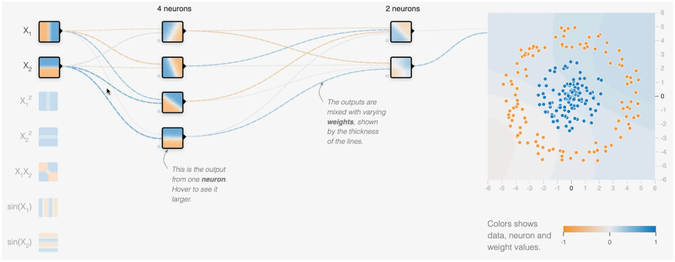
What are Deep Generative Models?

$$p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$



- ✓ Capture linear/non-linear relationships between $\boldsymbol{x}$ and y.
- ✓ Quantify uncertainty.
- ✓ Learn data structure.





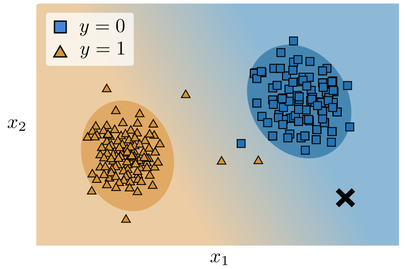
Multi Layer Perceptron (MLP)

[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

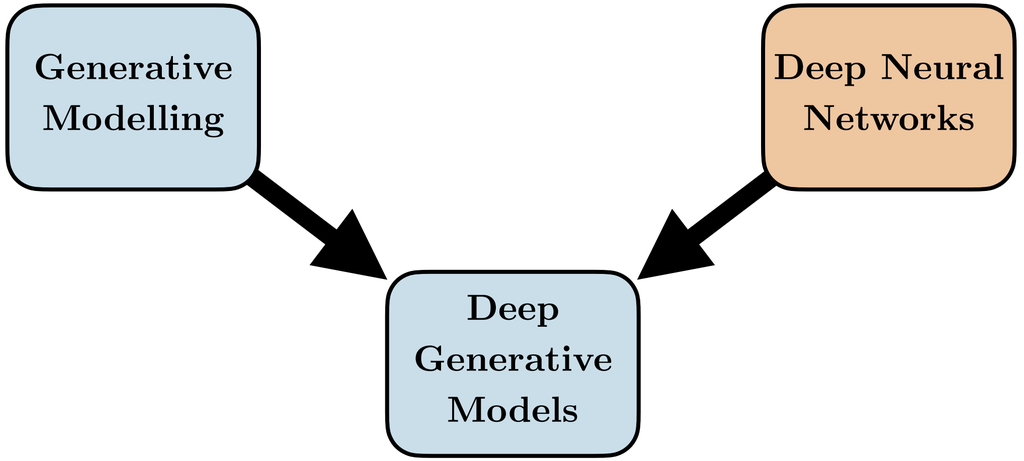
Introduction

What are Deep Generative Models?

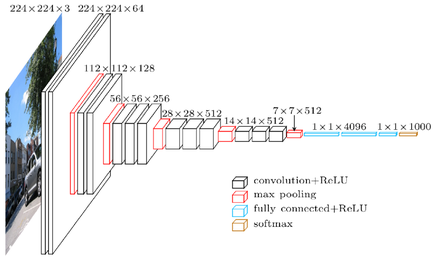
$$p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$



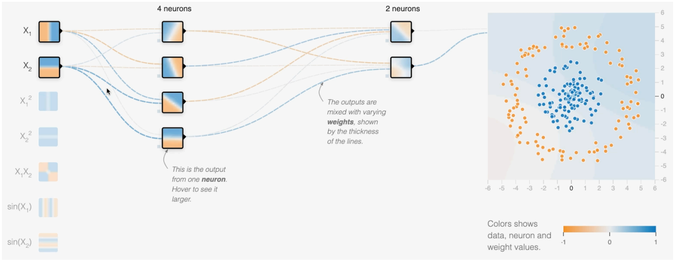
- ✓ Capture linear/non-linear relationships between $\boldsymbol{x}$ and y.
- ✓ Quantify uncertainty.
- ✓ Learn data structure.



Convolutional Neural Network (CNN)



The VGG16 network [22]



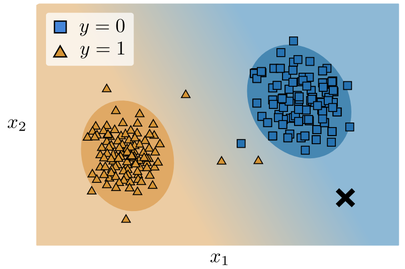
Multi Layer Perceptron (MLP)

[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

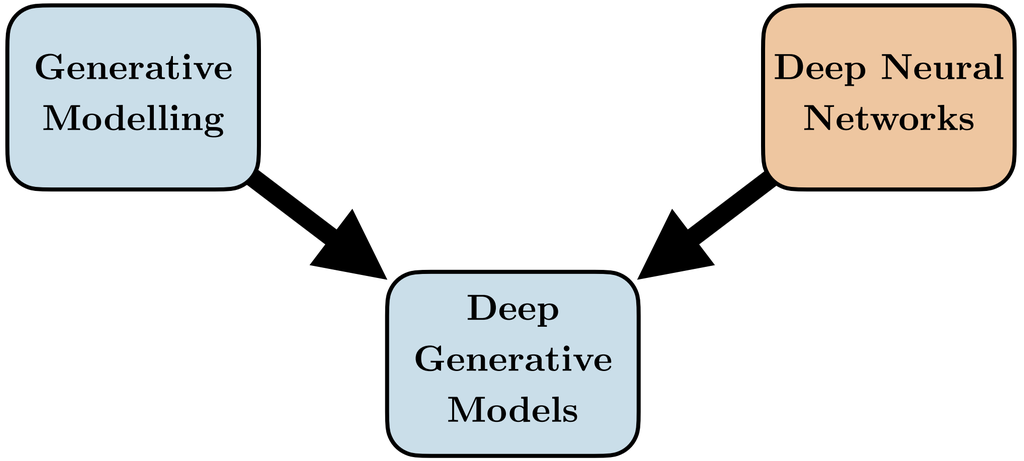
Introduction

What are Deep Generative Models?

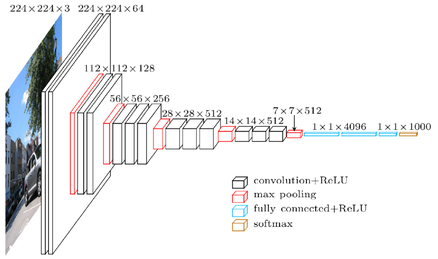
$$p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$



- ✓ Capture linear/non-linear relationships between $\boldsymbol{x}$ and y.
- ✓ Quantify uncertainty.
- ✓ Learn data structure.

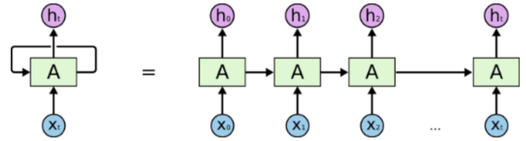


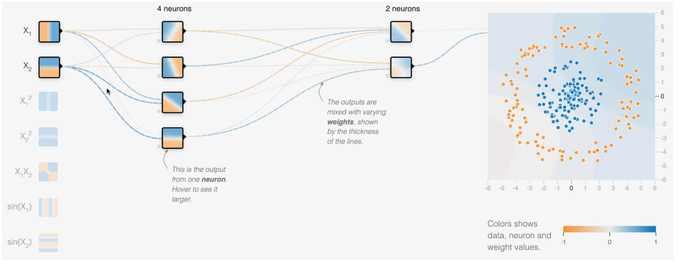
Convolutional Neural Network (CNN)



The VGG16 network [22]

Recurrent Neural Network (RNN)





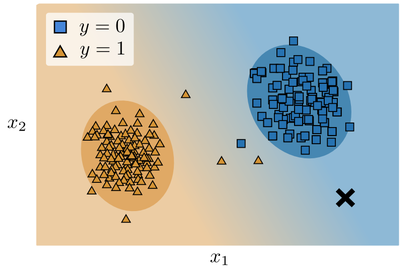
Multi Layer Perceptron (MLP)

[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

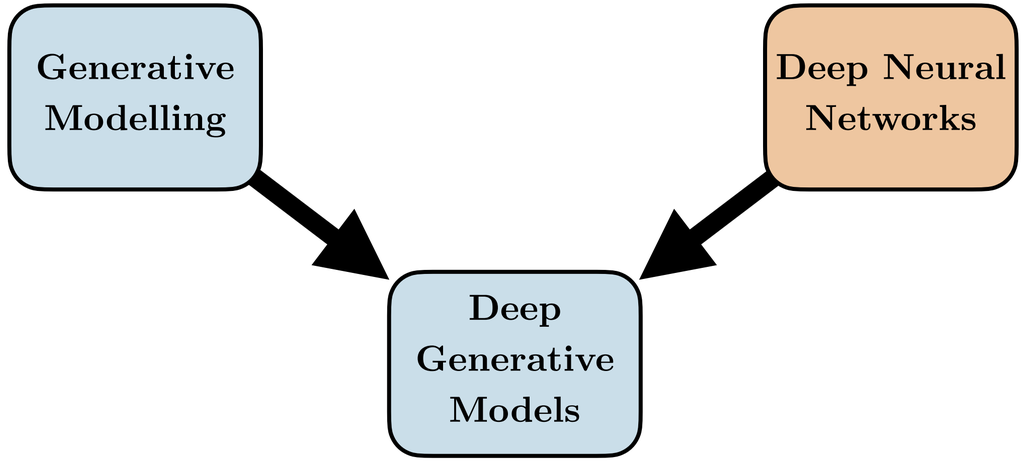
Introduction

What are Deep Generative Models?

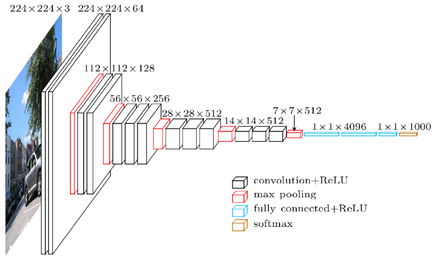
$$p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$



- ✓ Capture linear/non-linear relationships between $\boldsymbol{x}$ and y.
- ✓ Quantify uncertainty.
- ✓ Learn data structure.

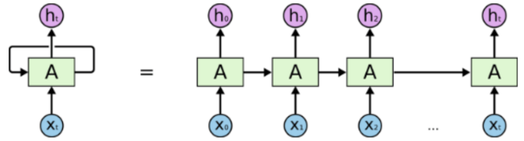


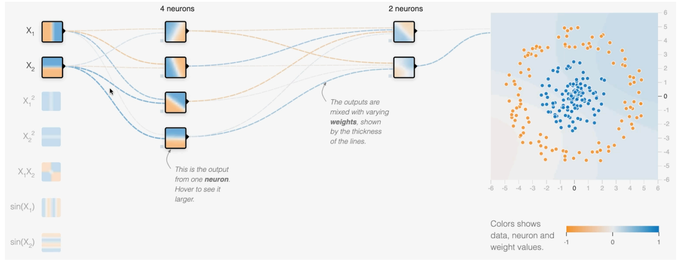
Convolutional Neural Network (CNN)



The VGG16 network [22]

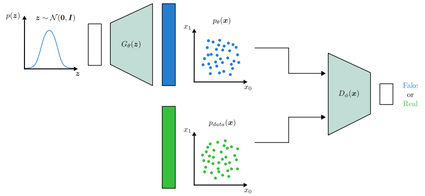
Recurrent Neural Network (RNN)



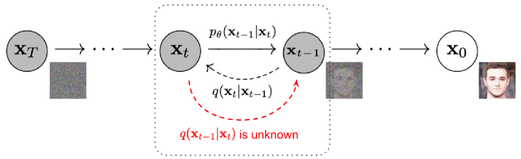


Multi Layer Perceptron (MLP)

Generative Adversarial Network (GAN)



Diffusion Models [25]

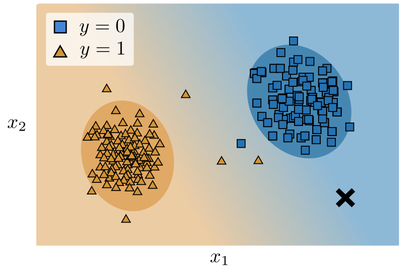


[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

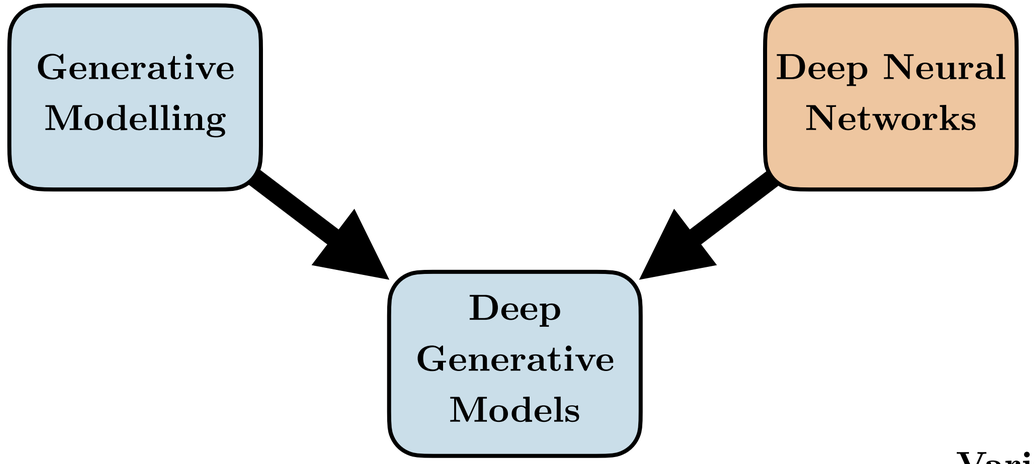
Introduction

What are Deep Generative Models?

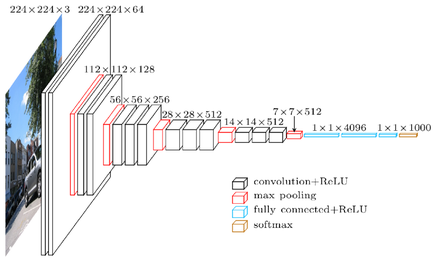
$$p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$



- ✓ Capture linear/non-linear relationships between $\boldsymbol{x}$ and y.
- ✓ Quantify uncertainty.
- ✓ Learn data structure.

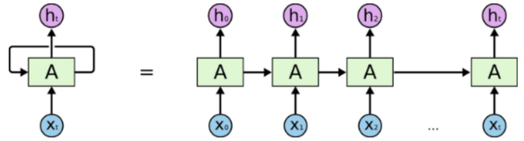


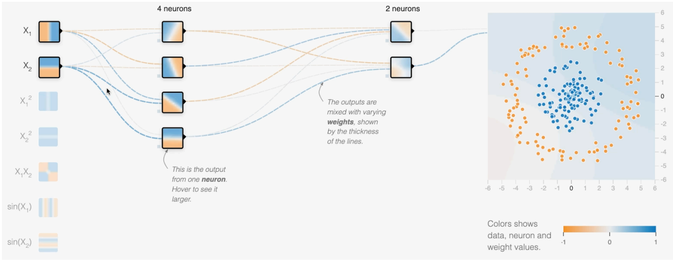
Convolutional Neural Network (CNN)



The VGG16 network [22]

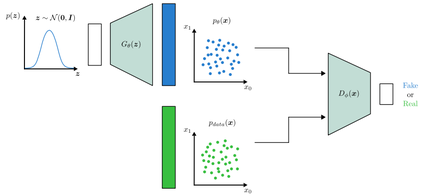
Recurrent Neural Network (RNN)



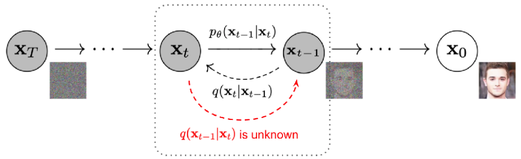


Multi Layer Perceptron (MLP)

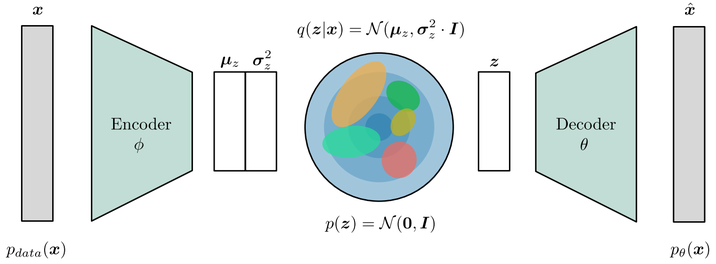
Generative Adversarial Network (GAN)



Diffusion Models [25]



Variational Autoencoder (VAE)



[21] (Goodfellow et al., 2016) [22] (Simonyan and Zisserman, 2014) [23] (Peis et al., 2019)

Introduction

Why VAEs

Model	Stable training	Exact likelihood	Fast sampling	Easy reconstruction	Invertible	Compression
Autoregressive	✓	✓	✗	–	–	✗
Flows	✓	✓	✓	✓	✓	✗
Score-based	✓	✓	✗	–	–	✗
Energy-based	✗	✗	✗	–	–	✗
VAEs	✓	✗	✓	✓	✗	✓
DDGMs	✓	✗	✗	✗	✗	✗
GANs	✗	✗	✓	✗	✗	✗

- VAEs excel in the **inference** task.

> ▸ Inference is the task of making predictions or determinations about underlying or latent structures in data, based on a given model and observed evidence.

Introduction

Why VAEs

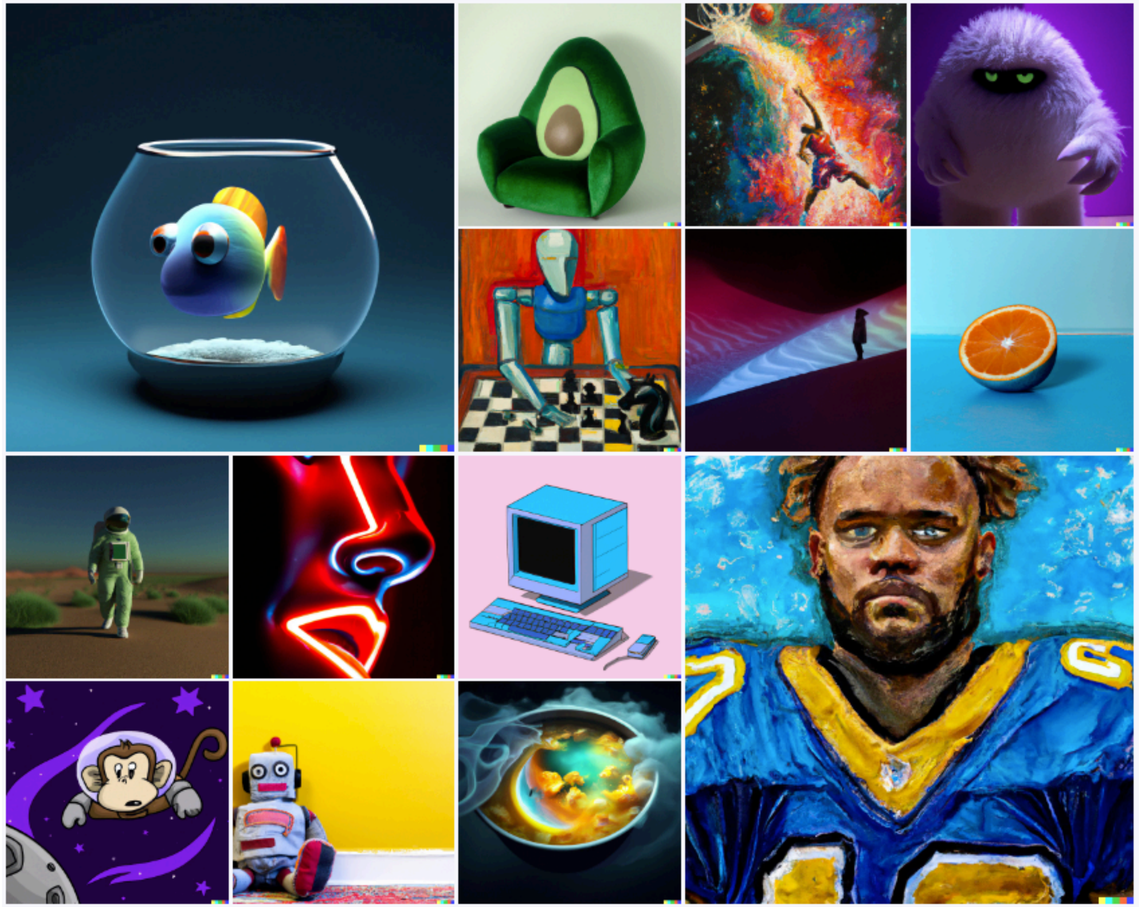
Model	Stable training	Exact likelihood	Fast sampling	Easy reconstruction	Invertible	Compression
Autoregressive	✓	✓	✗	–	–	✗
Flows	✓	✓	✓	✓	✓	✗
Score-based	✓	✓	✗	–	–	✗
Energy-based	✗	✗	✗	–	–	✗
VAEs	✓	✗	✓	✓	✗	✓
DDGMs	✓	✗	✗	✗	✗	✗
GANs	✗	✗	✓	✗	✗	✗

- VAEs excel in the **inference** task.

> ▸ Inference is the task of making predictions or determinations about underlying or latent structures in data, based on a given model and observed evidence.

Introduction

Why inference? Why VAEs?



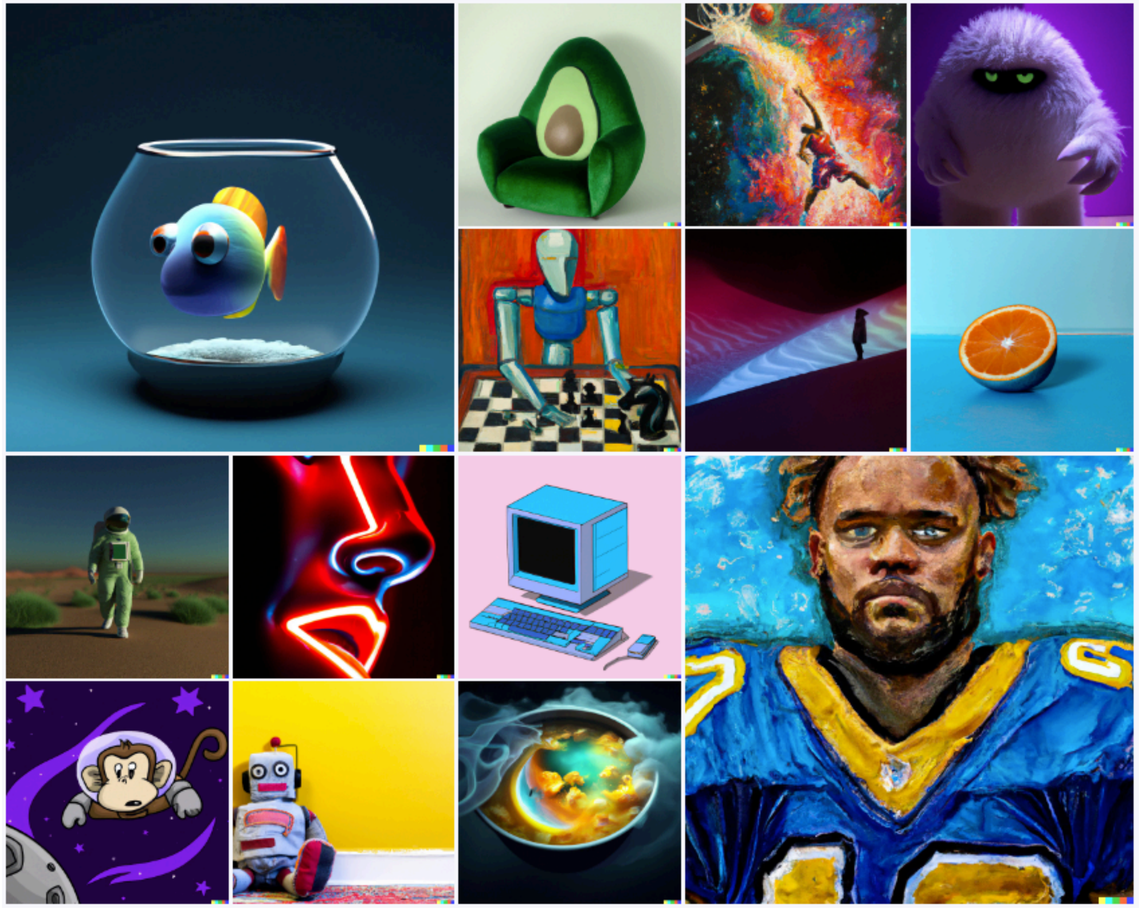
https://openai.com/dall-e-2

[2] (Ramesh et al., 2022)

Introduction

Why inference? Why VAEs?

- VAEs are applied in innumerable tasks:
  - Generation, completion, edition, manipulation [2]



https://openai.com/dall-e-2

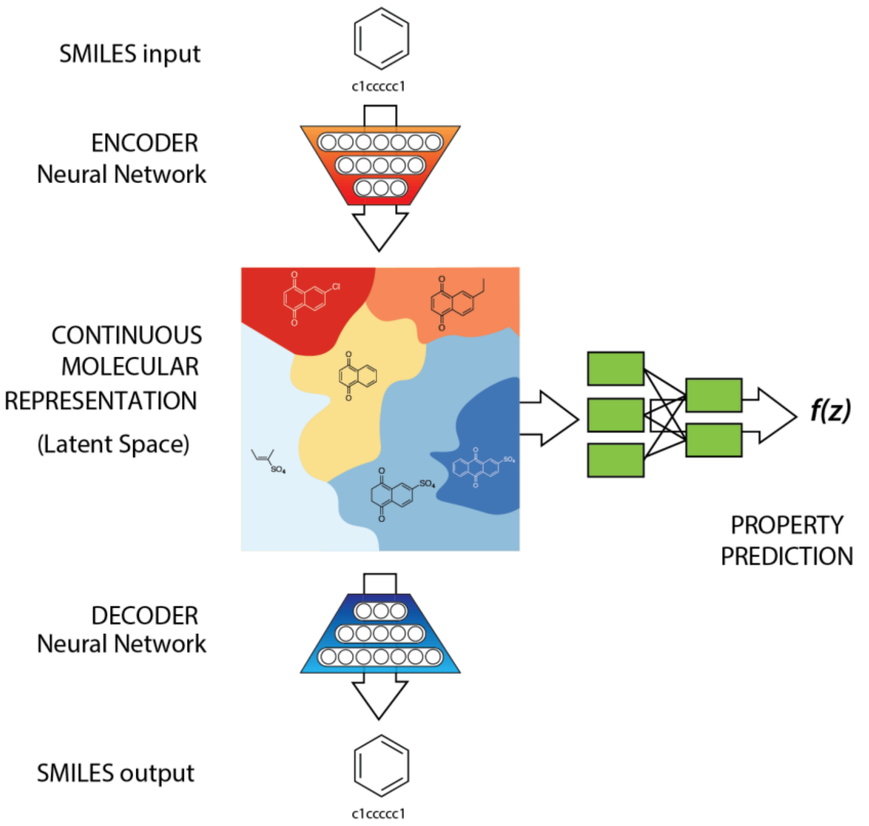
[2] (Ramesh et al., 2022)

Introduction

Why inference? Why VAEs?

- VAEs are applied in innumerable tasks:
  - Generation, completion, edition, manipulation [2]
  - Drug Discovery [3]

[3]



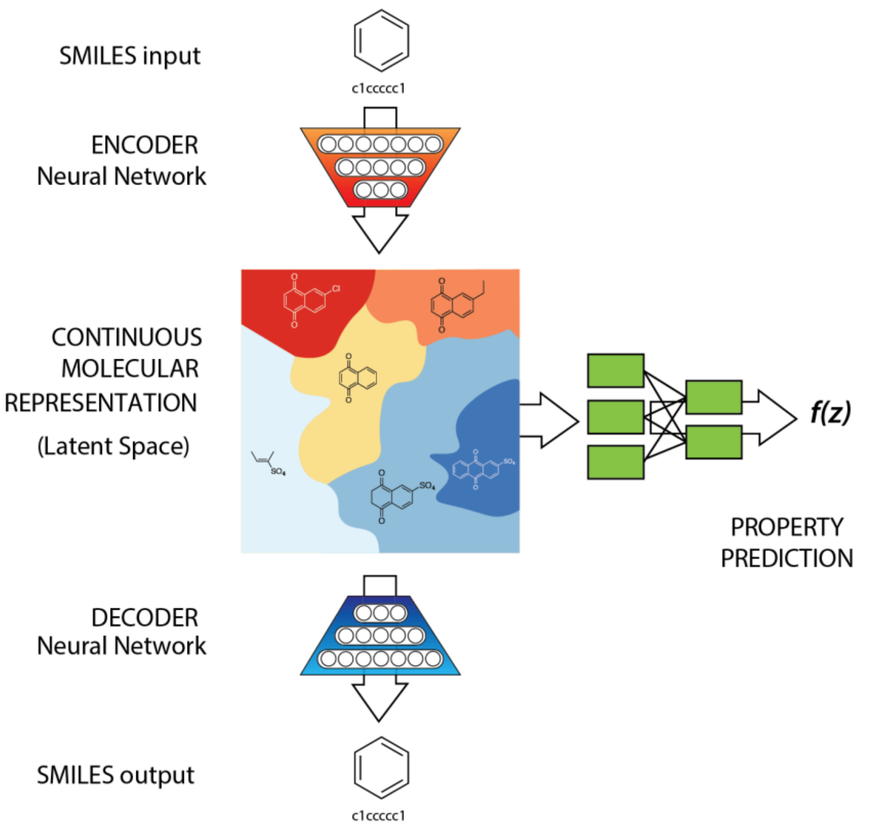
[2] (Ramesh et al., 2022) [3] (Gómez-Bombarelli et al., 2018)

Introduction

Why inference? Why VAEs?

- VAEs are applied in innumerable tasks:
  - Generation, completion, edition, manipulation [2]
  - Drug Discovery [3]
  - Anomaly Detection

[3]



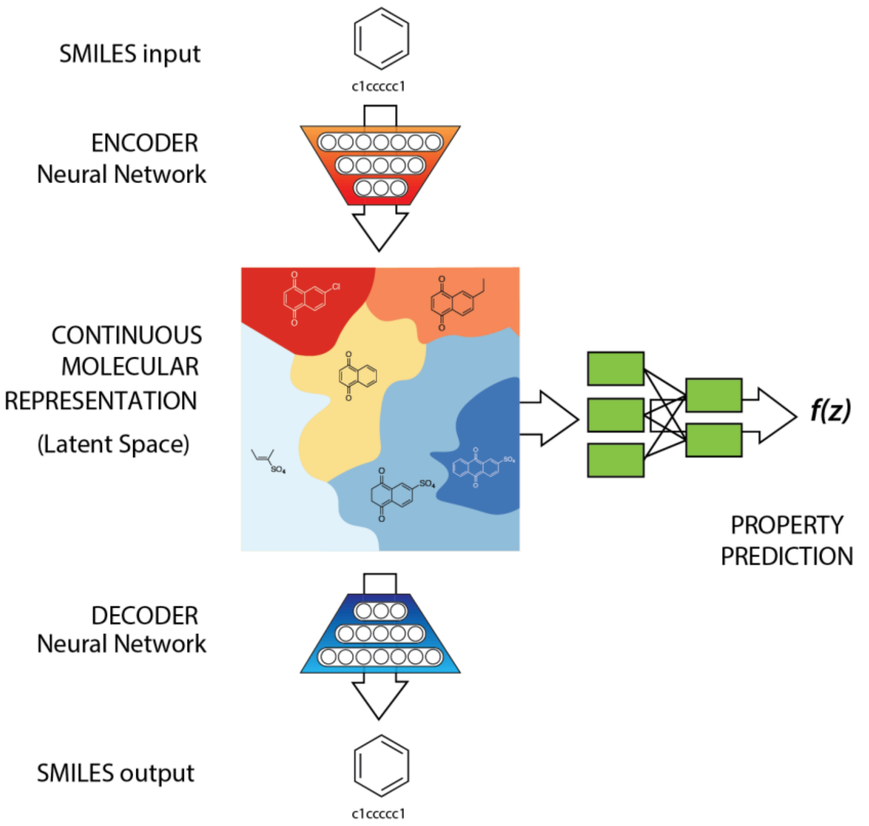
[2] (Ramesh et al., 2022) [3] (Gómez-Bombarelli et al., 2018)

Introduction

Why inference? Why VAEs?

- VAEs are applied in innumerable tasks:
  - Generation, completion, edition, manipulation [2]
  - Drug Discovery [3]
  - Anomaly Detection
    - ➡ Medical imaging

[3]



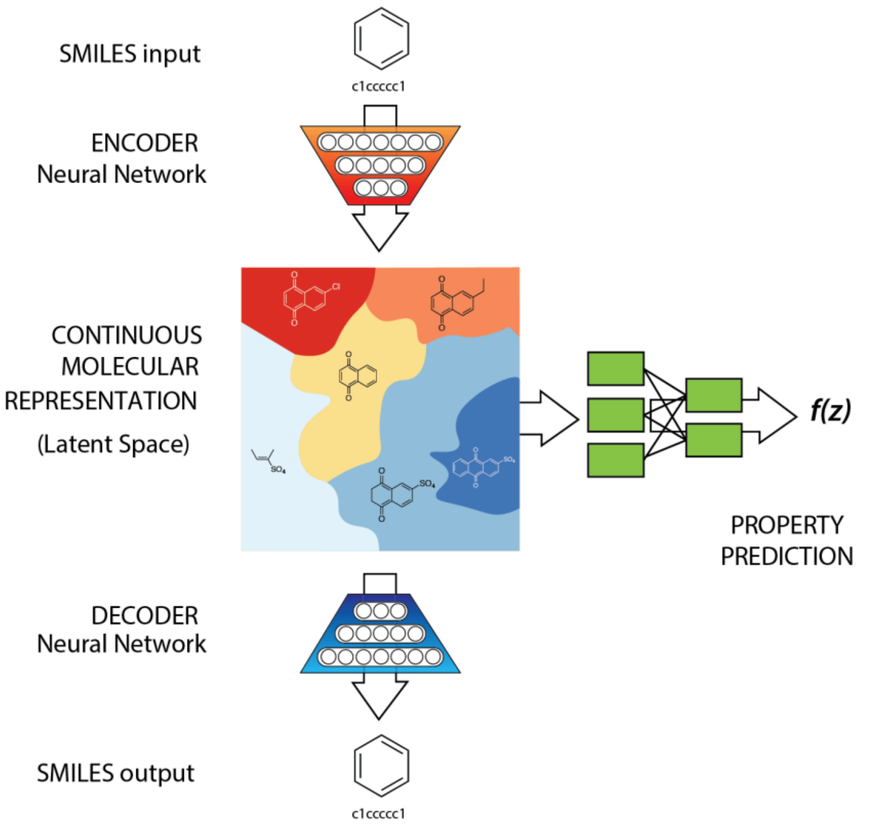
[2] (Ramesh et al., 2022) [3] (Gómez-Bombarelli et al., 2018)

Introduction

Why inference? Why VAEs?

- VAEs are applied in innumerable tasks:
  - Generation, completion, edition, manipulation [2]
  - Drug Discovery [3]
  - Anomaly Detection
    - ➡ Medical imaging
    - ➡ Industry manufacturing

[3]



[2] (Ramesh et al., 2022) [3] (Gómez-Bombarelli et al., 2018)

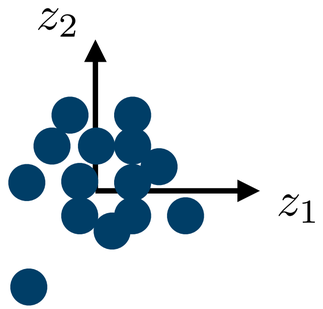
Outline

Contents

Variational Autoencoders

Latent space

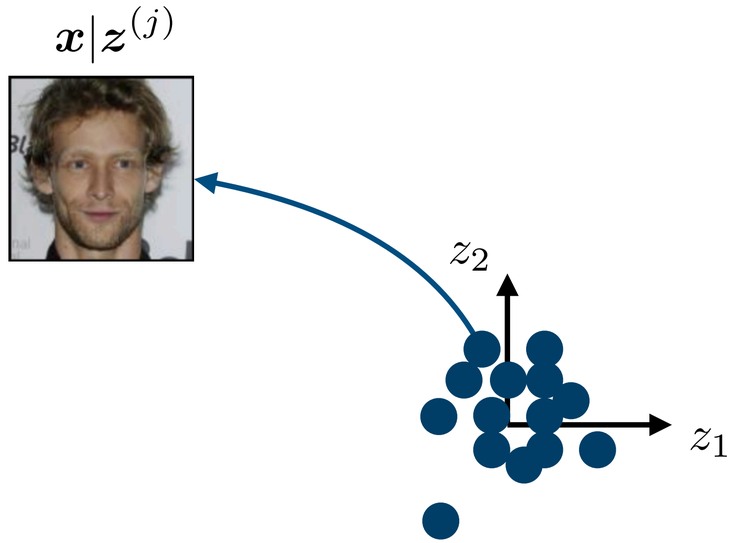
- The **latent space** $\boldsymbol{z}$ of lower dimensionality *encodes* the information of a datapoint.



Variational Autoencoders

Latent space

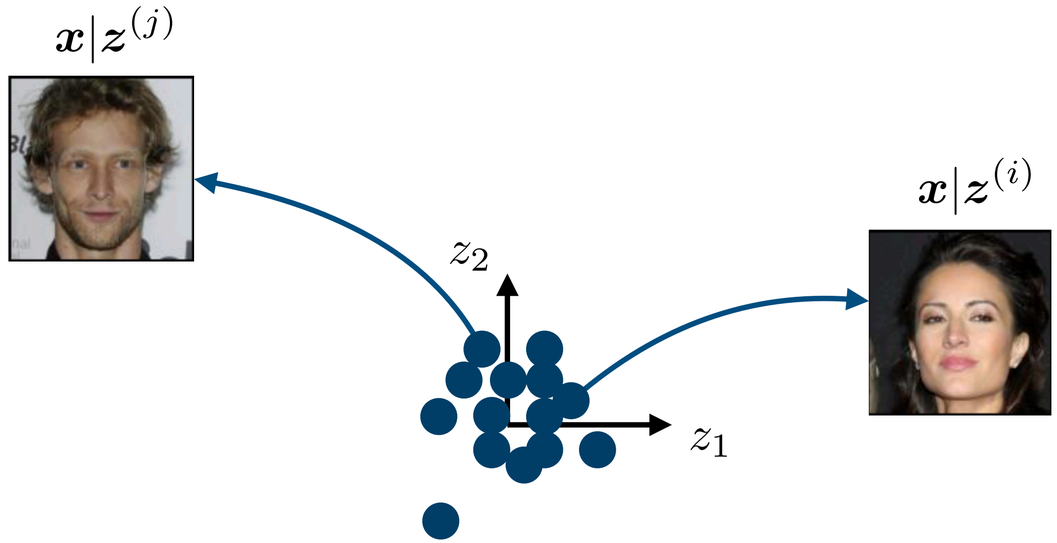
- The **latent space** $\boldsymbol{z}$ of lower dimensionality *encodes* the information of a datapoint.



Variational Autoencoders

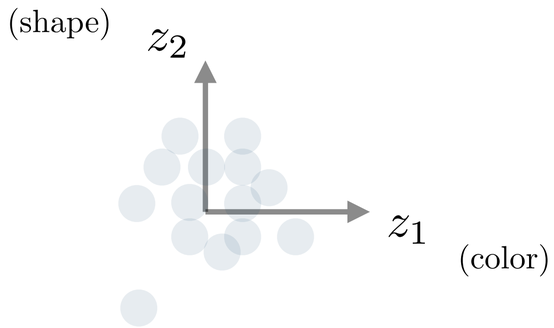
Latent space

- The **latent space** $\boldsymbol{z}$ of lower dimensionality *encodes* the information of a datapoint.



Variational Autoencoders

Disentanglement

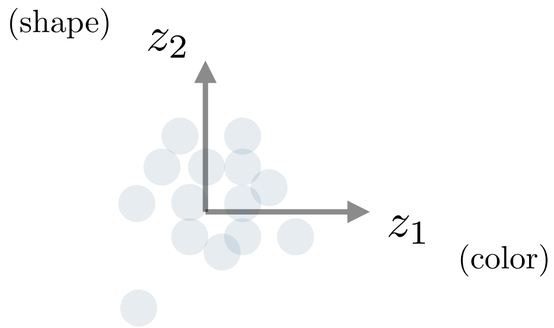


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

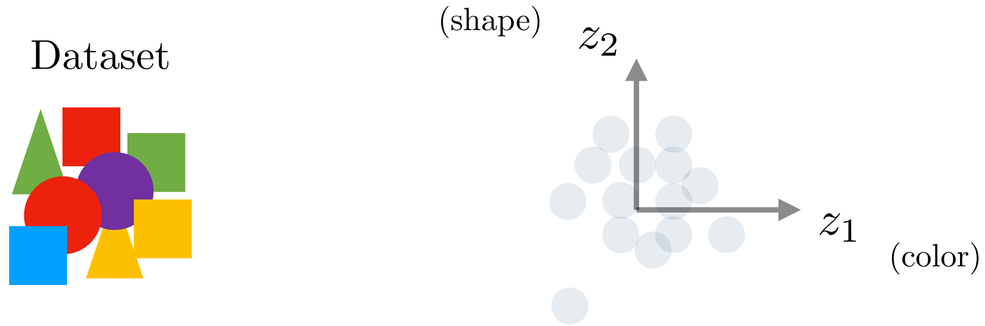


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

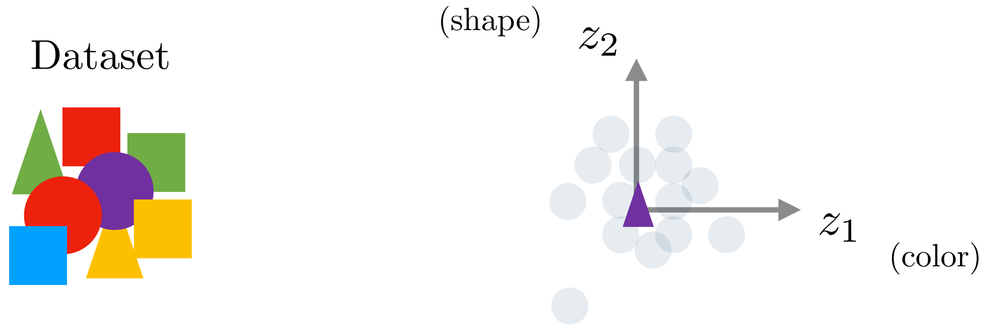


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

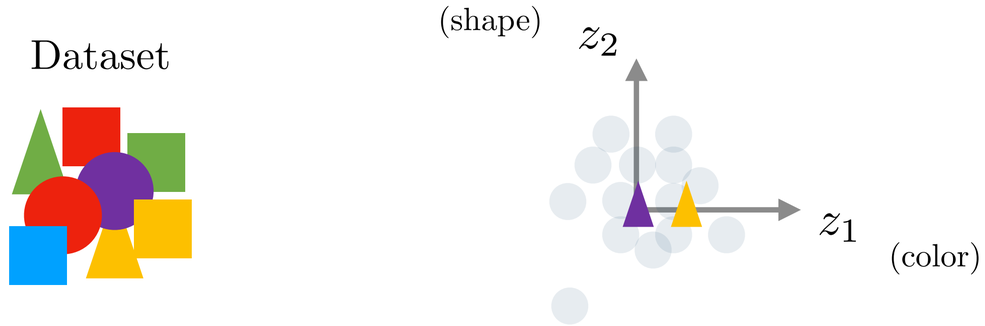


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

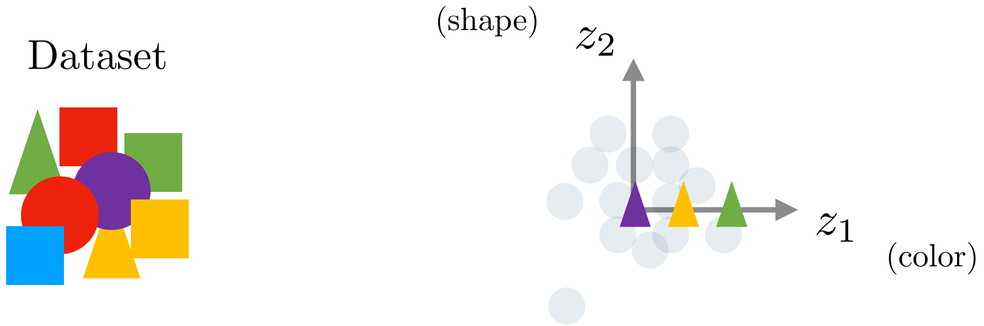


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

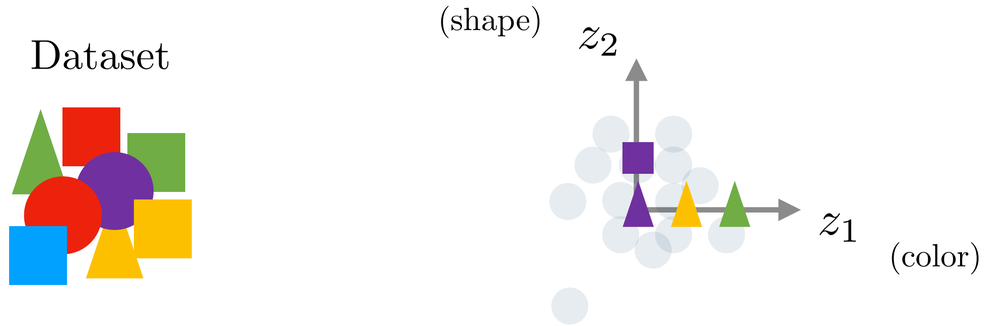


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

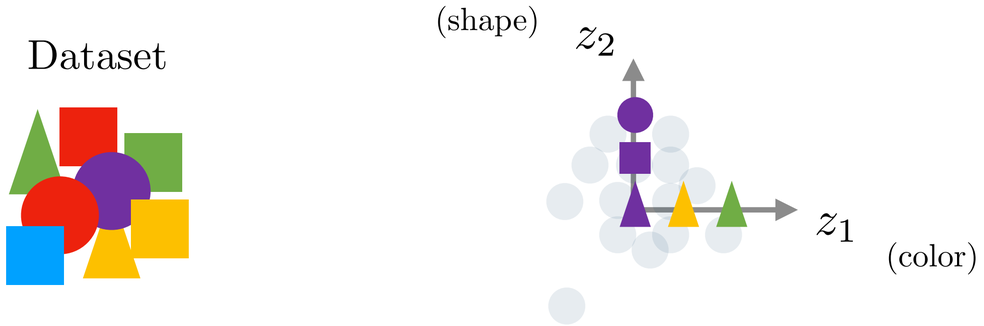


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].

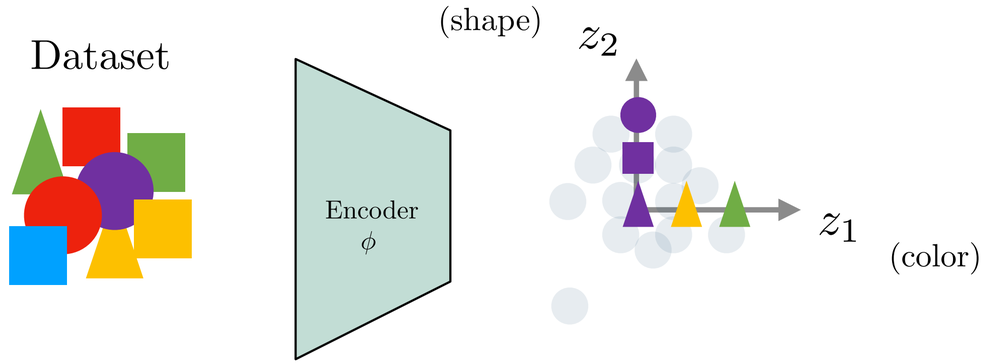


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Disentanglement

- The modulation of true generative factors by dimensions of the latent space is denoted in the literature by **disentanglement** [18,19].
  - Enforcing/assessing the disentanglement is a non-trivial task [18].

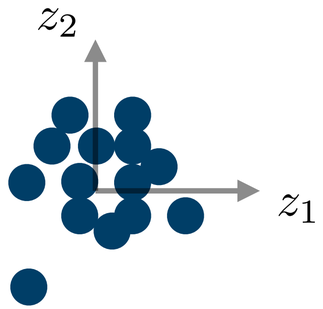


[18] (Locatello et al., 2019) [19] (Mathieu et al., 2019)

Variational Autoencoders

Prior

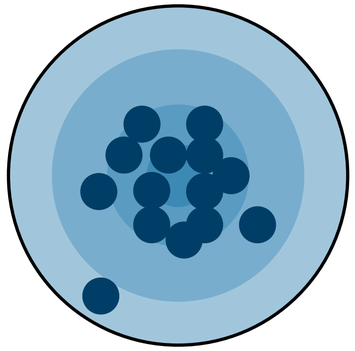
- In VAEs, a **prior** probability distribution is defined over a **latent space**.



Variational Autoencoders

Prior

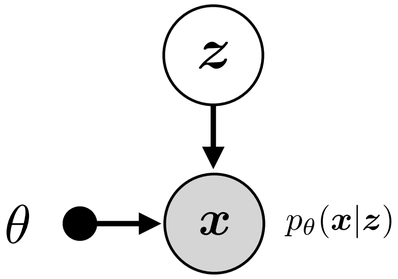
- In VAEs, a **prior** probability distribution is defined over a **latent space**.



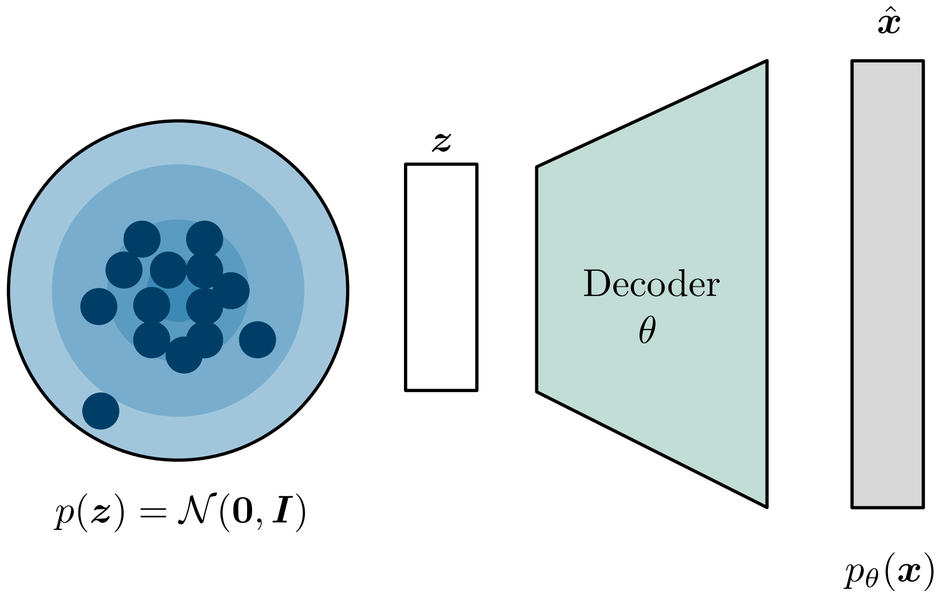
$$p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$

Variational Autoencoders

Decoder

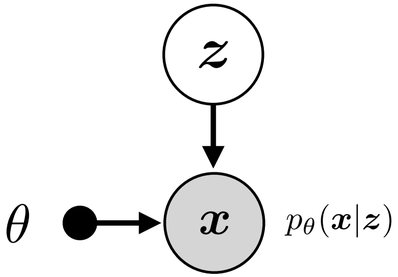


- To *decode* the latent samples we use a Neural Network as a **decoder**.



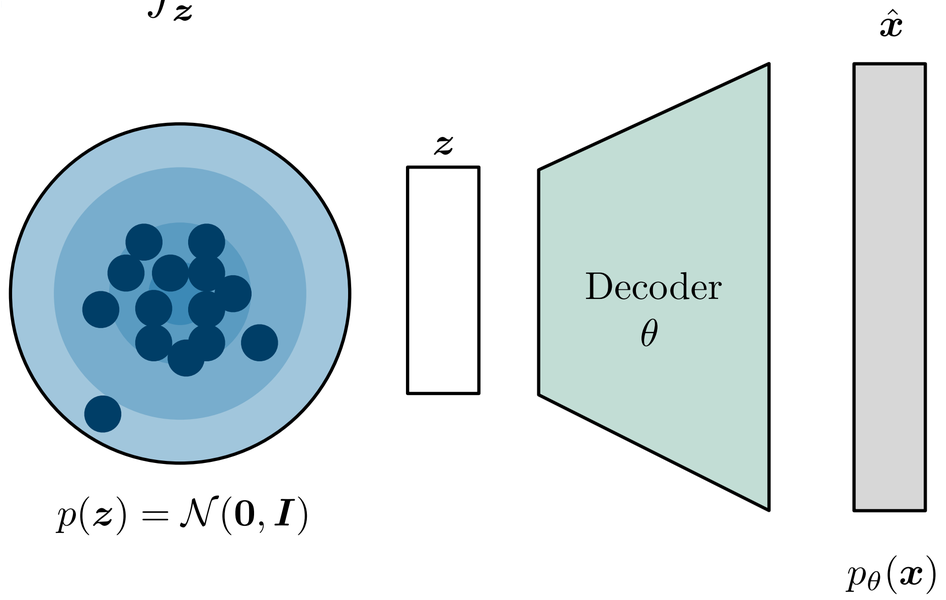
Variational Autoencoders

Marginal Likelihood



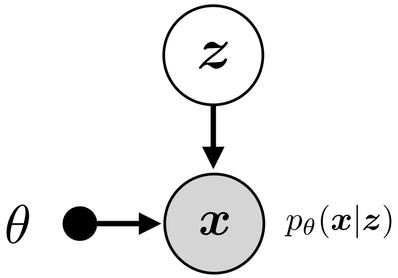
- To *learn* the parameters of this model, we should maximise the log **marginal likelihood**, or log **evidence**:

$$\log p_\theta(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$$



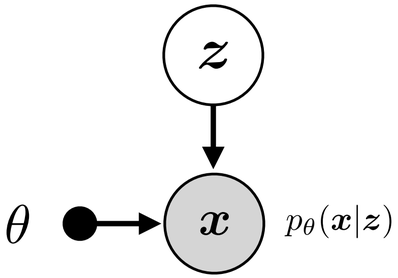
Variational Autoencoders

Marginal Likelihood



Variational Autoencoders

Marginal Likelihood

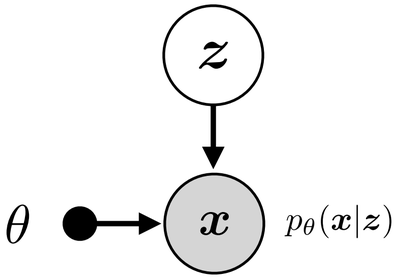


- Unfortunately, the integral is **intractable**.

$$\log p_\theta(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$$

Variational Autoencoders

Marginal Likelihood

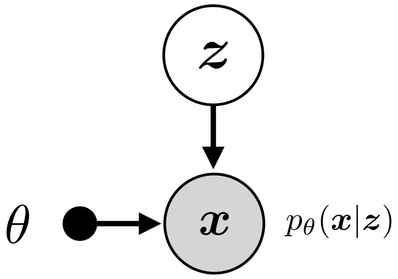


- Unfortunately, the integral is **intractable**.

$$\log p_\theta(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} \overline{p_\theta(\boldsymbol{x}|\boldsymbol{z})} p(\boldsymbol{z}) d\boldsymbol{z}$$

Variational Autoencoders

Marginal Likelihood



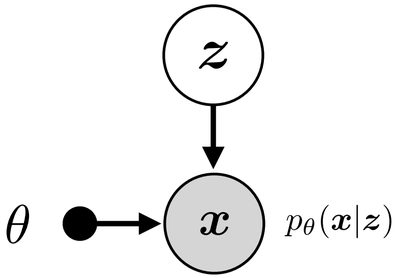
- Unfortunately, the integral is **intractable**.

$$\log p_\theta(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} \overline{p_\theta(\boldsymbol{x}|\boldsymbol{z})} p(\boldsymbol{z}) d\boldsymbol{z}$$

- There is not a close expression for the objective. An **approximation** is required.

Variational Autoencoders

Marginal Likelihood



- Unfortunately, the integral is **intractable**.

$$\log p_\theta(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} \overline{p_\theta(\boldsymbol{x}|\boldsymbol{z})} p(\boldsymbol{z}) d\boldsymbol{z}$$

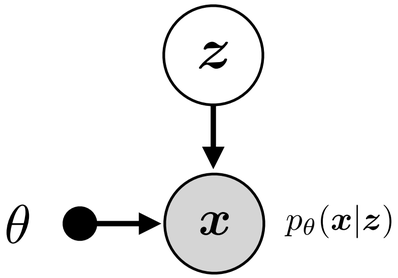
- There is not a close expression for the objective. An **approximation** is required.
- Equivalently, we can't *infer* the **posterior** of an observation!

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p_\theta(\boldsymbol{x})}$$

Bayes Theorem

Variational Autoencoders

Marginal Likelihood



- Unfortunately, the integral is **intractable**.

$$\log p_\theta(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$$

- There is not a close expression for the objective. An **approximation** is required.
- Equivalently, we can't *infer* the **posterior** of an observation!

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z})}{p_\theta(\boldsymbol{x})}$$

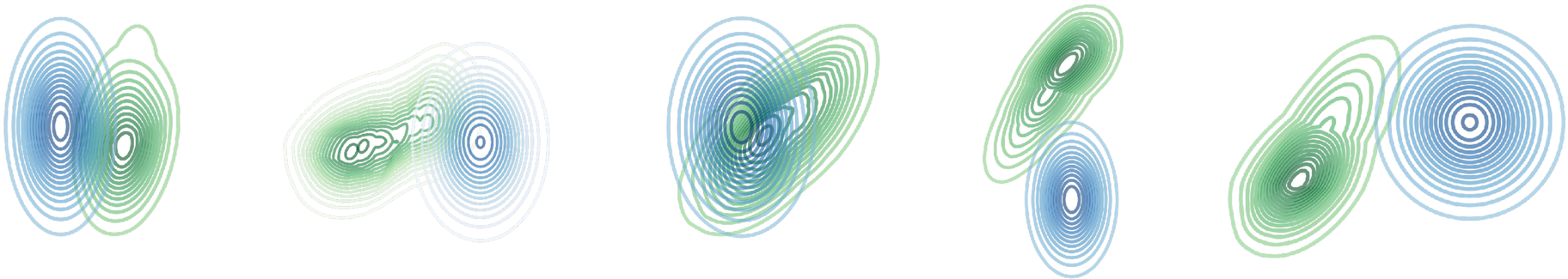
Bayes Theorem

Variational Autoencoders

Amortized Variational Inference

- A **Gaussian approximation** of the posterior is considered instead

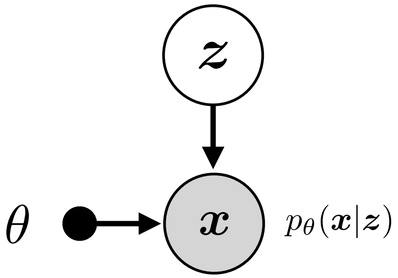
$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z};\ \boldsymbol{\mu},\ \boldsymbol{\sigma}^2\boldsymbol{I})$$



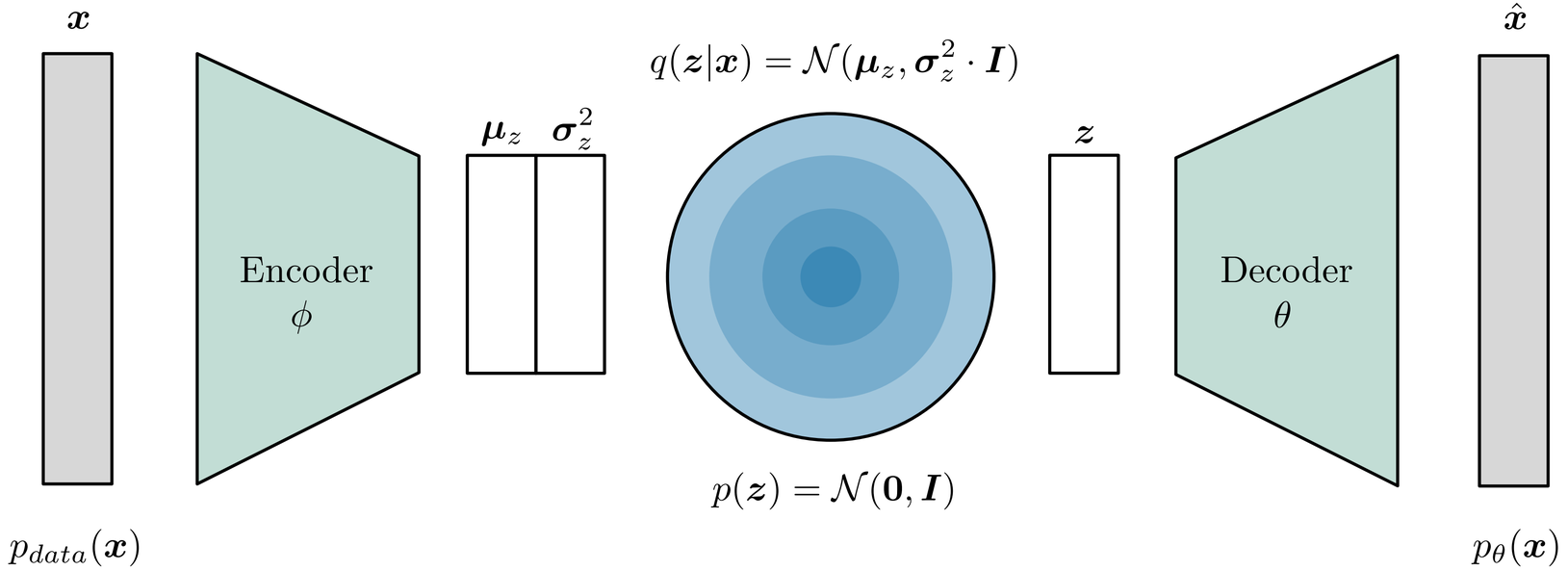
$q(\boldsymbol{z}|\boldsymbol{x})$ $p(\boldsymbol{z}|\boldsymbol{x})$

Variational Autoencoders

Encoder

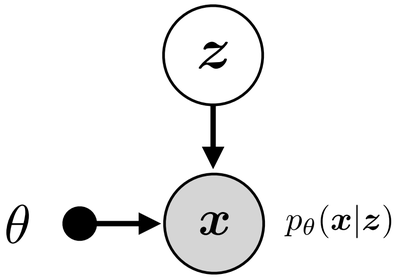


- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

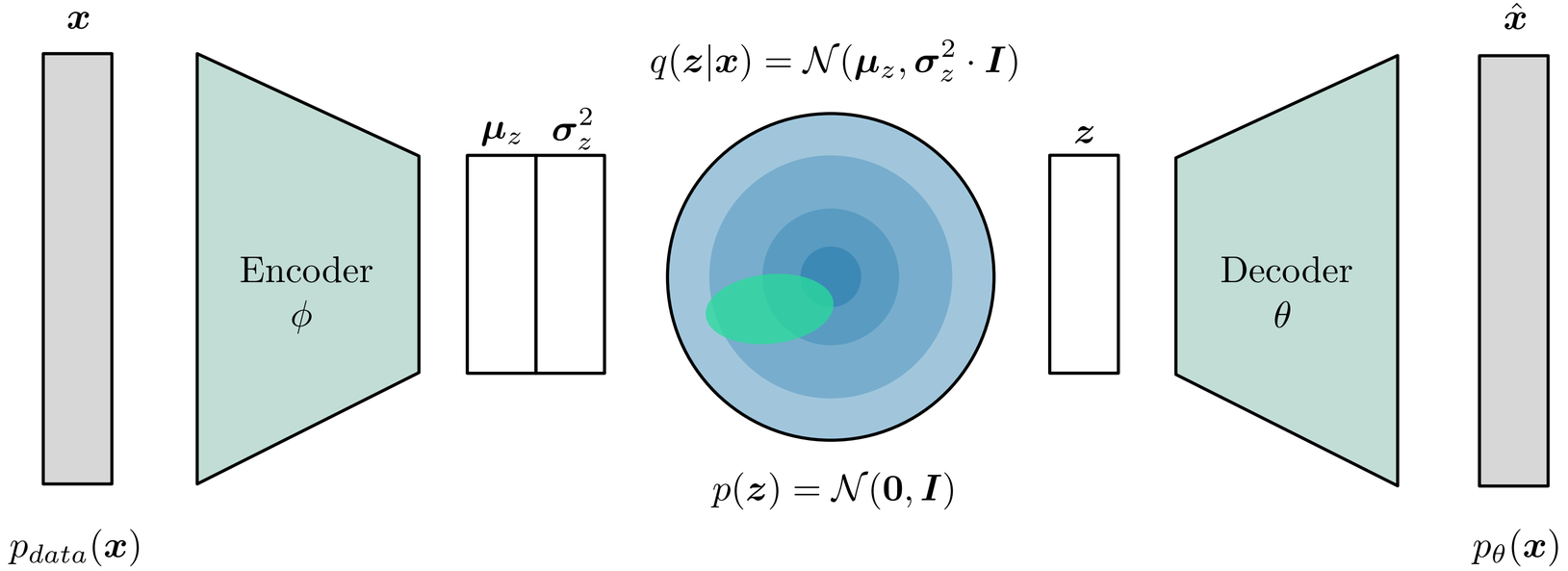


Variational Autoencoders

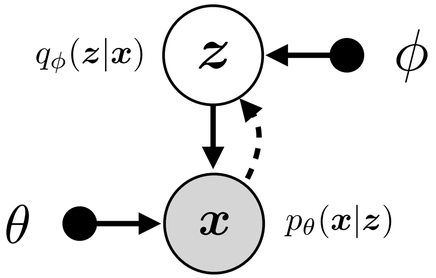
Encoder



- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

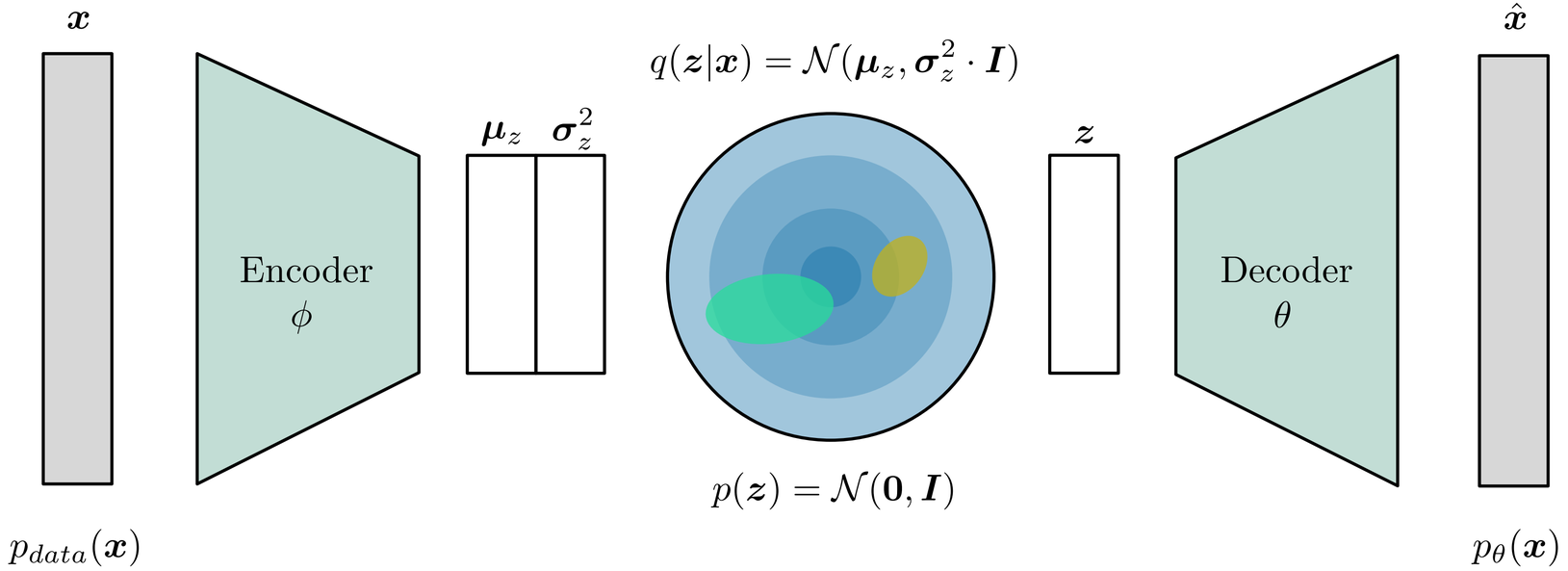


Variational Autoencoders

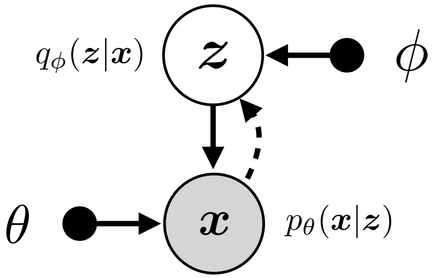


Encoder

- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

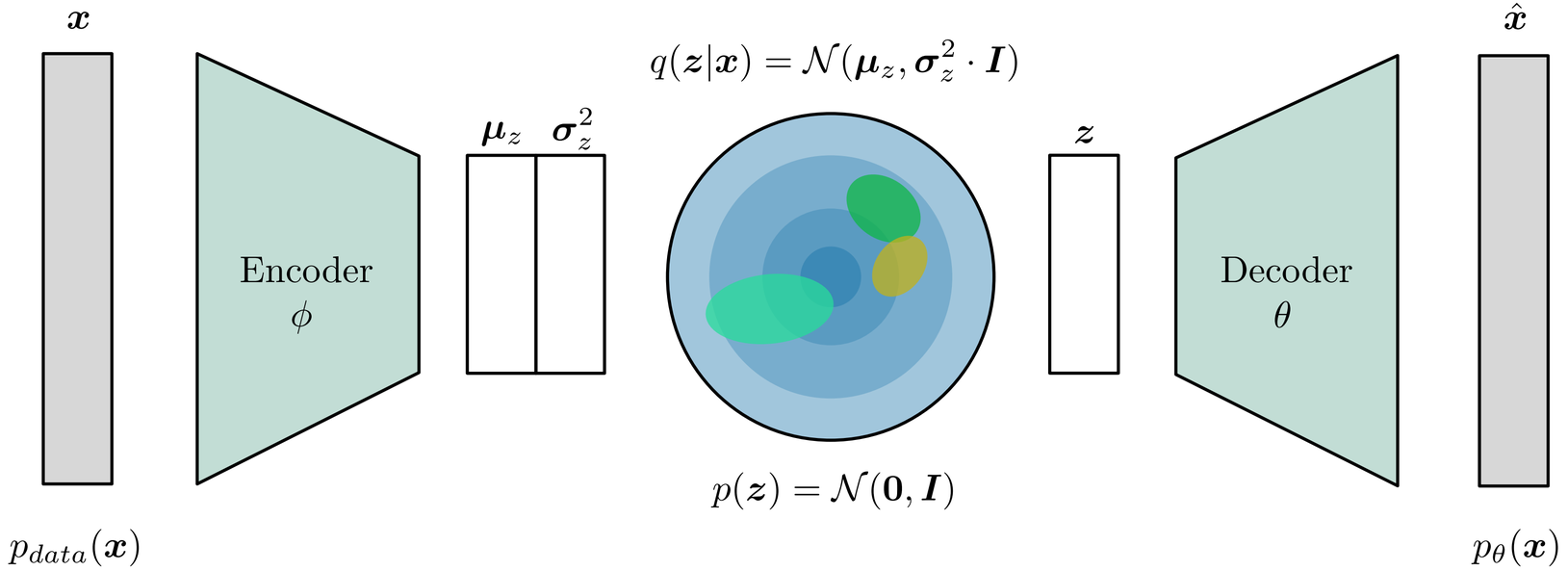


Variational Autoencoders

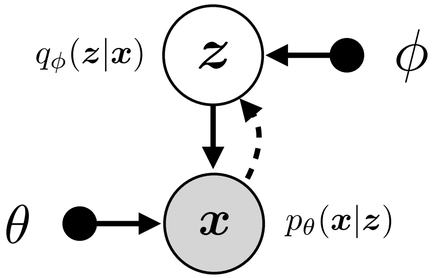


Encoder

- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

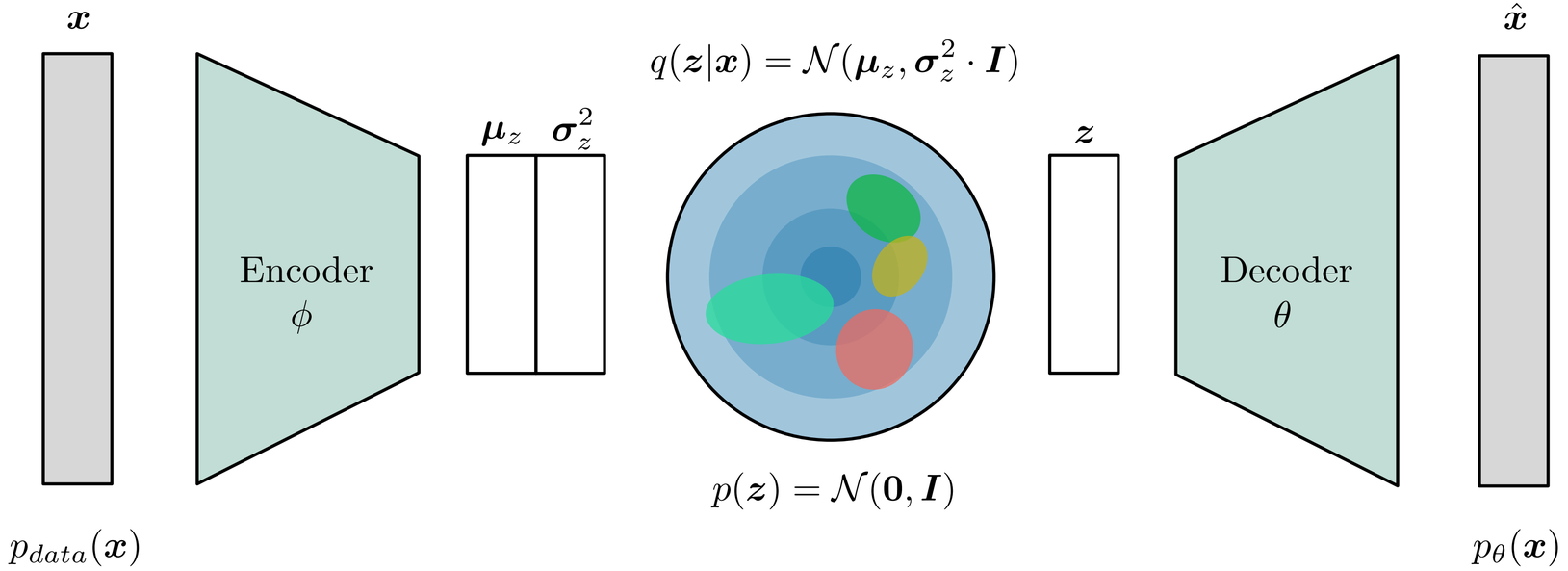


Variational Autoencoders



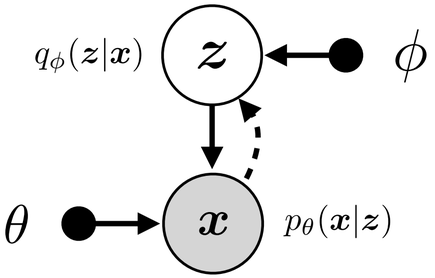
Encoder

- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

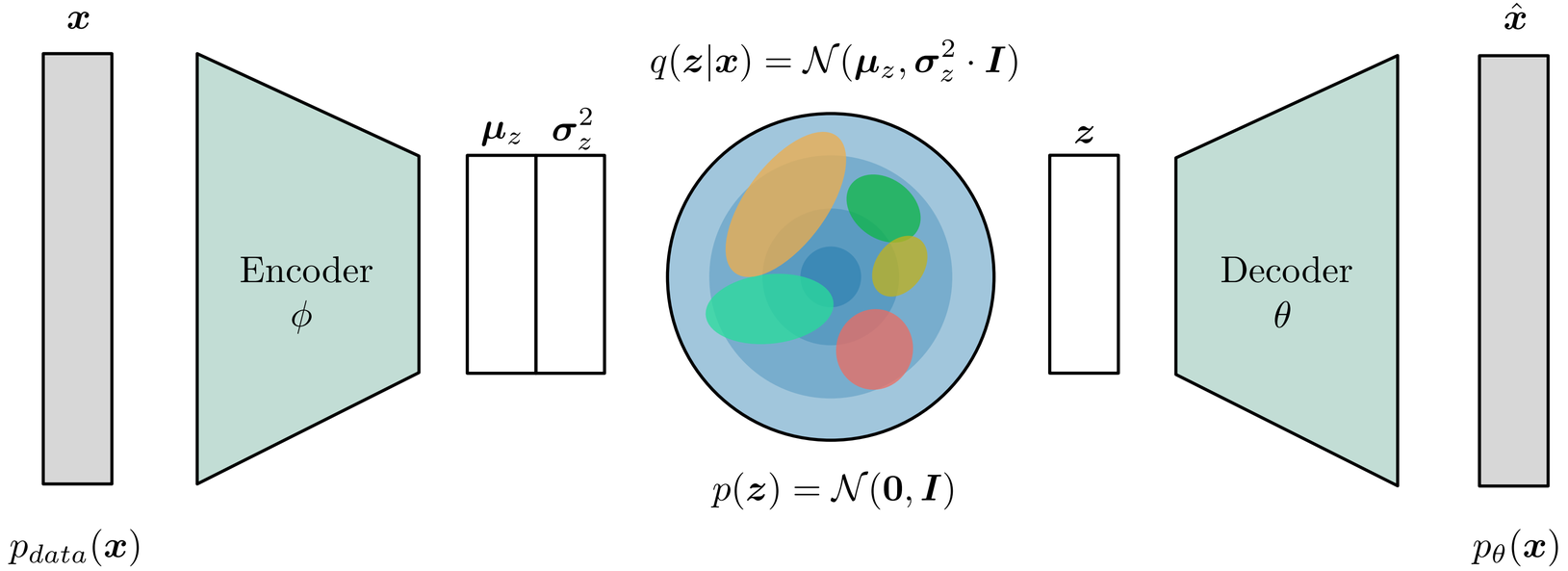


Variational Autoencoders

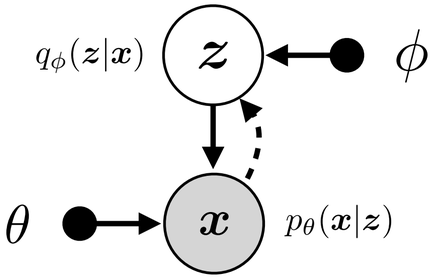
Encoder



- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

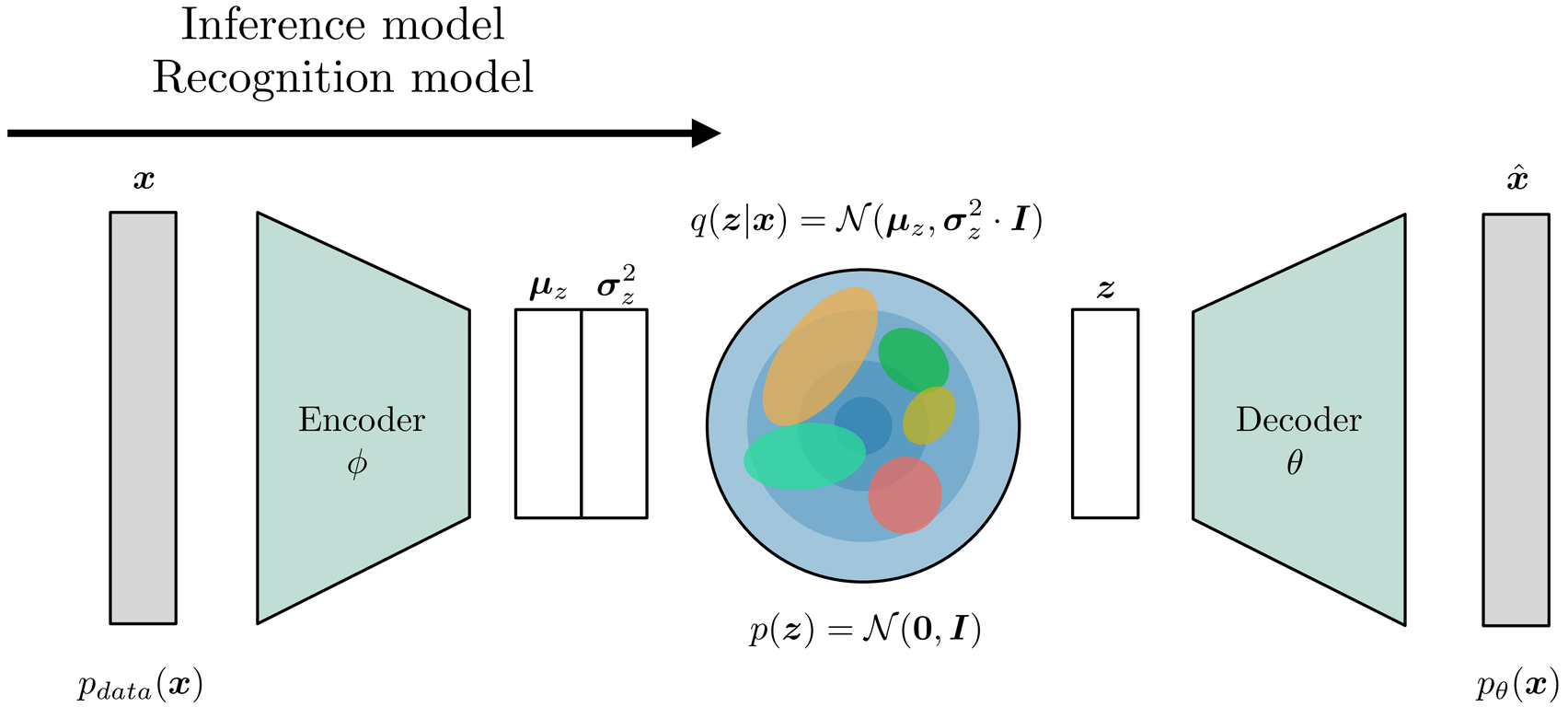


Variational Autoencoders



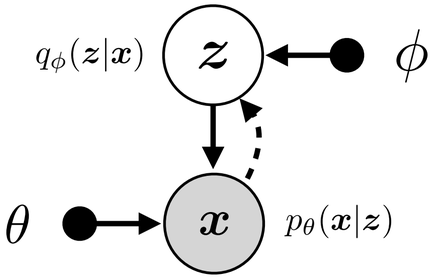
Encoder

- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.

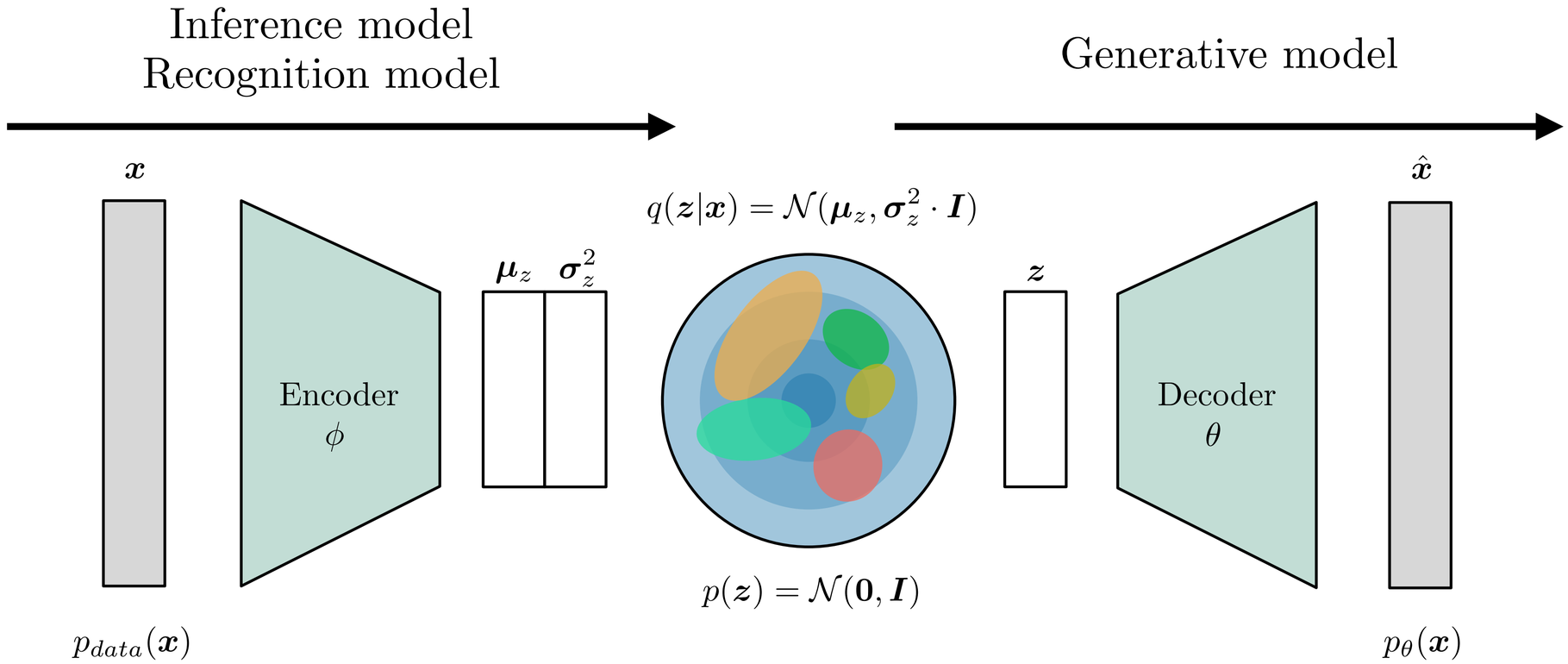


Variational Autoencoders

Encoder



- A second Neural Network, called **encoder**, maps observations to the Gaussian parameters.



Variational Autoencoders

Evidence Lower Bound

Variational Autoencoders

Evidence Lower Bound

- We match the approximate and true posterior by minimizing

$$D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right)$$

Variational Autoencoders

Evidence Lower Bound

- We match the approximate and true posterior by minimizing

$$D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right)$$

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p_\theta(\boldsymbol{x})}$$

- After a few derivations...

$$\log p(\boldsymbol{x}) = D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right]$$

Variational Autoencoders

Evidence Lower Bound

- We match the approximate and true posterior by minimizing

$$D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right)$$

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p_\theta(\boldsymbol{x})}$$

- After a few derivations...

$$\log p(\boldsymbol{x}) = D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right]$$

Variational Autoencoders

Evidence Lower Bound

- We match the approximate and true posterior by minimizing

$$D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right)$$

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p_\theta(\boldsymbol{x})}$$

- After a few derivations...

$$\log p(\boldsymbol{x}) = D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})\right) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right]$$

- It turns out that we can equivalently maximize a **lower bound**

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \log \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}$$

Variational Autoencoders

Evidence Lower Bound

Variational Autoencoders

Evidence Lower Bound

- This objective is the so-called **Evidence Lower Bound (ELBO)**, typically expressed as

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})\right)$$

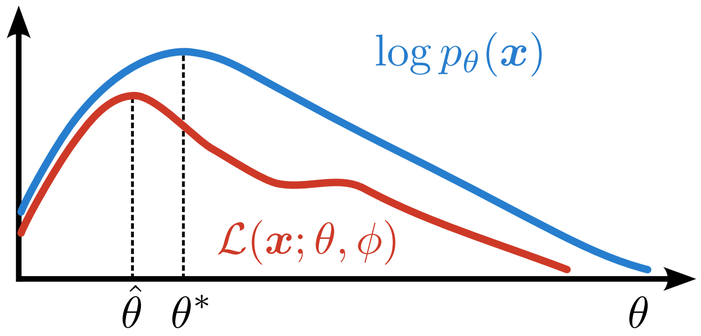
Variational Autoencoders

Evidence Lower Bound

- This objective is the so-called **Evidence Lower Bound (ELBO)**, typically expressed as

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})\right)$$

- It is a well-defined objective for learning the parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ of the VAE.



Variational Autoencoders

Scalable training with Stochastic Gradient Descent (SGD)

For each batch of B samples:

1. Encode to the parameters of the approximate posteriors $q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)$.
2. Draw a sample from each $q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)$.
3. Optimization step on $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.

$$\nabla_{(\theta,\phi)} \left(\frac{1}{B} \sum_{i=1}^{B} \left(\log p_\theta(\boldsymbol{x}_i|\boldsymbol{z}) - D_{KL} \left(q_\phi(\boldsymbol{z}|\boldsymbol{x}_i) \| p(\boldsymbol{z}) \right) \right) \right)$$

[1] (Kingma et al., 2013)

Variational Autoencoders

Scalable training with Stochastic Gradient Descent (SGD)

For each batch of B samples:

1. Encode to the parameters of the approximate posteriors $q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)$.
2. Draw a sample from each $q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)$.
3. Optimization step on $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.

$$\nabla_{(\theta,\phi)} \left(\frac{1}{B} \sum_{i=1}^{B} \left(\log p_\theta(\boldsymbol{x}_i|\boldsymbol{z}) - D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}_i) \| p(\boldsymbol{z}) \right) \right) \right)$$

Reparameterization trick[1]

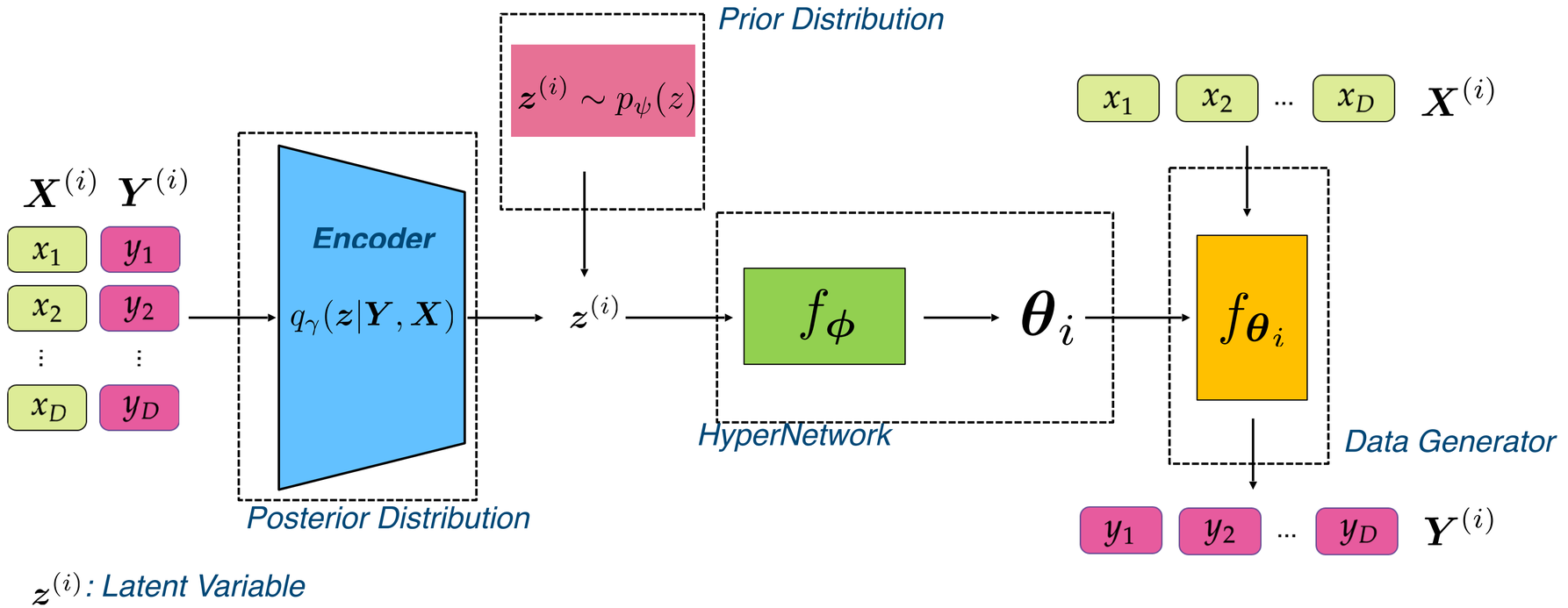
$$\boldsymbol{z}^{(s)} = f_{\boldsymbol{\mu}}(\boldsymbol{x}) + f_{\boldsymbol{\sigma}}(\boldsymbol{x}) \cdot \epsilon^{(s)}$$
$$\epsilon^{(s)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$

Works reasonably good even for *S=1*

[1] (Kingma et al., 2013)

Variational Autoencoders

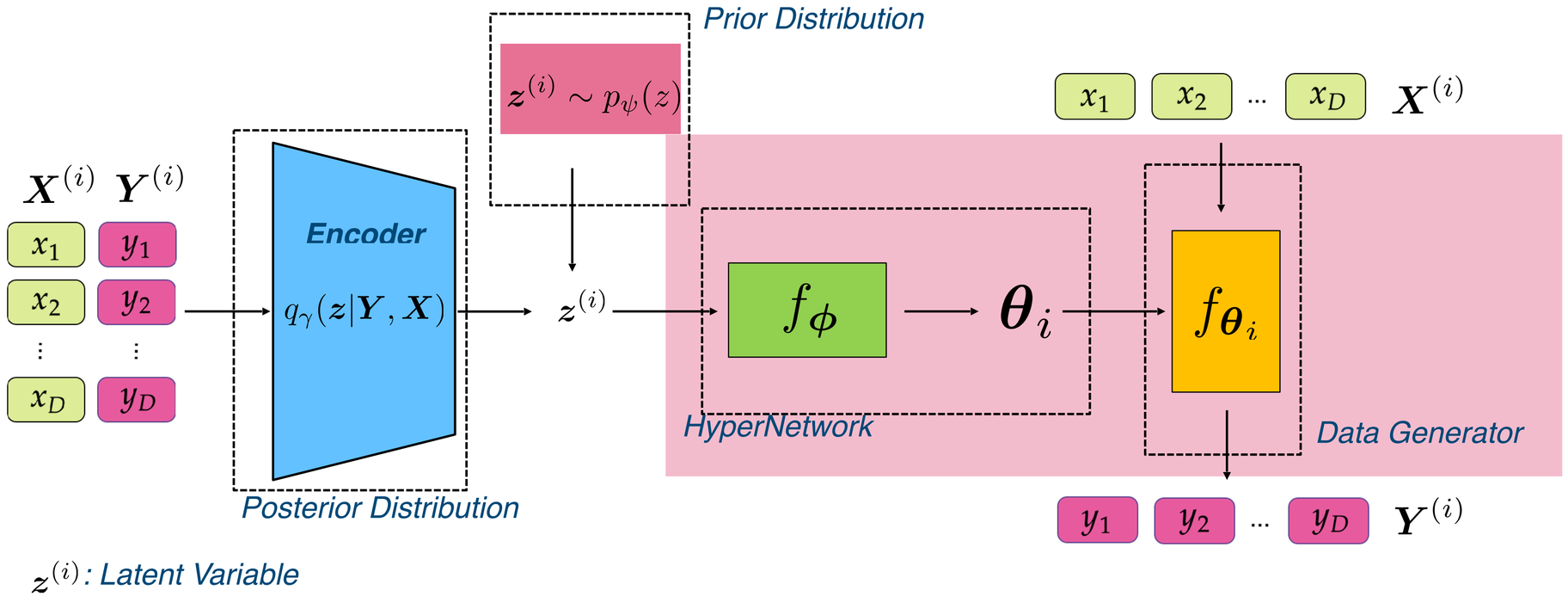
VAEs in the functional space [26]



B. Koyuncu, P. Sanchez-Martin, **I. Peis**, P. M. Olmos and I. Valera. Variational Mixture of HyperGenerators for Learning Distributions Over Functions. In *Proceedings of the 40th International Conference on Machine Learning (**ICML**)*, 2023.

Variational Autoencoders

VAEs in the functional space [26]

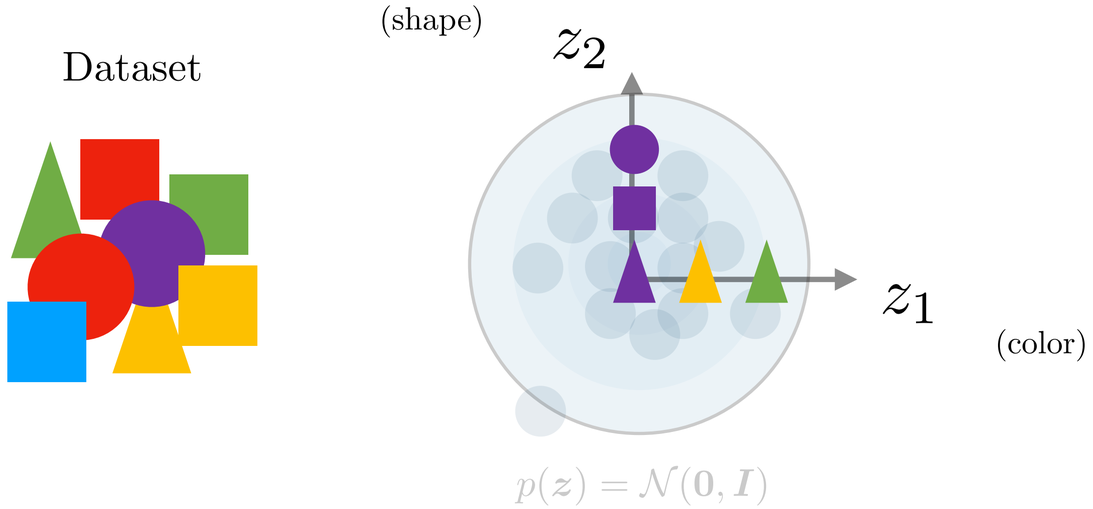


B. Koyuncu, P. Sanchez-Martin, **I. Peis**, P. M. Olmos and I. Valera. Variational Mixture of HyperGenerators for Learning Distributions Over Functions. In *Proceedings of the 40th International Conference on Machine Learning (**ICML**)*, 2023.

Variational Autoencoders

Current challenges and gaps in VAEs

- How to control the **Representation learning** in VAEs [15-17]?

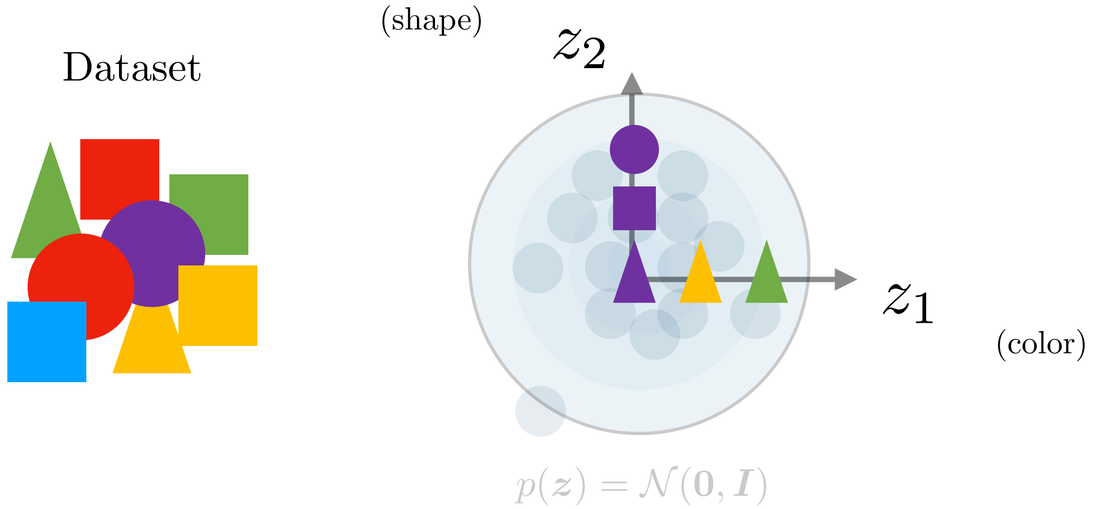


[15] (Bengio et al., 2013) [16] (Eastwood and Williams, 2018) [17] (Higgins et al., 2018) [18] (Locatello et al., 2019) [19] (Mathieu et al., 2019) [20] (Bouchacourt et al., 2018)

Variational Autoencoders

Current challenges and gaps in VAEs

- How to control the **Representation learning** in VAEs [15-17]?
  - Defining metrics to assess the *disentanglement* of VAEs [18,19].

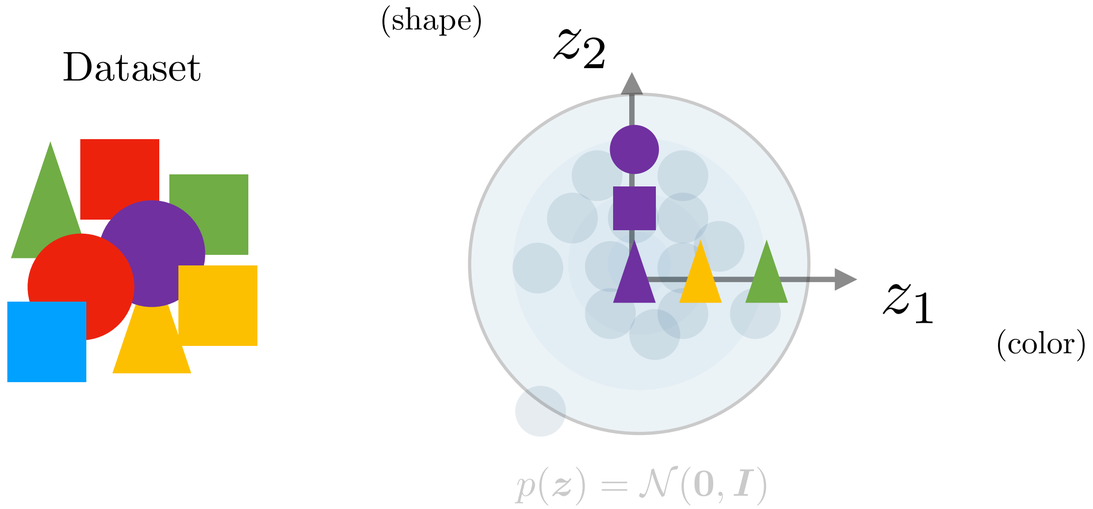


[15] (Bengio et al., 2013) [16] (Eastwood and Williams, 2018) [17] (Higgins et al., 2018) [18] (Locatello et al., 2019) [19] (Mathieu et al., 2019) [20] (Bouchacourt et al., 2018)

Variational Autoencoders

Current challenges and gaps in VAEs

- How to control the **Representation learning** in VAEs [15-17]?



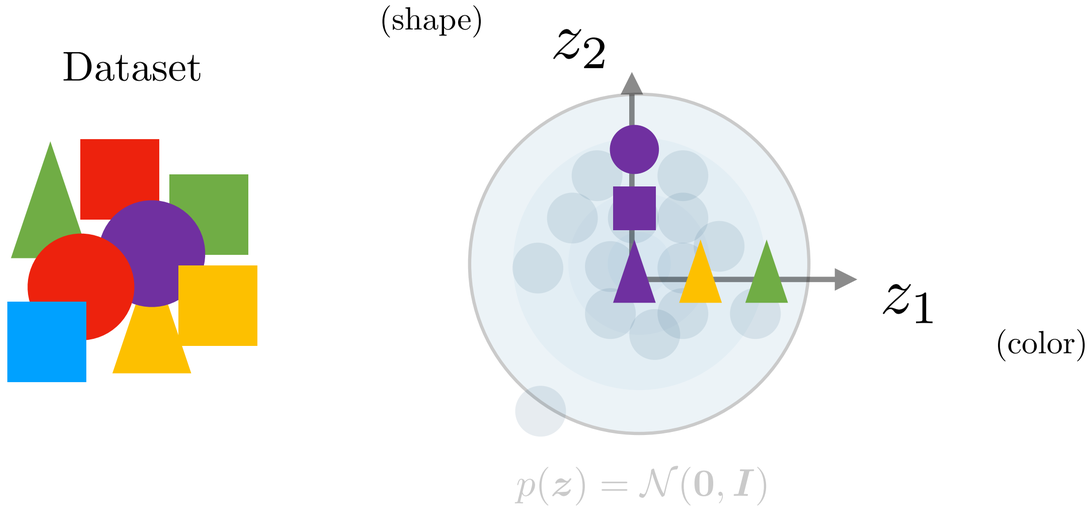
- Defining metrics to assess the *disentanglement* of VAEs [18,19].
- Achieving *disentanglement* via model design or semi-supervision [20].

[15] (Bengio et al., 2013) [16] (Eastwood and Williams, 2018) [17] (Higgins et al., 2018) [18] (Locatello et al., 2019) [19] (Mathieu et al., 2019) [20] (Bouchacourt et al., 2018)

Variational Autoencoders

Current challenges and gaps in VAEs

- How to control the **Representation learning** in VAEs [15-17]?
  - Defining metrics to assess the *disentanglement* of VAEs [18,19].

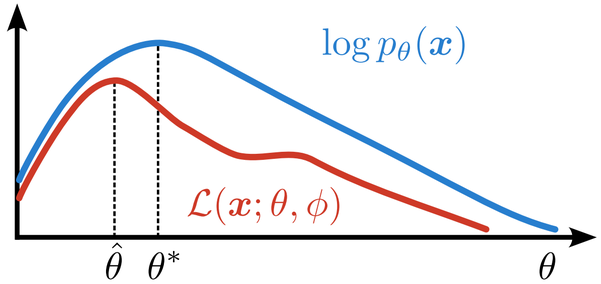


  - Achieving *disentanglement* via model design or semi-supervision [20].

[15] (Bengio et al., 2013) [16] (Eastwood and Williams, 2018) [17] (Higgins et al., 2018) [18] (Locatello et al., 2019) [19] (Mathieu et al., 2019) [20] (Bouchacourt et al., 2018)

Variational Autoencoders

Current challenges and gaps in VAEs

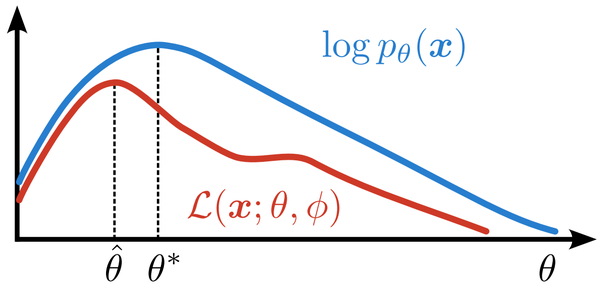


[1] (Kingma et al., 2013) [4] (Cremer et at., 2018) [5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)
[10] (Dilokthanakul et al., 2016) [11] (Tomczak and Welling, 2021) [12] (Vahdat and Kautz, 2020) [13] (Child, 2020) [14] (Maaløe et al., 2019)

Variational Autoencoders

Current challenges and gaps in VAEs

- They require **approximate inference** [1,4].
  - Better approximation of the log-evidence [5,6] or its gradients [7,8].
  - Better quality of the samples from the approximate posterior [9].

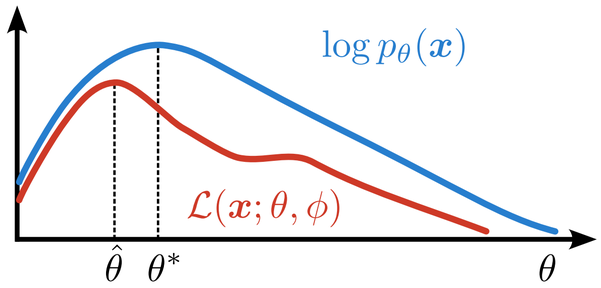


[1] (Kingma et al., 2013) [4] (Cremer et at., 2018) [5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)
[10] (Dilokthanakul et al., 2016) [11] (Tomczak and Welling, 2021) [12] (Vahdat and Kautz, 2020) [13] (Child, 2020) [14] (Maaløe et al., 2019)

Variational Autoencoders

Current challenges and gaps in VAEs

- They require **approximate inference** [1,4].
  - Better approximation of the log-evidence [5,6] or its gradients [7,8].
  - Better quality of the samples from the approximate posterior [9].
- Increasing **flexibility** with prior design.
  - Mixtures [10,11], Hierarchical VAEs [12-14], etc.

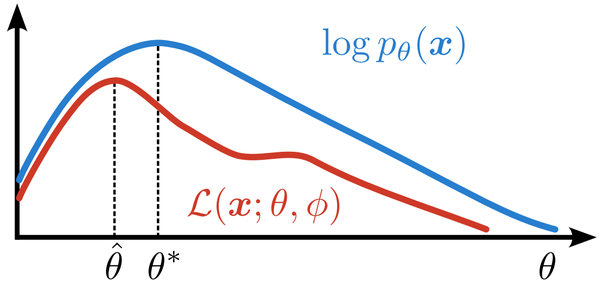


[1] (Kingma et al., 2013) [4] (Cremer et at., 2018) [5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)
[10] (Dilokthanakul et al., 2016) [11] (Tomczak and Welling, 2021) [12] (Vahdat and Kautz, 2020) [13] (Child, 2020) [14] (Maaløe et al., 2019)

Variational Autoencoders

Current challenges and gaps in VAEs

- They require **approximate inference** [1,4].
  - Better approximation of the log-evidence [5,6] or its gradients [7,8].
  - Better quality of the samples from the approximate posterior [9].
- Increasing **flexibility** with prior design.
  - Mixtures [10,11], Hierarchical VAEs [12-14], etc.



[1] (Kingma et al., 2013) [4] (Cremer et at., 2018) [5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)
[10] (Dilokthanakul et al., 2016) [11] (Tomczak and Welling, 2021) [12] (Vahdat and Kautz, 2020) [13] (Child, 2020) [14] (Maaløe et al., 2019)

Outline

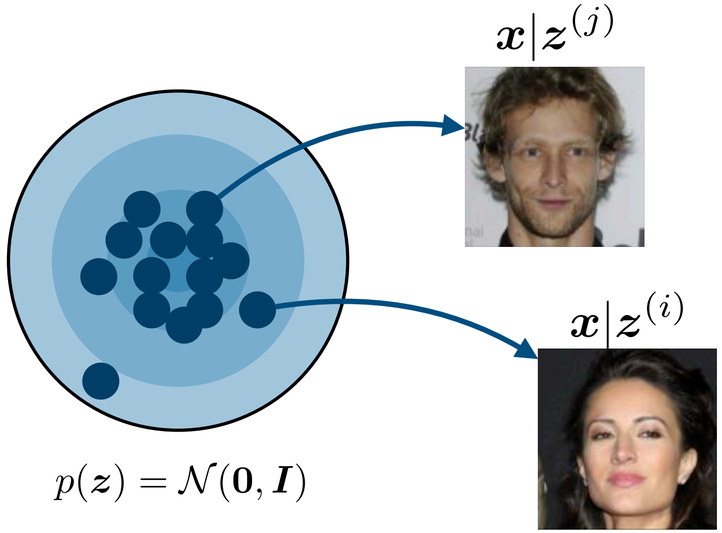
Contents

Contribution I: Unsupervised Learning of Global Factors in VAEs

Problem Statement

The *i.i.d.* assumption

- The large majority of VAEs are designed over the assumption that data is $i.i.d.$

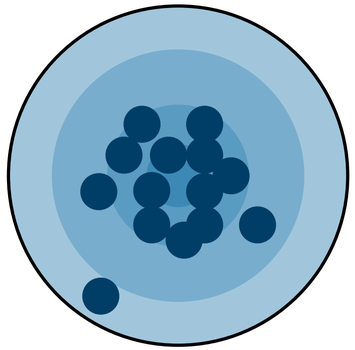


- Each latent representation encodes a single, independent datapoint.

Problem Statement

Research Questions

- What if we wanted to generate grouped data?



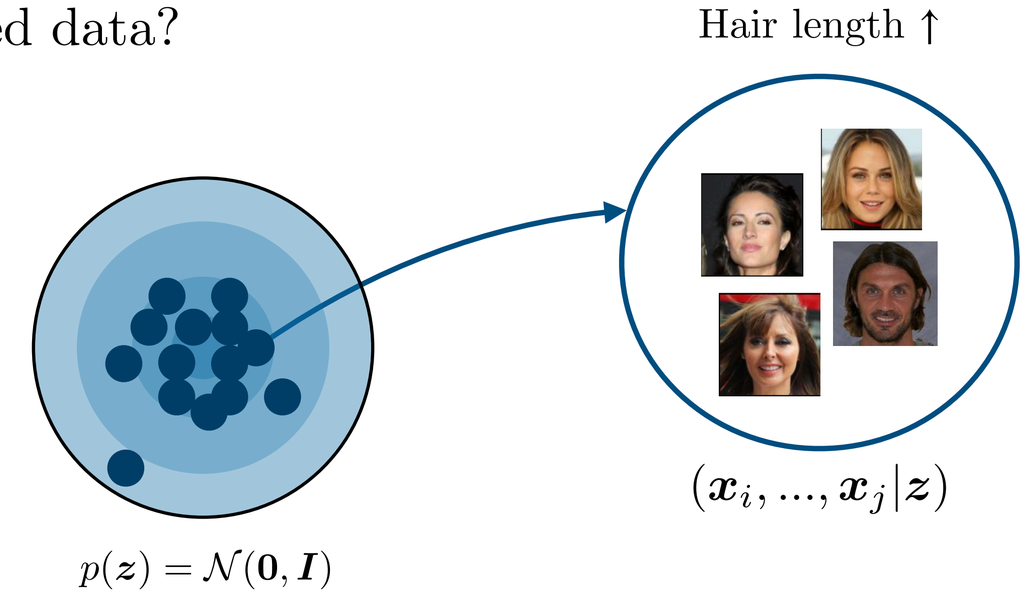
$$p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$

- Data within each group would share some generative properties.

Problem Statement

Research Questions

- What if we wanted to generate grouped data?

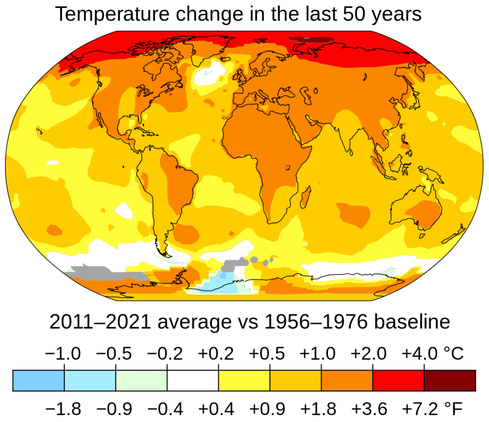


- Data within each group would share some generative properties.

Problem Statement

Research Questions

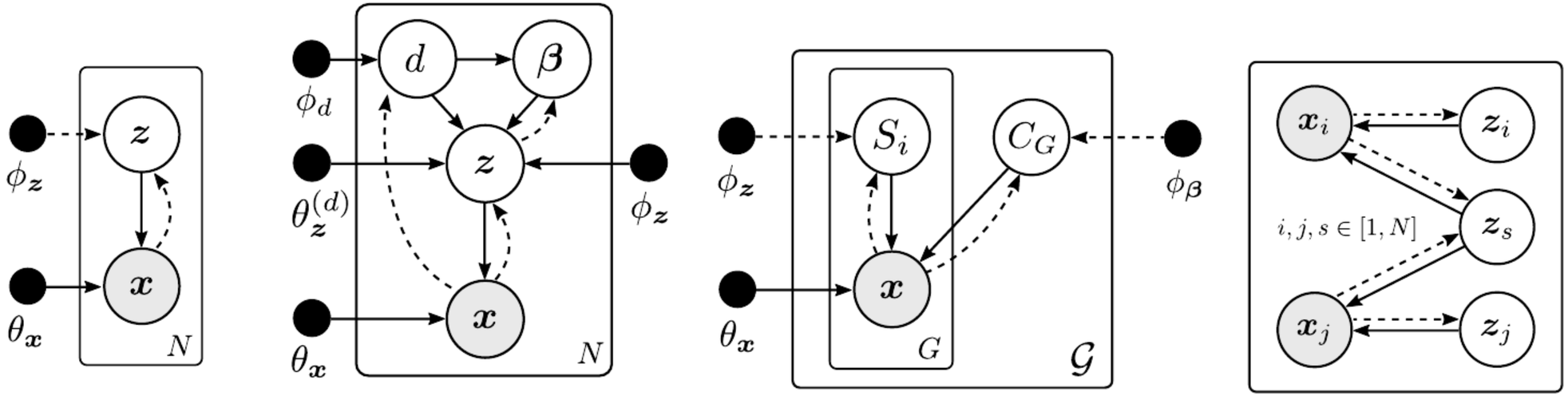
- Why we would want to capture this shared information?
  - Discover global properties in patients
  - Global factors in climate data
  - ...



Problem Statement

Limitations of Existing Approaches

- Existing approaches are limited to “expensive” semi-supervision.



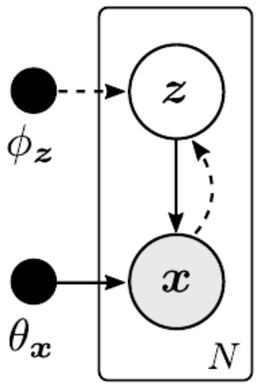
(a) VAE [1] (b) GMVAE [10] (c) ML-VAE [20] (d) NestedVAE

[1] (Kingma et al., 2013) [10] (Dilokthanakul et al., 2016) [20] (Bouchacourt et al., 2018) [27] (Vowels et al., 2020)

Problem Statement

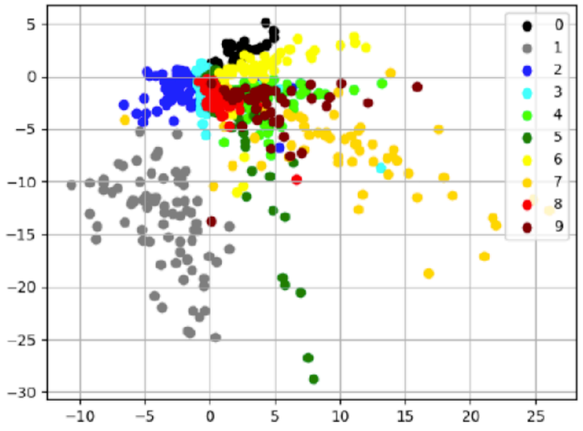
Limitations of Existing Approaches

- Vanilla VAE learns to generate *i.i.d.* data.



(a) VAE

The MNIST dataset

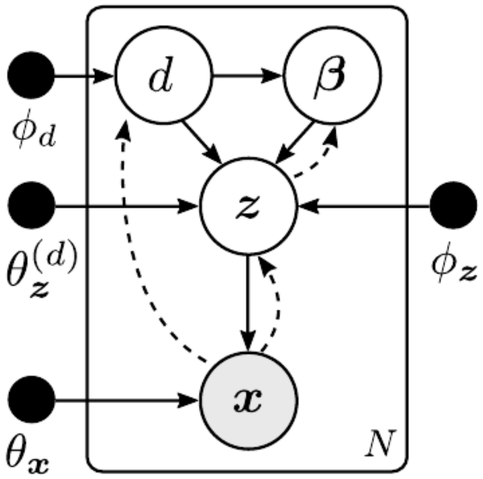


VAE

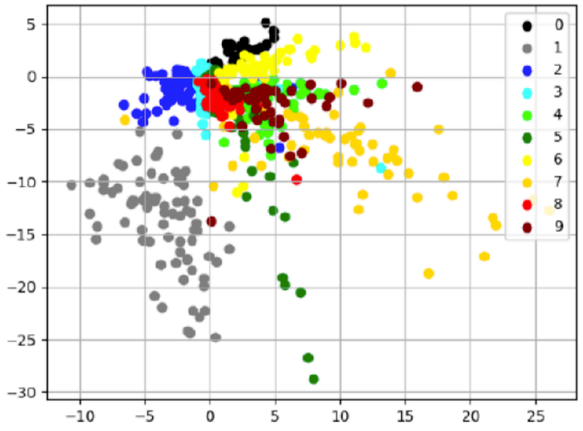
Problem Statement

Limitations of Existing Approaches

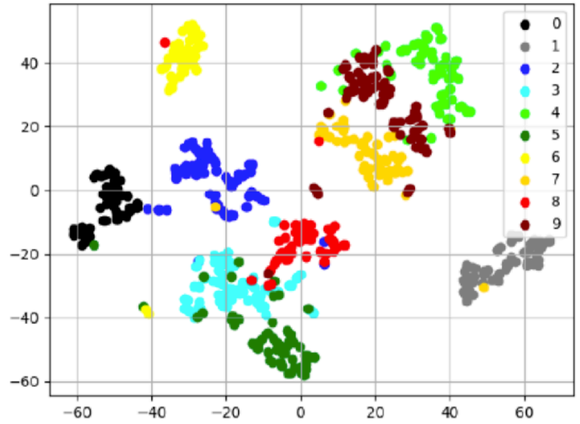
- Incorporating mixture models in the latent space of a VAE increases flexibility, but don't explicitly generate shared global properties.



(b) GMVAE



VAE

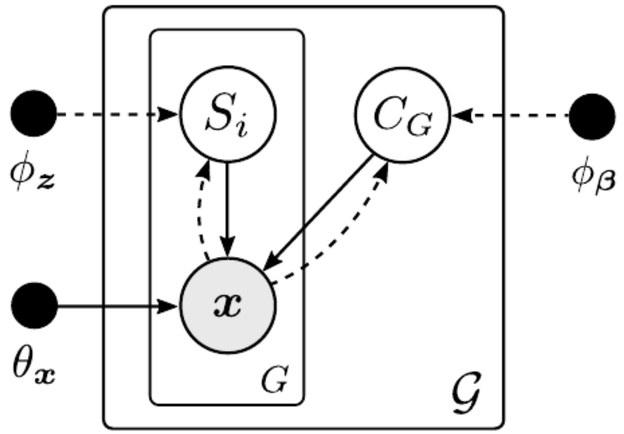


GMVAE

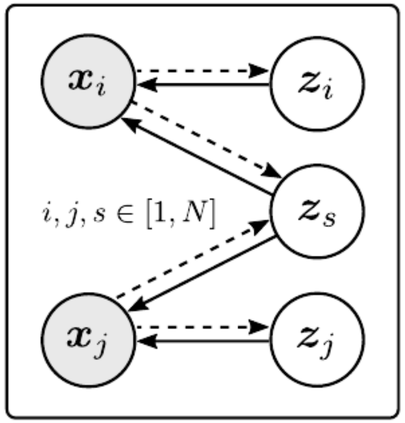
Problem Statement

Limitations of Existing Approaches

- More related models capture shared properties via semi-supervision.



(c) ML-VAE

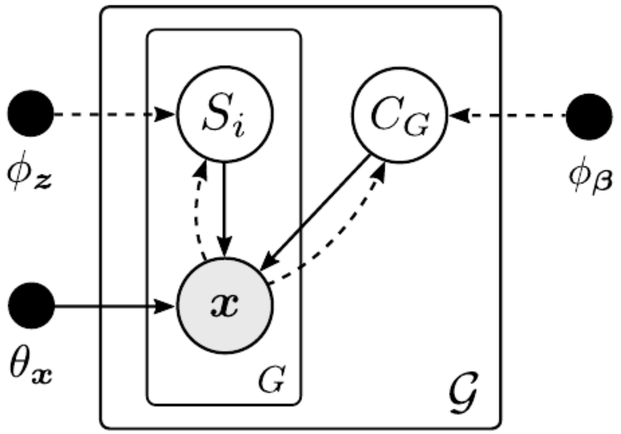


(d) NestedVAE

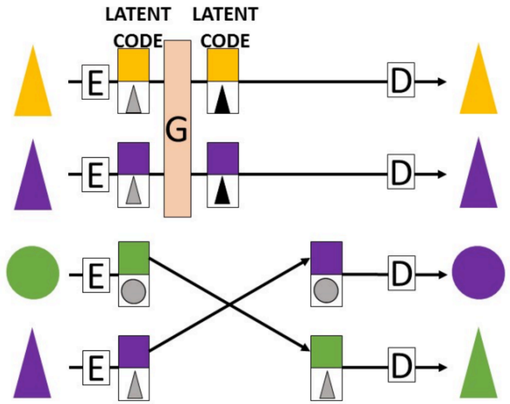
Problem Statement

Limitations of Existing Approaches

- More related models capture shared properties via semi-supervision.



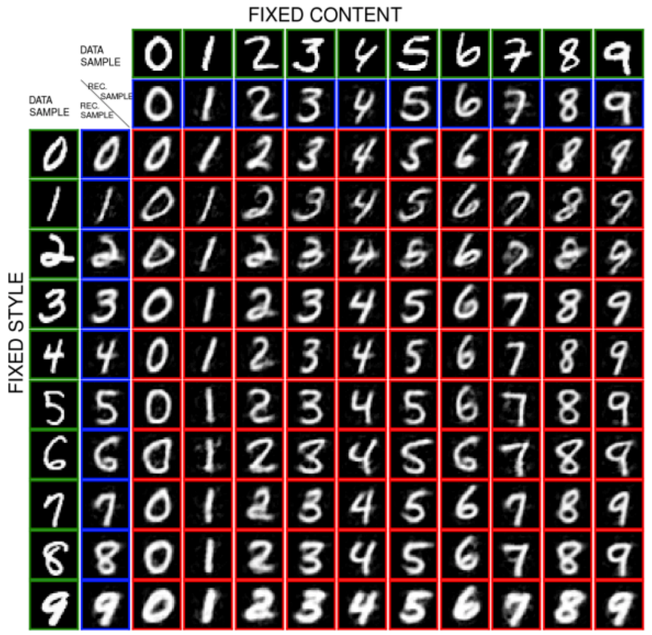
(c) ML-VAE

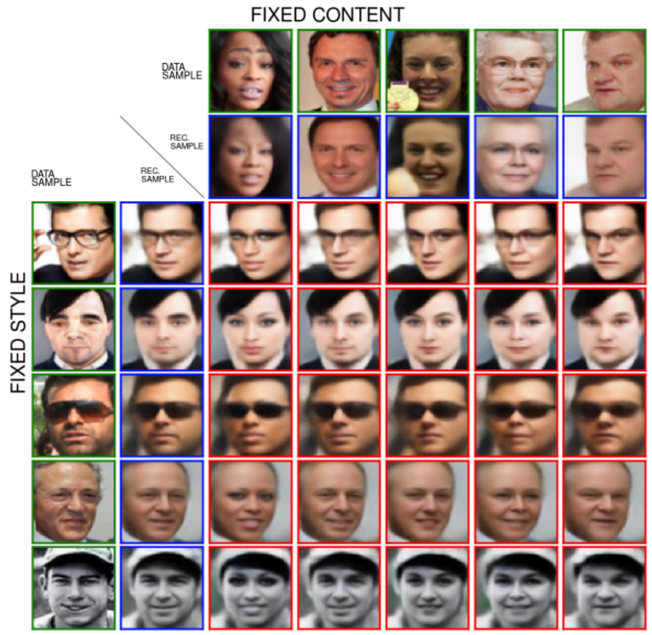


Problem Statement

Limitations of Existing Approaches

- More related models capture shared properties via semi-supervision.



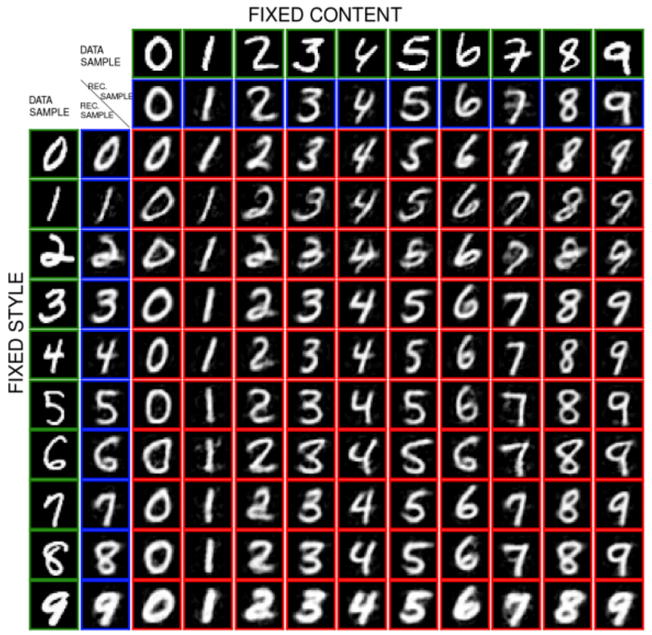


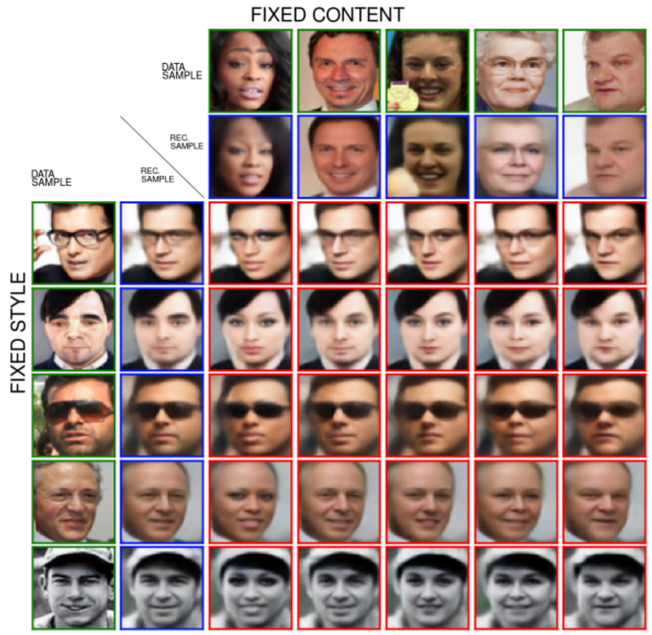
Problem Statement

Limitations of Existing Approaches

- More related models capture shared properties via semi-supervision.

❌ The group each data belongs **needs** to is **observed**.



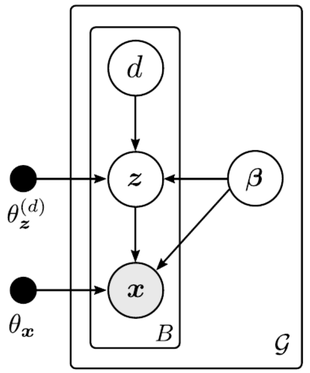


Contribution

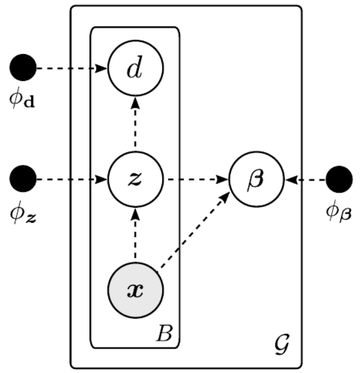
The Unsupervised Global VAE

- None of the existing approaches capture global interdependencies in an **unsupervised** fashion.
- Can we build a VAE that learns this shared information without any kind of supervision?

  ✓ We present the **Unsupervised Global VAE (UG-VAE)**.



(a) Generative model



(b) Inference model

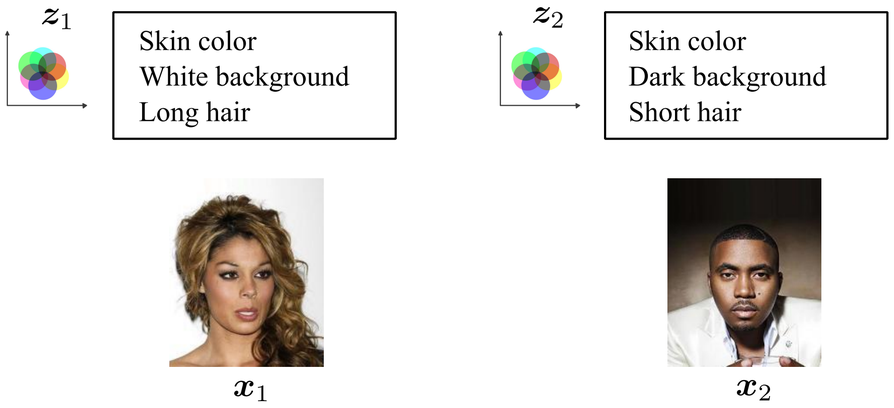
Model Description

Generative model

Model Description

Generative model

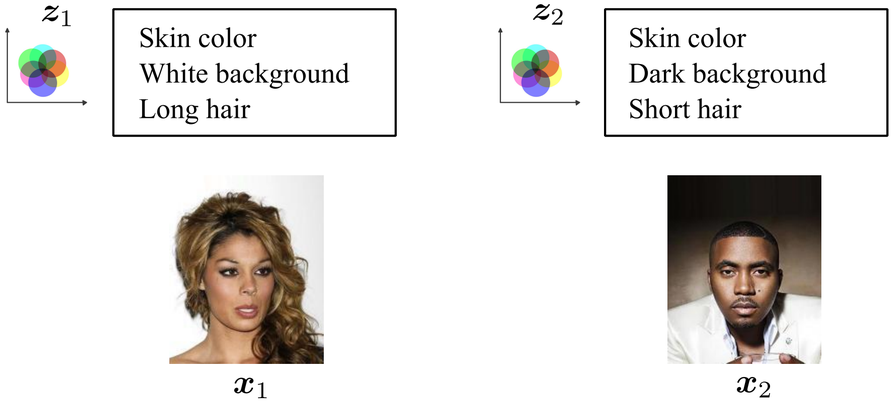
β-VAE



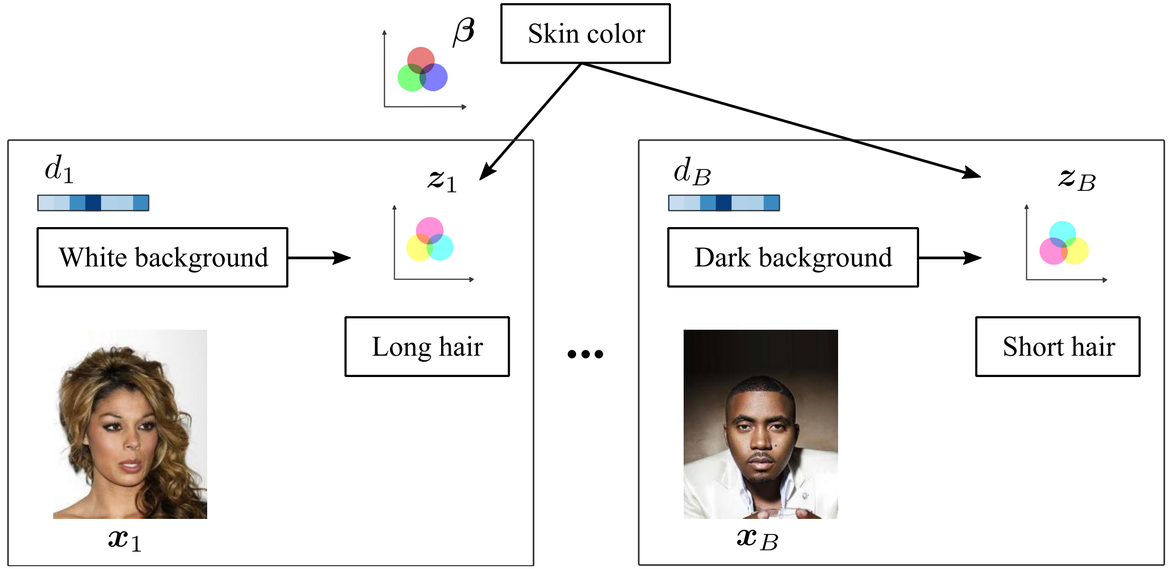
Model Description

Generative model

β-VAE

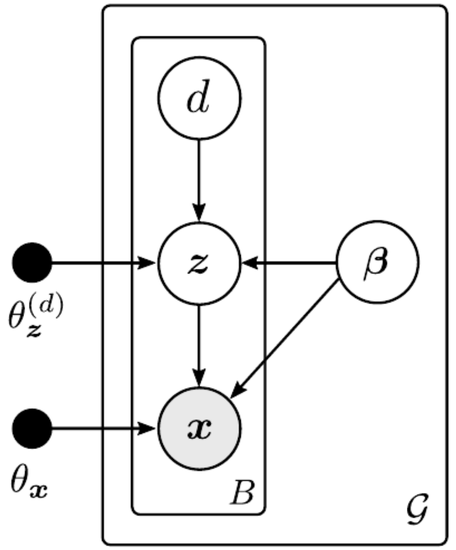


UG-VAE



Model Description

Generative model



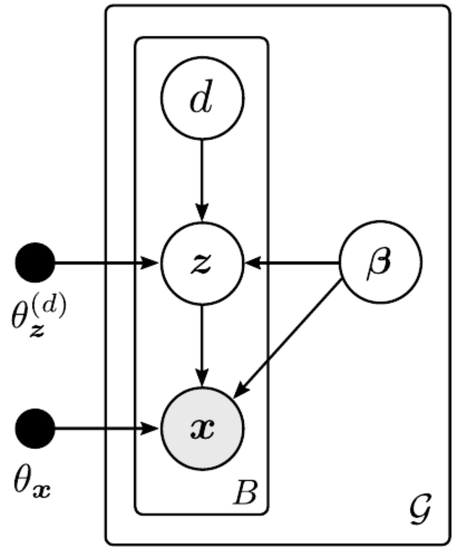
(a) Generative model

Model Description

Generative model

- The generative model for a **batch** is

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$



(a) Generative model

Model Description

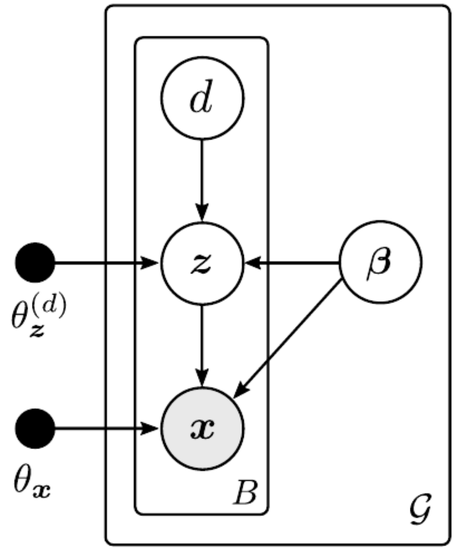
Generative model

- The generative model for a **batch** is

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$

$$p(\boldsymbol{d}) = \prod_{i=1}^{B} \mathrm{Cat}(\boldsymbol{\pi}) \quad \pi_k = 1/K$$

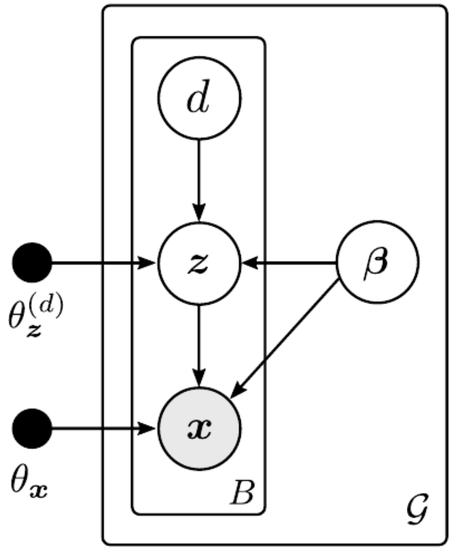
$$p(\boldsymbol{\beta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$



(a) Generative model

Model Description

Generative model

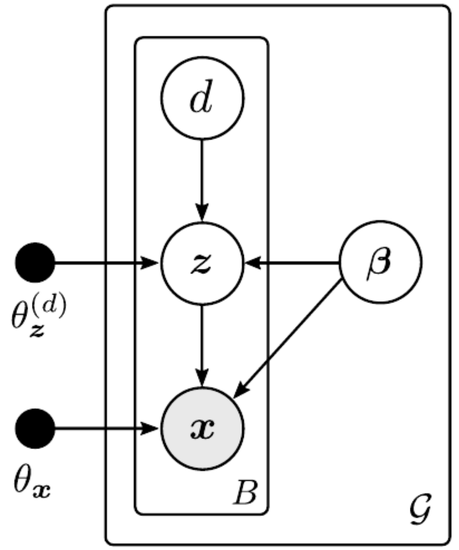


(a) Generative model

Model Description

Generative model

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$



(a) Generative model

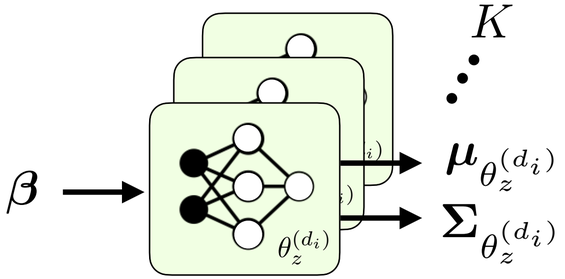
Model Description

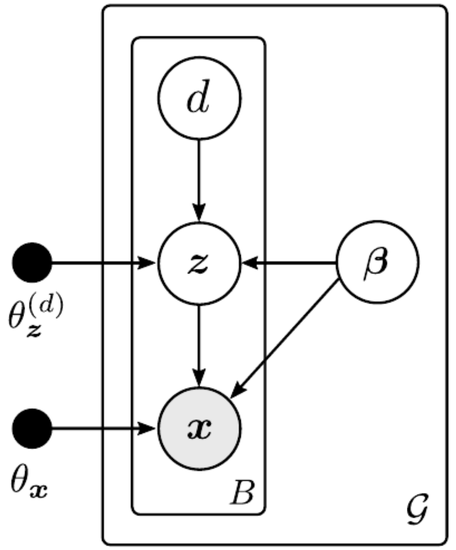
Generative model

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$

- The prior of the **local representation** is a Gaussian mixture, modulated by $\boldsymbol{\beta}$

$$p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) = \prod_{i=1}^{B} p_{\theta_z}(\boldsymbol{z}_i|d_i, \boldsymbol{\beta}) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\theta_z}^{(d_i)}(\boldsymbol{\beta}), \boldsymbol{\Sigma}_{\theta_z}^{(d_i)}(\boldsymbol{\beta})\right)$$



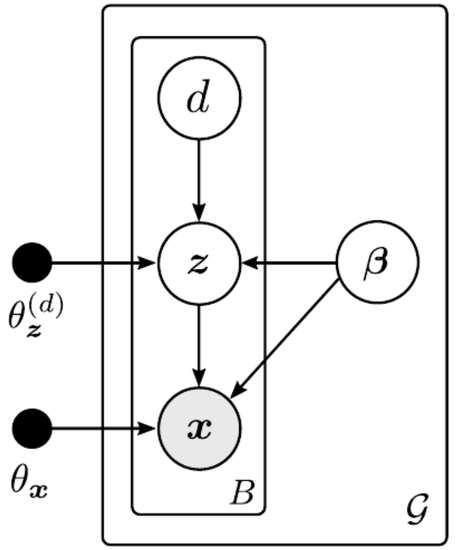


(a) Generative model

- The mixture increases flexibility of the local space.

Model Description

Generative model



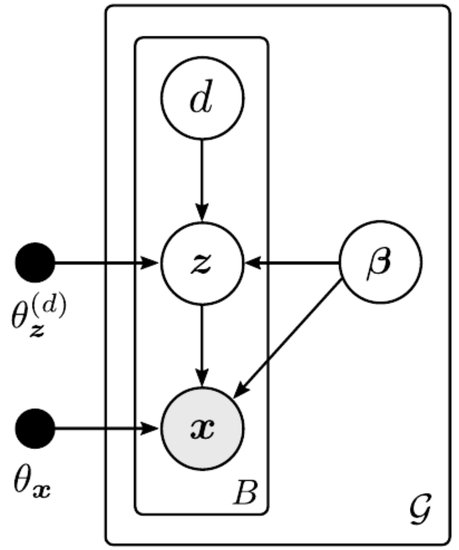
(a) Generative model

Model Description

Generative model

- The generative model for a **batch** is

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$



(a) Generative model

Model Description

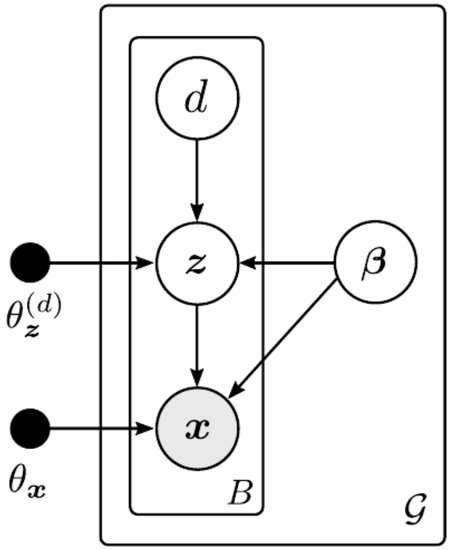
Generative model

- The generative model for a **batch** is

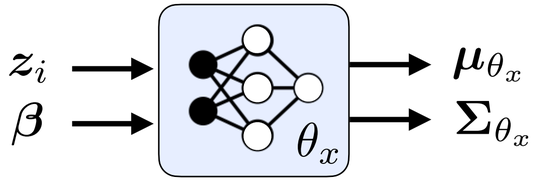
$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$

- Where the **likelihood model** of each datapoint is a Gaussian

$$p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) = \prod_{i=1}^{B} p_{\theta_x}(\boldsymbol{x}_i|\boldsymbol{z}_i, \boldsymbol{\beta}) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\theta_x}([\boldsymbol{z}_i, \boldsymbol{\beta}]), \boldsymbol{\Sigma}_{\theta_x}([\boldsymbol{z}_i, \boldsymbol{\beta}])\right)$$



(a) Generative model



Model Description

Generative model

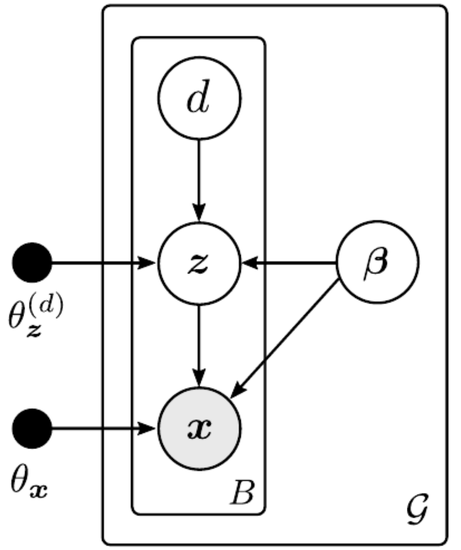
- The generative model for a **batch** is

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) p_{\theta_z}(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta}) p(\boldsymbol{d}) p(\boldsymbol{\beta})$$

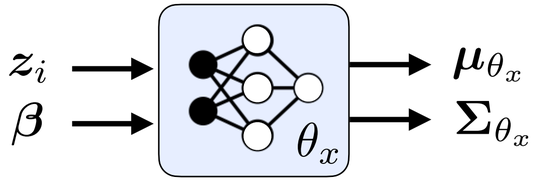
- Where the **likelihood model** of each datapoint is a Gaussian

$$p_{\theta_x}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) = \prod_{i=1}^{B} p_{\theta_x}(\boldsymbol{x}_i|\boldsymbol{z}_i, \boldsymbol{\beta}) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\theta_x}\left([\overline{\boldsymbol{z}_i}, \overline{\boldsymbol{\beta}}]\right), \boldsymbol{\Sigma}_{\theta_x}\left([\overline{\boldsymbol{z}_i}, \overline{\boldsymbol{\beta}}]\right)\right)$$

  - The batch $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_B\}$ shares the **global** representation $\boldsymbol{\beta}$.
  - Every point $\boldsymbol{x}_i$ has a conditional **local** representation $\boldsymbol{z}_i$.

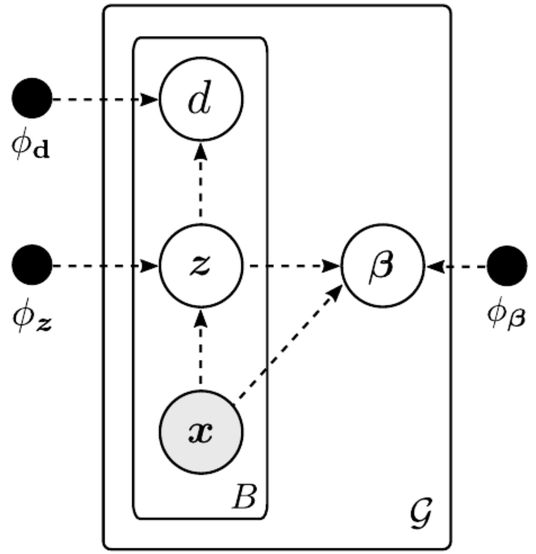


(a) Generative model



Model Description

Inference model

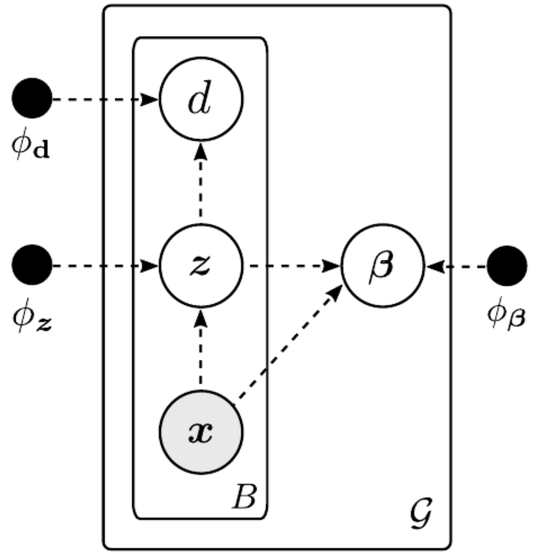


(b) Inference model

Model Description

Inference model

$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z}) q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z})$$



(b) Inference model

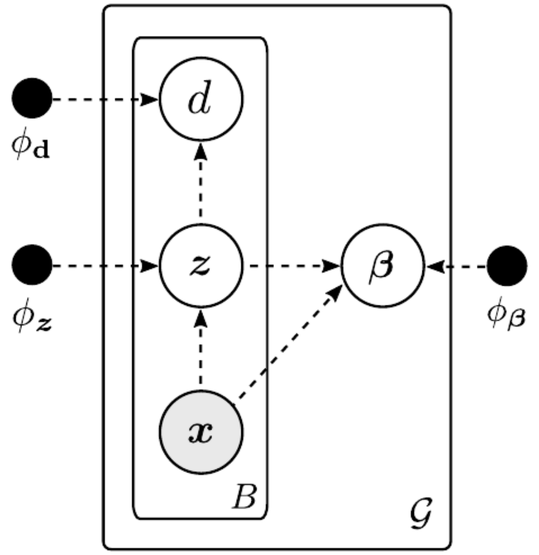
Model Description

Inference model

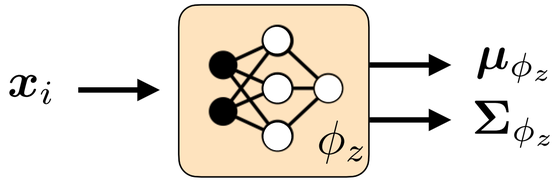
$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z}) q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z})$$

- The *local encoder* parameterises the Gaussian posterior

$$q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) = \prod_{i=1}^{B} q_{\phi_z}(\boldsymbol{z}_i|\boldsymbol{x}_i) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\phi_z}(\boldsymbol{x}_i), \boldsymbol{\Sigma}_{\phi_z}(\boldsymbol{x}_i)\right)$$



(b) Inference model



Model Description

Inference model

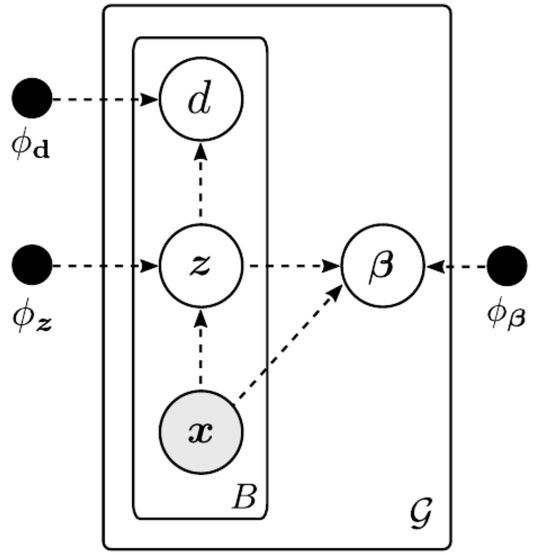
$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X})\, q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z})\, q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z})$$

- The *local encoder* parameterises the Gaussian posterior

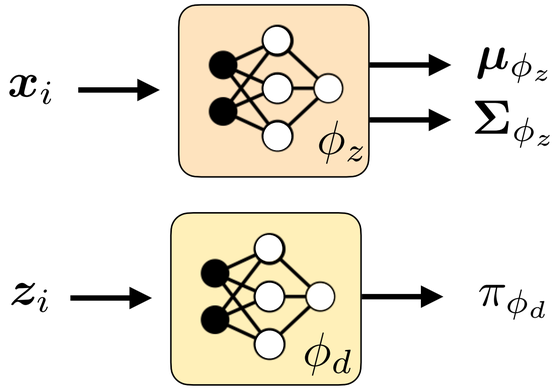
$$q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) = \prod_{i=1}^{B} q_{\phi_z}(\boldsymbol{z}_i|\boldsymbol{x}_i) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\phi_z}(\boldsymbol{x}_i), \boldsymbol{\Sigma}_{\phi_z}(\boldsymbol{x}_i)\right)$$

- A second *local encoder* parameterises the posterior of the components of the mixture

$$q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z}) = \prod_{i=1}^{B} q_{\phi_d}(d_i|\boldsymbol{z}_i) = \prod_{i=1}^{B} \mathrm{Cat}\left(\pi_{\phi_d}(\boldsymbol{z}_i)\right)$$

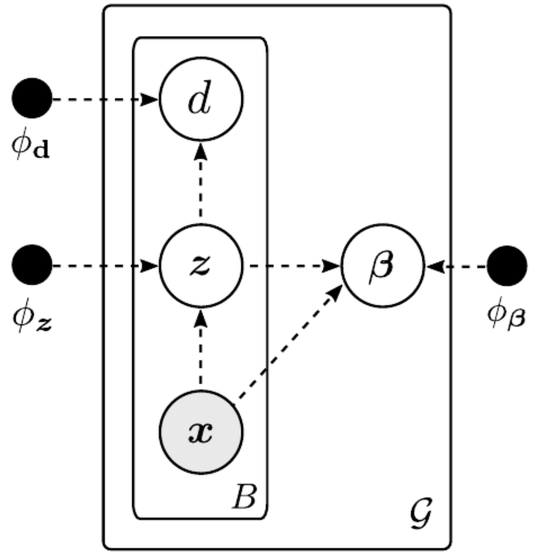


(b) Inference model



Model Description

Inference model

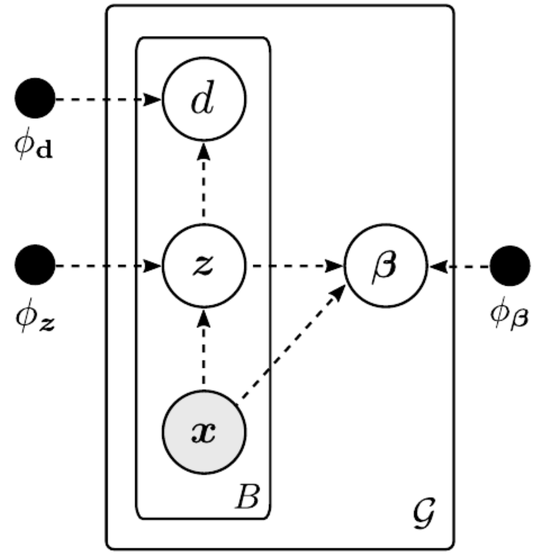


(b) Inference model

Model Description

Inference model

$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z}) q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z})$$



(b) Inference model

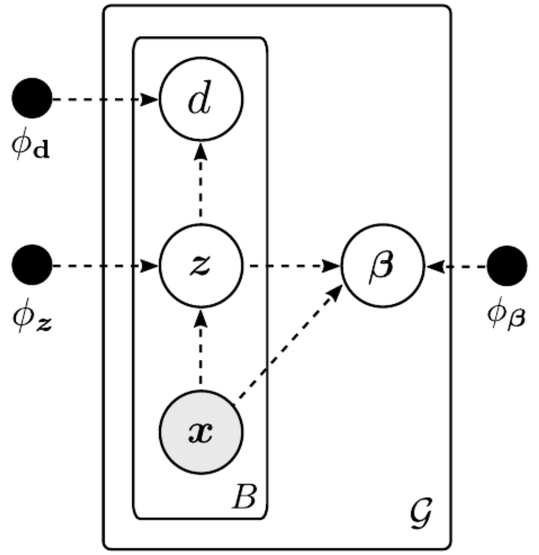
Model Description

Inference model

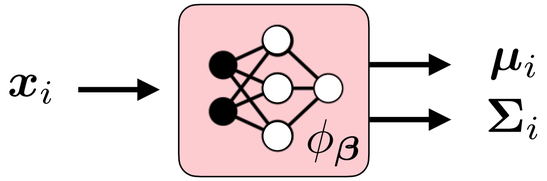
$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z}) q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z})$$

- The global *encoder* requires **exchangeability**

$$q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) = \frac{1}{Z} \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\phi_\beta}\left([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)]\right), \boldsymbol{\Sigma}_{\phi_\beta}\left([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)]\right)\right\}$$



(b) Inference model



Model Description

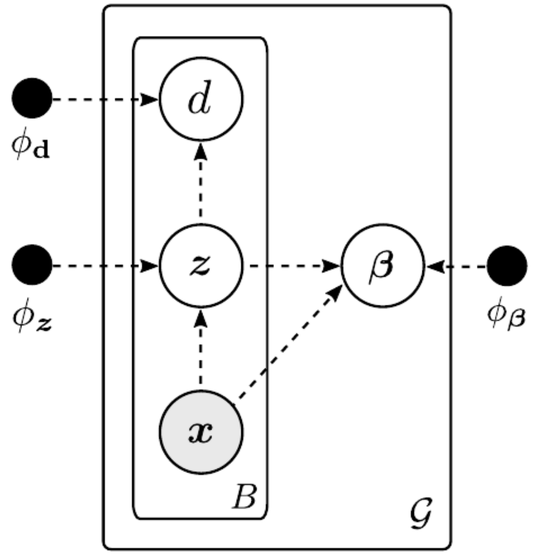
Inference model

$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q_{\phi_z}(\boldsymbol{Z}|\boldsymbol{X}) q_{\phi_d}(\boldsymbol{d}|\boldsymbol{Z}) q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z})$$

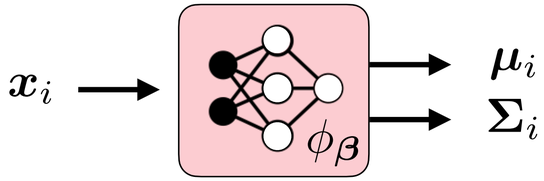
- The global *encoder* requires **exchangeability**

$$q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) = \frac{1}{Z} \prod_{i=1}^{B} \mathcal{N}\left(\overbrace{\boldsymbol{\mu}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\mu}_i}, \overbrace{\boldsymbol{\Sigma}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\Sigma}_i}\right)$$

  - **Product of Gaussian** local contributions.



(b) Inference model



Model Description

Product of Gaussians

[28] (Bromiley, 2003)

Model Description

Product of Gaussians

$$q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X},\boldsymbol{Z}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) = \frac{1}{Z}\prod_{i=1}^{B} \mathcal{N}\left(\overbrace{\boldsymbol{\mu}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\mu}_i}, \overbrace{\boldsymbol{\Sigma}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\Sigma}_i}\right)$$

- The parameters of the product can be expressed as ([28]):

$$\boldsymbol{\Lambda}_\beta = \boldsymbol{\Sigma}_\beta^{-1} = \sum_{i=1}^{B} \boldsymbol{\Lambda}_i$$

$$\boldsymbol{\mu}_\beta = (\boldsymbol{\Lambda}_\beta)^{-1} \sum_{i=1}^{B} \boldsymbol{\Lambda}_i \boldsymbol{\mu}_i$$

[28] (Bromiley, 2003)

Model Description

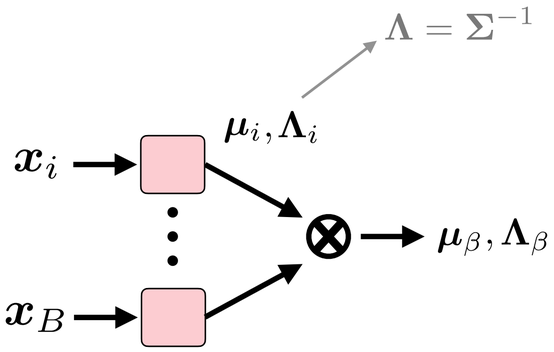
Product of Gaussians

$$q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X},\boldsymbol{Z}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) = \frac{1}{Z}\prod_{i=1}^{B} \mathcal{N}\Big(\overbrace{\boldsymbol{\mu}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\mu}_i}, \overbrace{\boldsymbol{\Sigma}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\Sigma}_i}\Big)$$

- The parameters of the product can be expressed as ([28]):

$$\boldsymbol{\Lambda}_\beta = \boldsymbol{\Sigma}_\beta^{-1} = \sum_{i=1}^{B} \boldsymbol{\Lambda}_i$$

$$\boldsymbol{\mu}_\beta = (\boldsymbol{\Lambda}_\beta)^{-1} \sum_{i=1}^{B} \boldsymbol{\Lambda}_i \boldsymbol{\mu}_i$$



[28] (Bromiley, 2003)

Model Description

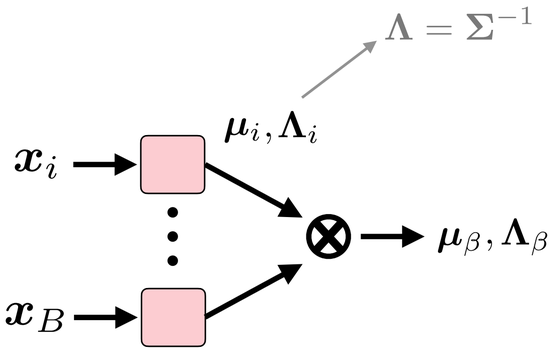
Product of Gaussians

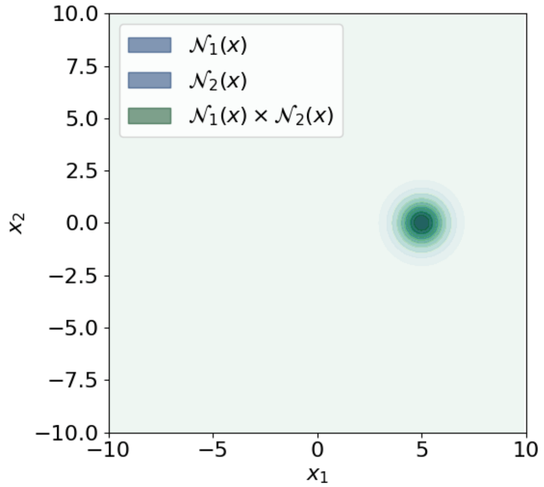
$$q_{\phi_\beta}(\boldsymbol{\beta}|\boldsymbol{X},\boldsymbol{Z}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) = \frac{1}{Z}\prod_{i=1}^{B} \mathcal{N}\Big(\overbrace{\boldsymbol{\mu}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\mu}_i}, \overbrace{\boldsymbol{\Sigma}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])}^{\boldsymbol{\Sigma}_i}\Big)$$

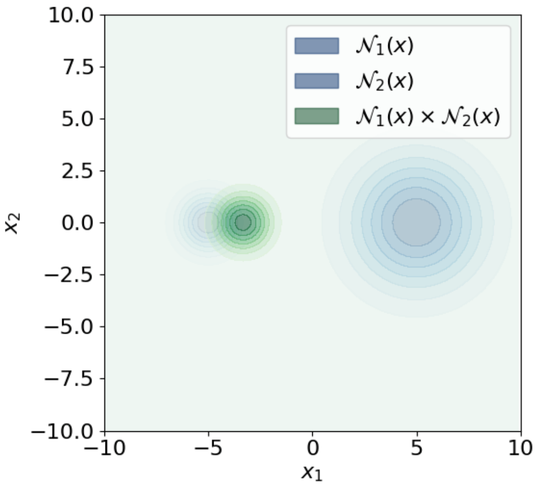
- The parameters of the product can be expressed as ([28]):

$$\boldsymbol{\Lambda}_\beta = \boldsymbol{\Sigma}_\beta^{-1} = \sum_{i=1}^{B} \boldsymbol{\Lambda}_i$$

$$\boldsymbol{\mu}_\beta = (\boldsymbol{\Lambda}_\beta)^{-1} \sum_{i=1}^{B} \boldsymbol{\Lambda}_i \boldsymbol{\mu}_i$$







[28] (Bromiley, 2003)

Experiments

Unsupervised Learning of Global Factors

Experiments

Unsupervised Learning of Global Factors

Method	UG-VAE	ML-VAE	β-VAE
CelebA	$\mathbf{162.3 \pm 1.2}$	204.7 ± 2.4	173.5 ± 0.6
MNIST	$\mathbf{63.6 \pm 2.4}$	108.9 ± 4.5	133.2 ± 0.8

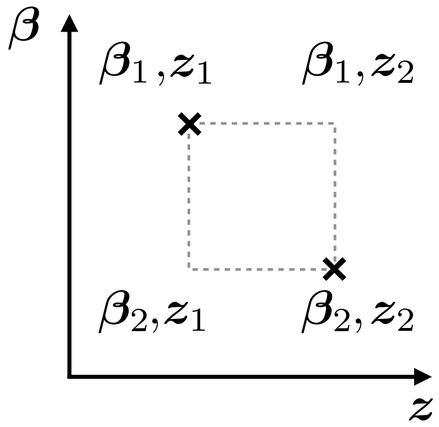
FID Scores

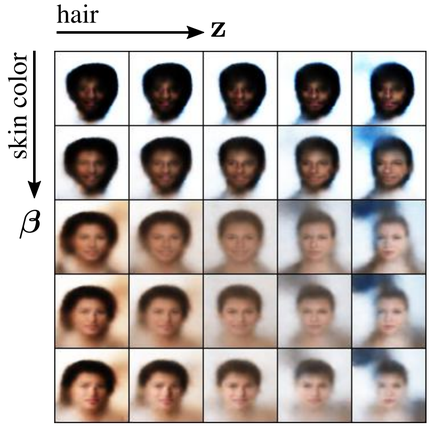
Experiments

Unsupervised Learning of Global Factors

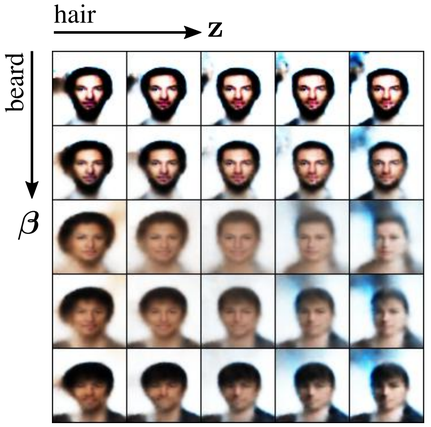
Method	UG-VAE	ML-VAE	β-VAE
CelebA	$\mathbf{162.3 \pm 1.2}$	204.7 ± 2.4	173.5 ± 0.6
MNIST	$\mathbf{63.6 \pm 2.4}$	108.9 ± 4.5	133.2 ± 0.8

FID Scores

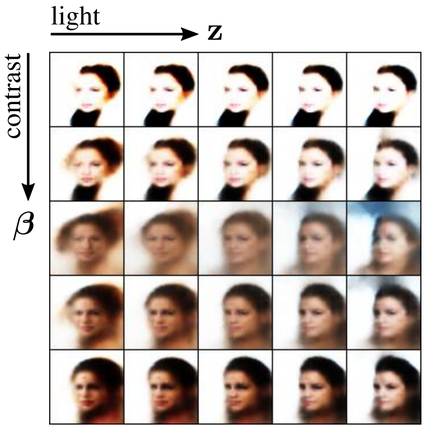




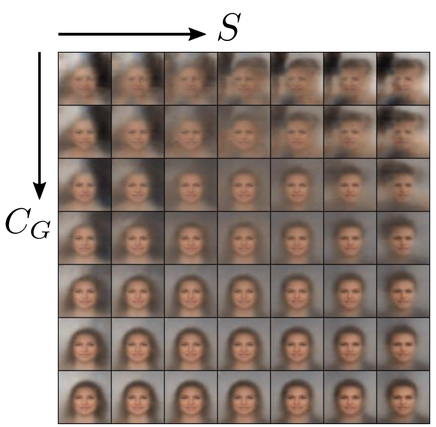
$d = 5$



$d = 7$



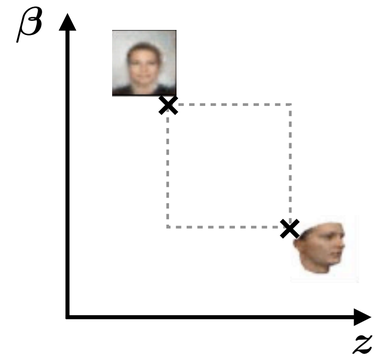
$d = 15$

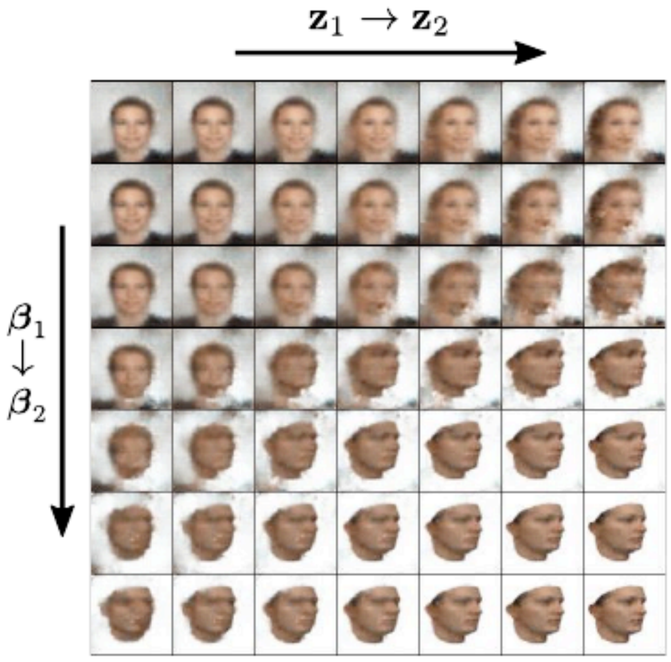


ML-VAE

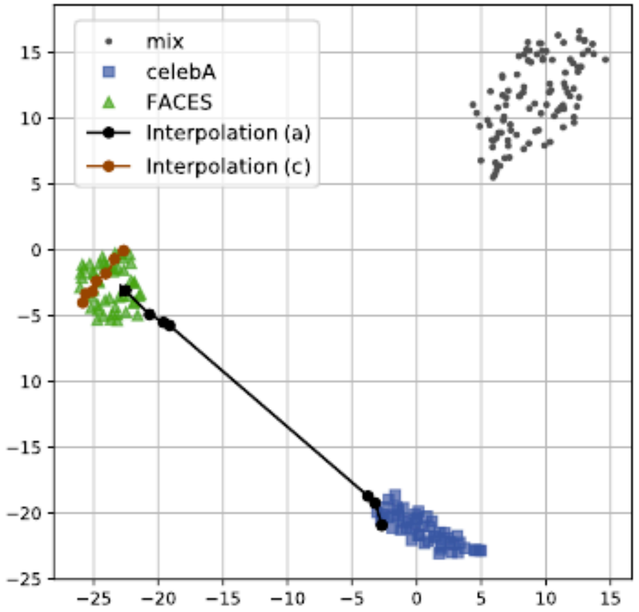
Experiments

Domain Alignment

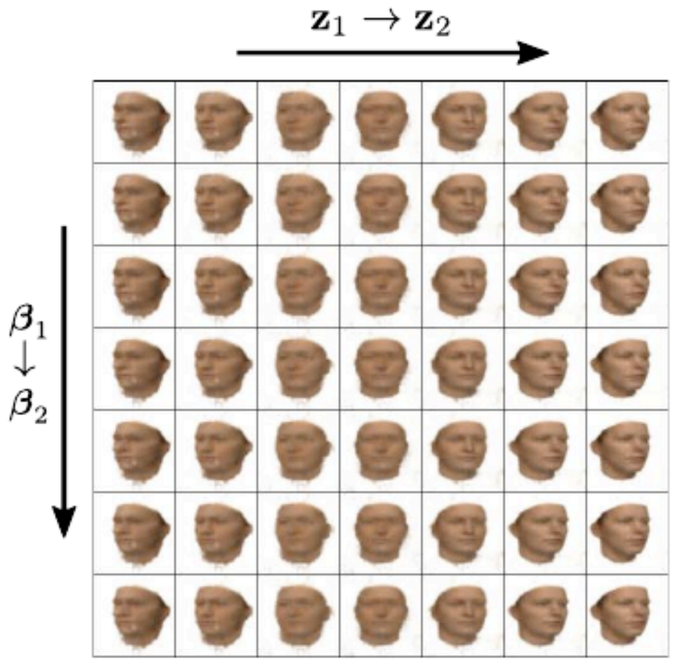




(a) CelebA-FACES



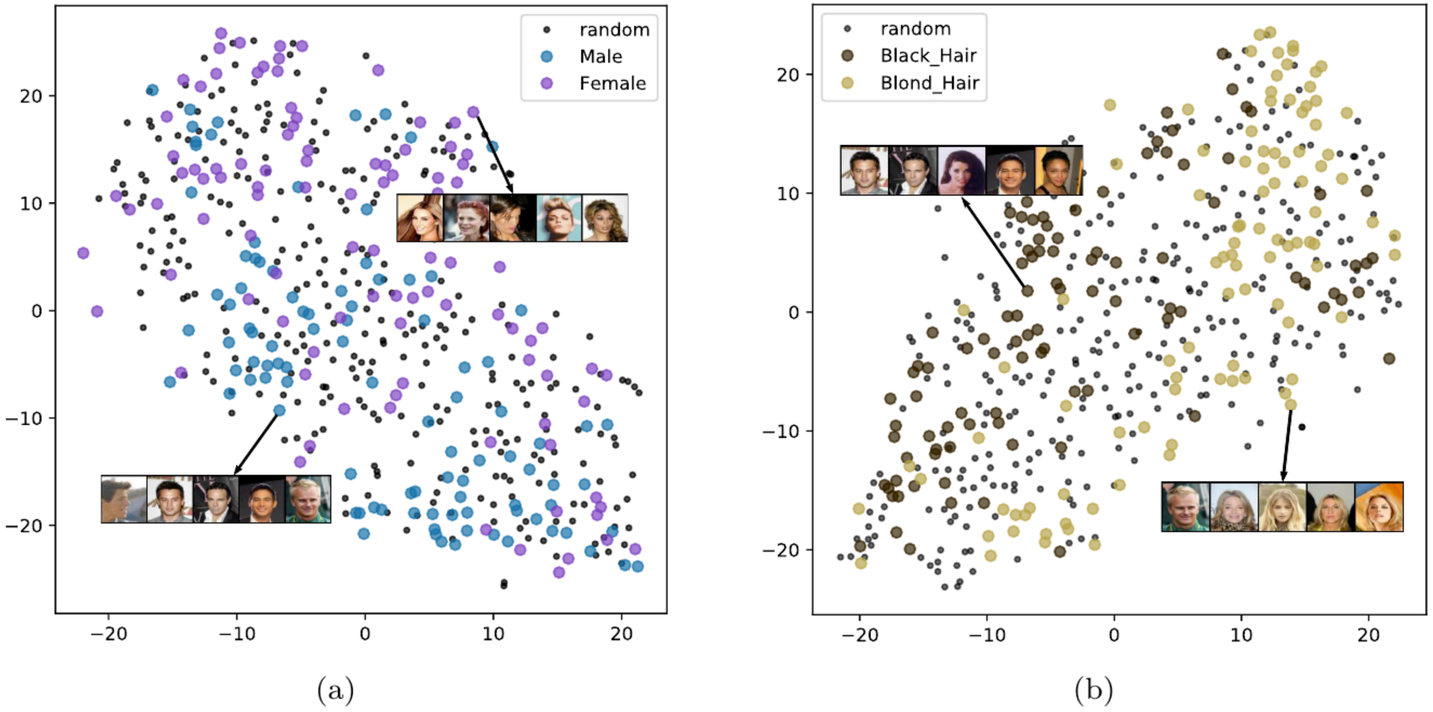
(b) $\boldsymbol{\beta}$ TSNE 2D space.



(c) FACES-FACES

Experiments

Global Representation of Structured Batches



Experiments

Global Representation of Structured Batches

Batch categories	Classifier	Train accuracy	Test accuracy
Black (0) vs blond (1)	Linear SVM	1.0	0.95
	RBF SVM	1.0	0.98
Black (0) vs blond (1) vs random (2)	Linear SVM	0.91	0.54
	RBF SVM	0.85	0.56
Male (0) vs female (1)	Linear SVM	1.0	0.85
	RBF SVM	1.0	0.85
Male (0) vs female (1) vs random (2)	Linear SVM	0.84	0.66
	RBF SVM	0.89	0.63

Conclusion

Based on the provided experiments, we evidence that:

- VAEs can be used to learn global shared information in an unsupervised manner.
- The learned global representations are interpretable with UG-VAE.
- Potential applications in clinical data, climate data, etc.

I. Peis, P. M. Olmos and A. Artés-Rodríguez. Unsupervised Learning of Global Factors in Deep Generative Models. In ***Pattern Recognition***, *134, 109130*, 2023.

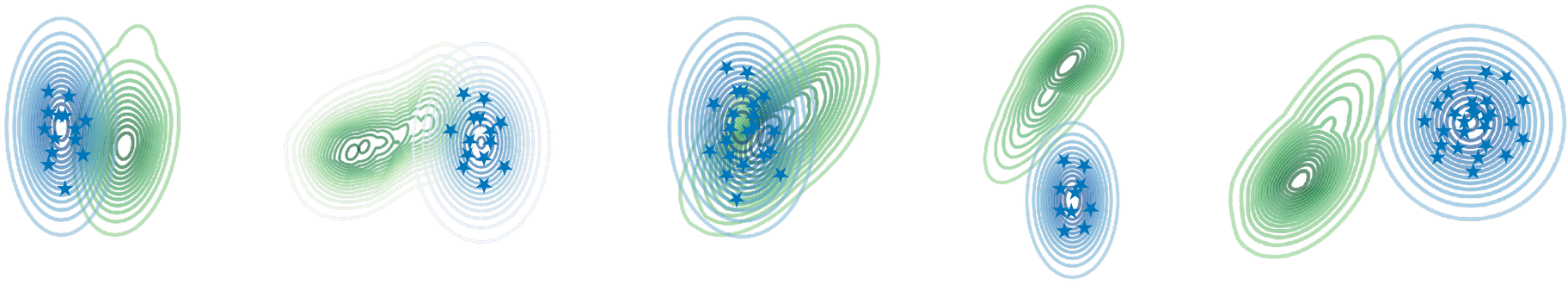
Outline

Contents

Contribution II: Hierarchical VAEs and Hamiltonian Monte Carlo

Problem Statement

Improved MCMC inference



$q(\boldsymbol{z}|\boldsymbol{x})$ $p(\boldsymbol{z}|\boldsymbol{x})$

[4]

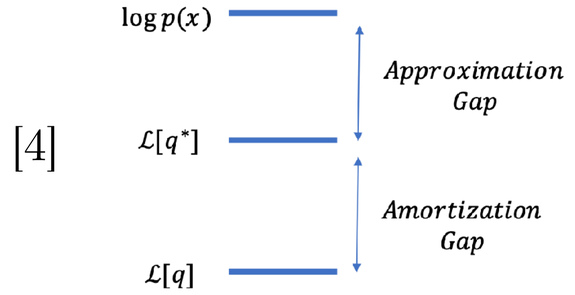
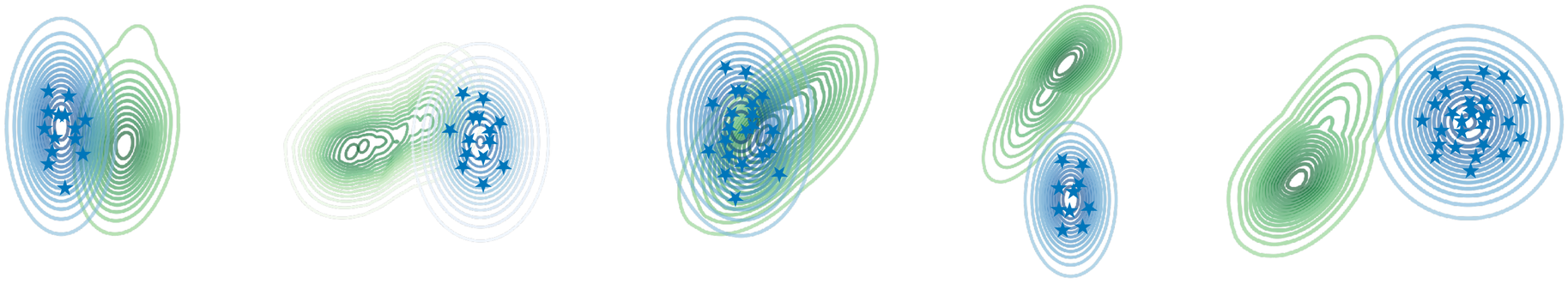


Figure 1. Gaps in Inference

[4] (Cremer et at., 2018) [9] (Campbell et at., 2021)

Problem Statement

Improved MCMC inference



$q(\boldsymbol{z}|\boldsymbol{x})$ $p(\boldsymbol{z}|\boldsymbol{x})$

- Can we get better samples that follow the green contour?

[4]

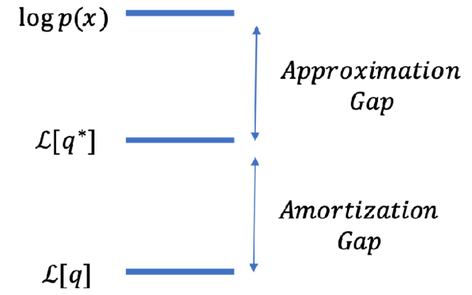
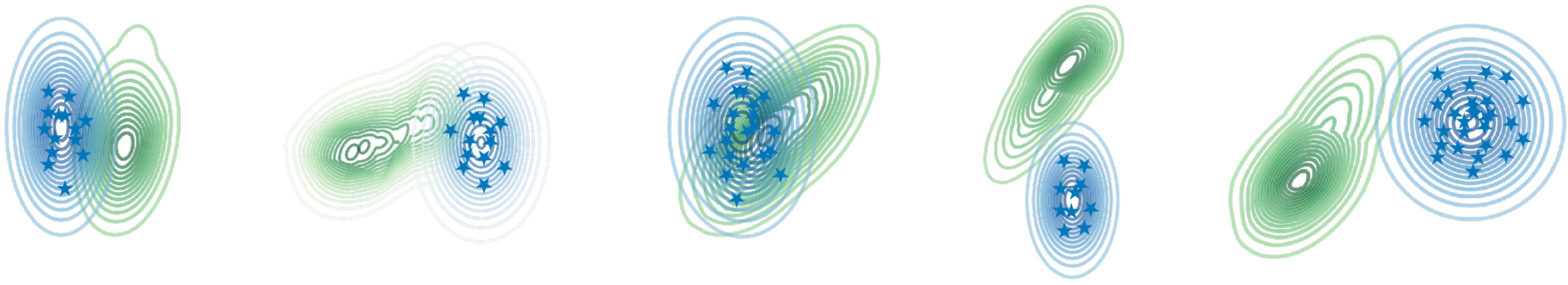


Figure 1. Gaps in Inference

[4] (Cremer et at., 2018) [9] (Campbell et at., 2021)

Problem Statement

Improved MCMC inference



$q(\boldsymbol{z}|\boldsymbol{x})$ $p(\boldsymbol{z}|\boldsymbol{x})$

- Can we get better samples that follow the green contour? ✓ [9]

[4]

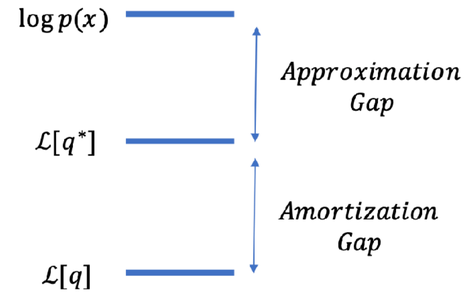


Figure 1. Gaps in Inference

[4] (Cremer et at., 2018) [9] (Campbell et at., 2021)

Problem Statement

Improved MCMC inference in Hierarchical VAEs

[5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)

Problem Statement

Improved MCMC inference in Hierarchical VAEs

- One-layered VAEs approximate inference can be improved via Markov Chain Monte Carlo [5-9].

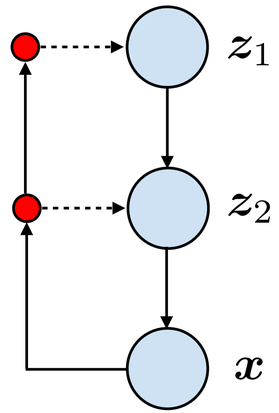
[5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)

Problem Statement

Improved MCMC inference in Hierarchical VAEs

- One-layered VAEs approximate inference can be improved via Markov Chain Monte Carlo [5-9].

  1. Could we leverage MCMC methods for Hierarchical VAEs?



[5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)

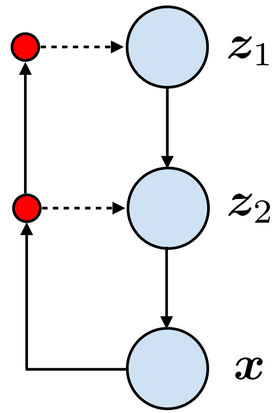
Problem Statement

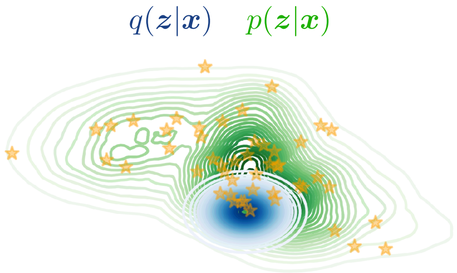
Improved MCMC inference in Hierarchical VAEs

- One-layered VAEs approximate inference can be improved via Markov Chain Monte Carlo [5-9].

  1. Could we leverage MCMC methods for Hierarchical VAEs?

  2. If so, could we improve incomplete data handling with MCMC?





HMC samples (orange)

[5] (Burda et at., 2015) [6] (Salimans et at., 2015) [7] (Ruiz et at., 2021) [8] (Caterini et at., 2018) [9] (Campbell et at., 2021)

Problem Statement

Improve missing data imputation with VAEs

[29] (Ma et at., 2018) [30] (Ma et at., 2020) [31] (Nazabal et at., 2020) [32] (Mattei et at., 2020)

Problem Statement

Improve missing data imputation with VAEs

- Imputation under a VAE framework [29-32]:

$$p(\boldsymbol{x}_U|\boldsymbol{x}_O) = \mathbb{E}_{p(\boldsymbol{z}|\boldsymbol{x}_O)}[p(\boldsymbol{x}_U|\boldsymbol{z})] \approx \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x}_O)}[p(\boldsymbol{x}_U|\boldsymbol{z})]$$

[29] (Ma et at., 2018) [30] (Ma et at., 2020) [31] (Nazabal et at., 2020) [32] (Mattei et at., 2020)

Problem Statement

Improve missing data imputation with VAEs

- Imputation under a VAE framework [29-32]:

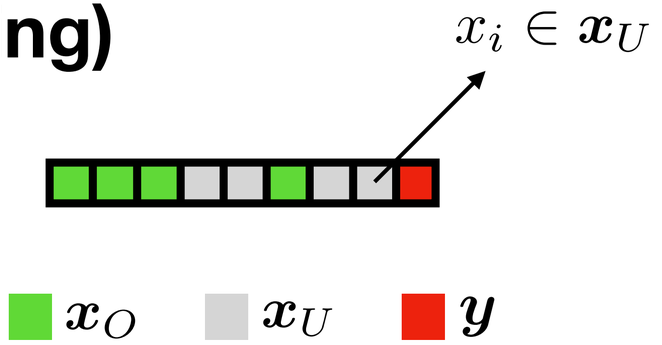
$$p(\boldsymbol{x}_U|\boldsymbol{x}_O) = \mathbb{E}_{p(\boldsymbol{z}|\boldsymbol{x}_O)}[p(\boldsymbol{x}_U|\boldsymbol{z})] \approx \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x}_O)}[p(\boldsymbol{x}_U|\boldsymbol{z})]$$

- Also based on **Gaussian** approximations of the true posterior.

[29] (Ma et at., 2018) [30] (Ma et at., 2020) [31] (Nazabal et at., 2020) [32] (Mattei et at., 2020)

Problem Statement

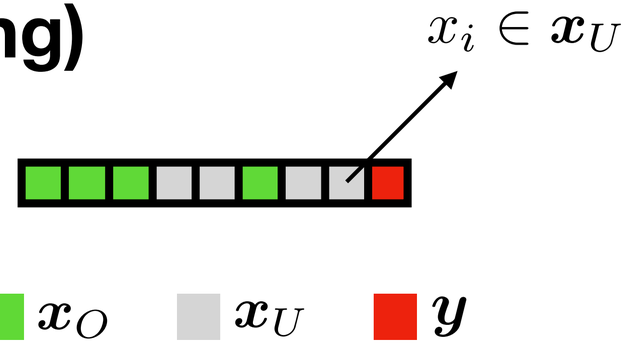
Discovery of high-value information (Active Learning)



[33] (Bernardo et at., 1979) [29] (Ma et at., 2018) [30] (Ma et at., 2020)

Problem Statement

Discovery of high-value information (Active Learning)



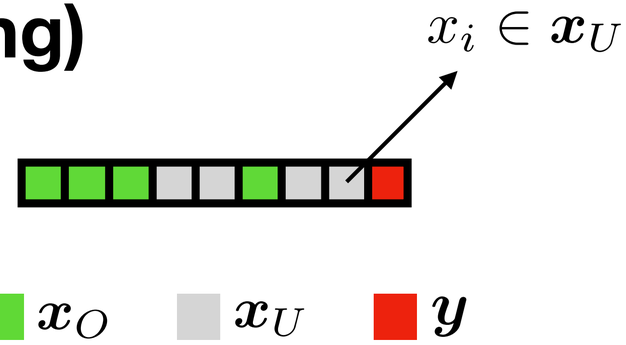
- Bayesian reward function [33] as an expected gain of information:

$$R\left(i, \boldsymbol{x}_O\right) = \mathbb{E}_{\boldsymbol{x}_i \sim p(\boldsymbol{x}_i|\boldsymbol{x}_O)} D_{\mathrm{KL}}\left[p\left(\boldsymbol{y}|\boldsymbol{x}_i, \boldsymbol{x}_O\right) || \, p\left(\boldsymbol{y}|\boldsymbol{x}_O\right)\right]$$

[33] (Bernardo et at., 1979) [29] (Ma et at., 2018) [30] (Ma et at., 2020)

Problem Statement

Discovery of high-value information (Active Learning)



- Bayesian reward function [33] as an expected gain of information:

$$R\left(i, \boldsymbol{x}_O\right) = \mathbb{E}_{\boldsymbol{x}_i \sim p\left(\boldsymbol{x}_i | \boldsymbol{x}_O\right)} D_{\mathrm{KL}}\left[p\left(\boldsymbol{y} | \boldsymbol{x}_i, \boldsymbol{x}_O\right) || \, p\left(\boldsymbol{y} | \boldsymbol{x}_O\right)\right]$$

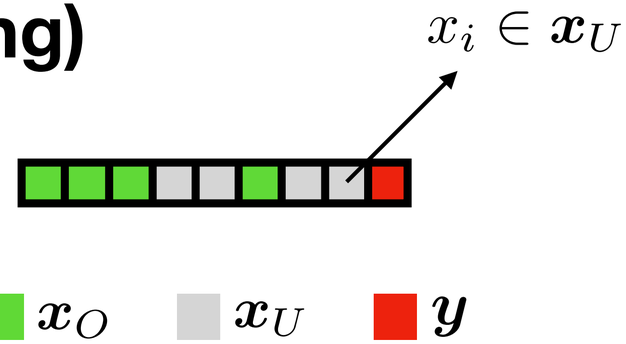
- Approximated in [29,30] by transforming the reward into the latent space:

$$\begin{aligned}\hat{R}\left(i, \boldsymbol{x}_o\right) = & \mathbb{E}_{\hat{p}\left(\boldsymbol{x}_i | \boldsymbol{x}_o\right)} D_{KL}\left[q\left(\boldsymbol{z} | \boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z} | \boldsymbol{x}_o\right)\right] - \\ & \mathbb{E}_{\hat{p}\left(\boldsymbol{y}, \boldsymbol{x}_i | \boldsymbol{x}_o\right)} D_{KL}\left[q\left(\boldsymbol{z} | \boldsymbol{y}, \boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z} | \boldsymbol{y}, \boldsymbol{x}_o\right)\right]\end{aligned}$$

[33] (Bernardo et at., 1979) [29] (Ma et at., 2018) [30] (Ma et at., 2020)

Problem Statement

Discovery of high-value information (Active Learning)



- Bayesian reward function [33] as an expected gain of information:

$$R\left(i, \boldsymbol{x}_O\right) = \mathbb{E}_{\boldsymbol{x}_i \sim p\left(\boldsymbol{x}_i | \boldsymbol{x}_O\right)} D_{\mathrm{KL}}\left[p\left(\boldsymbol{y} | \boldsymbol{x}_i, \boldsymbol{x}_O\right) || \, p\left(\boldsymbol{y} | \boldsymbol{x}_O\right)\right]$$

- Approximated in [29,30] by transforming the reward into the latent space:

$$\begin{aligned}\hat{R}\left(i, \boldsymbol{x}_o\right) = & \mathbb{E}_{\hat{p}\left(\boldsymbol{x}_i | \boldsymbol{x}_o\right)} D_{KL}\left[q\left(\boldsymbol{z} | \boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z} | \boldsymbol{x}_o\right)\right] - \\ & \mathbb{E}_{\hat{p}\left(\boldsymbol{y}, \boldsymbol{x}_i | \boldsymbol{x}_o\right)} D_{KL}\left[q\left(\boldsymbol{z} | \boldsymbol{y}, \boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z} | \boldsymbol{y}, \boldsymbol{x}_o\right)\right]\end{aligned}$$

- These methods are based on **Gaussian** approximations of the true posterior.

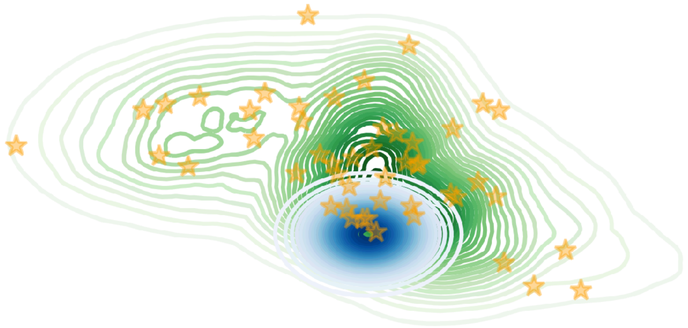
[33] (Bernardo et at., 1979) [29] (Ma et at., 2018) [30] (Ma et at., 2020)

Problem Statement

Sampling-based methods for incomplete data related tasks

Expectations over the intractable posterior should leverage a well-designed MCMC approximation method when compared to a Gaussian-based approximation.

$q(\boldsymbol{z}|\boldsymbol{x})$ $p(\boldsymbol{z}|\boldsymbol{x})$



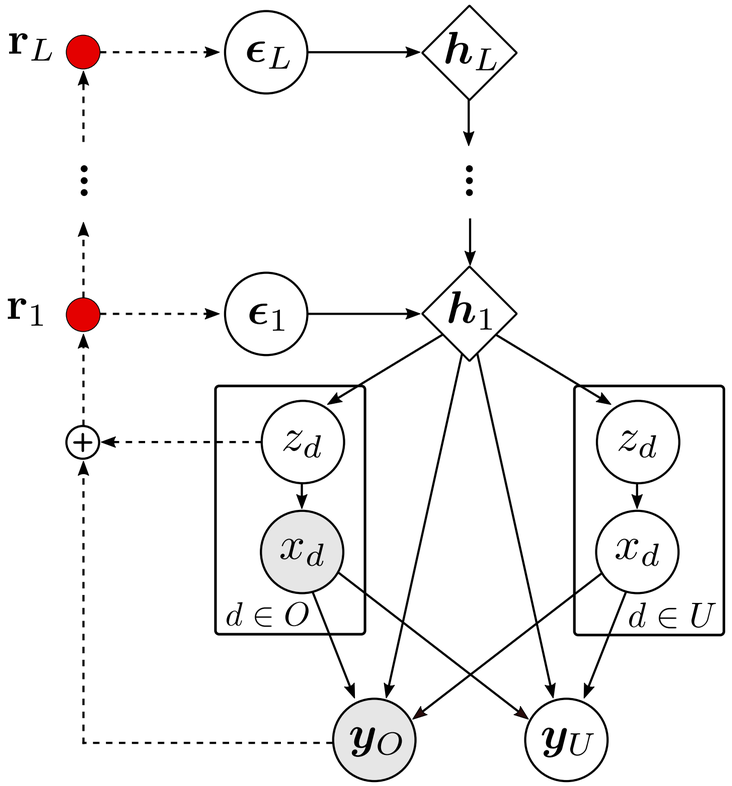
HMC samples (orange)

- **Imputation**: $p(\boldsymbol{x}_U|\boldsymbol{x}_O) \approx \mathbb{E}_{q^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O)}[p(\boldsymbol{x}_U|\boldsymbol{\epsilon})]$.
- **Prediction**: $p(\boldsymbol{y}|\boldsymbol{x}_O) \approx \mathbb{E}_{q^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O)}[p(\boldsymbol{y}|\boldsymbol{\epsilon}, \boldsymbol{x}_O, \hat{\boldsymbol{x}}_U)]$.
- **Sampling-based active learning**.

Contributions

Hierarchical Hamiltonian VAE for mixed-type incomplete data (HH-VAEM)

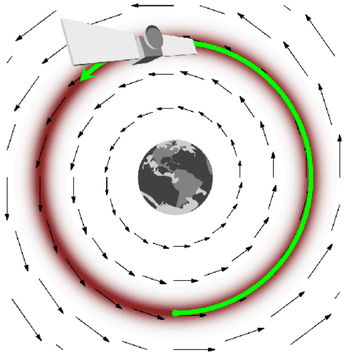
- Increased flexibility by using hierarchical latent space.
- Heterogeneous data handling [30].
- Improved inference by means of automatically tuned HMC.
- Reparameterization for well-posed HMC on relaxed posterior.



[30] (Ma et at., 2020)

Method

Hamiltonian Monte Carlo [35,36]



- Sample from complex distributions via unnormalised targets

1. Phase and momentum space

$$p(\boldsymbol{z}, \boldsymbol{r}) = p(\boldsymbol{r}|\boldsymbol{z})p(\boldsymbol{z})$$

$$H(\boldsymbol{z}, \boldsymbol{r}) = -\log p(\boldsymbol{z}, \boldsymbol{r}) = -\log p(\boldsymbol{r}|\boldsymbol{z}) - \log p(\boldsymbol{z}) = K(\boldsymbol{r}, \boldsymbol{z}) + V(\boldsymbol{z})$$

$$H(\boldsymbol{z}, \boldsymbol{r}) = -\log p^*(\boldsymbol{z}) + \frac{1}{2}\boldsymbol{r}^T \boldsymbol{M}^{-1}\boldsymbol{r}.$$

2. **Hamiltonian** equations

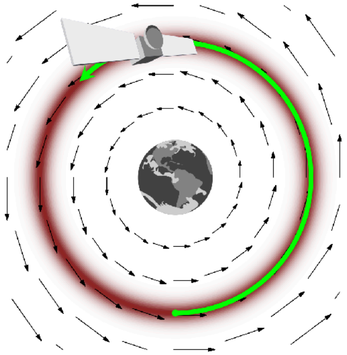
$$\frac{d\boldsymbol{z}}{dt} = +\frac{\partial H}{\partial \boldsymbol{r}} = \frac{\partial K}{\partial \boldsymbol{r}}$$
$$\frac{d\boldsymbol{r}}{dt} = -\frac{\partial H}{\partial \boldsymbol{z}} = -\frac{\partial K}{\partial \boldsymbol{z}} - \frac{\partial V}{\partial \boldsymbol{z}}$$

$$\boldsymbol{r}_{l+\frac{1}{2}} = \boldsymbol{r}_l + \frac{1}{2}\boldsymbol{\phi} \odot \nabla_{z_l} \log p^*(\boldsymbol{z}_l)\,,$$
$$\boldsymbol{z}_{l+1} = \boldsymbol{z}_k + \boldsymbol{r}_{l+\frac{1}{2}} \odot \boldsymbol{\phi} \odot \frac{1}{\boldsymbol{M}}\,,$$
$$\boldsymbol{r}_{l+1} = \boldsymbol{r}_{l+\frac{1}{2}} + \frac{1}{2}\boldsymbol{\phi} \odot \nabla_{z_{l+1}} \log p^*(\boldsymbol{z}_{l+1}),$$

[35] (Neal et at., 2011) [36] (Betancourt et at., 2017)

Method

Hamiltonian Monte Carlo [35,36]



- Sample from complex distributions via unnormalised targets

1. Phase and momentum space

$$p(\boldsymbol{z}, \boldsymbol{r}) = p(\boldsymbol{r}|\boldsymbol{z})p(\boldsymbol{z})$$

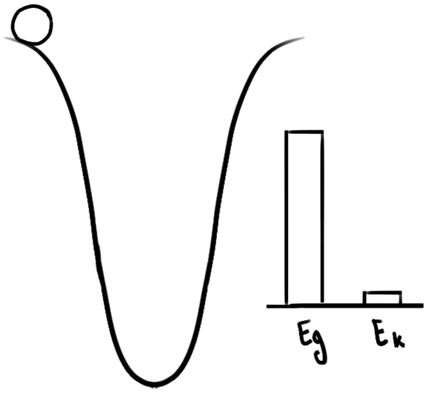
$$H(\boldsymbol{z}, \boldsymbol{r}) = -\log p(\boldsymbol{z}, \boldsymbol{r}) = -\log p(\boldsymbol{r}|\boldsymbol{z}) - \log p(\boldsymbol{z}) = K(\boldsymbol{r}, \boldsymbol{z}) + V(\boldsymbol{z})$$

$$H(\boldsymbol{z}, \boldsymbol{r}) = -\log p^*(\boldsymbol{z}) + \frac{1}{2}\boldsymbol{r}^T \boldsymbol{M}^{-1}\boldsymbol{r}.$$

2. **Hamiltonian** equations

$$\frac{d\boldsymbol{z}}{dt} = +\frac{\partial H}{\partial \boldsymbol{r}} = \frac{\partial K}{\partial \boldsymbol{r}}$$
$$\frac{d\boldsymbol{r}}{dt} = -\frac{\partial H}{\partial \boldsymbol{z}} = -\frac{\partial K}{\partial \boldsymbol{z}} - \frac{\partial V}{\partial \boldsymbol{z}}$$

$$\boldsymbol{r}_{l+\frac{1}{2}} = \boldsymbol{r}_l + \frac{1}{2}\boldsymbol{\phi} \odot \nabla_{z_l} \log p^*(\boldsymbol{z}_l)\,,$$
$$\boldsymbol{z}_{l+1} = \boldsymbol{z}_k + \boldsymbol{r}_{l+\frac{1}{2}} \odot \boldsymbol{\phi} \odot \frac{1}{\boldsymbol{M}}\,,$$
$$\boldsymbol{r}_{l+1} = \boldsymbol{r}_{l+\frac{1}{2}} + \frac{1}{2}\boldsymbol{\phi} \odot \nabla_{z_{l+1}} \log p^*(\boldsymbol{z}_{l+1}),$$



[35] (Neal et at., 2011) [36] (Betancourt et at., 2017)

Method

Hamiltonian Monte Carlo

- Discrete trajectories (*chains*) of T updates, ending in

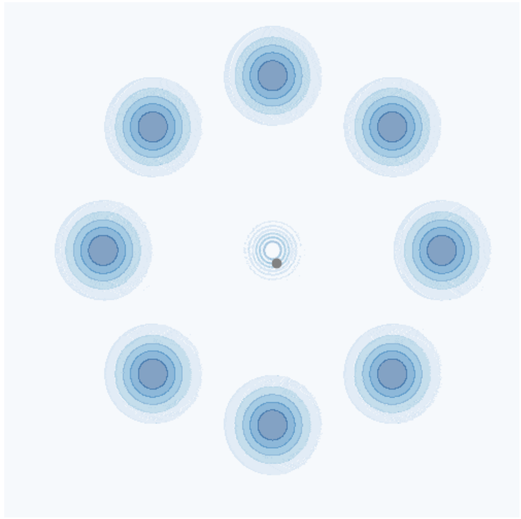
  ★ $\boldsymbol{z}^{(T)} \sim q^{(T)}(\boldsymbol{z} \mid \boldsymbol{x}) \approx p(\boldsymbol{z} \mid \boldsymbol{x})$

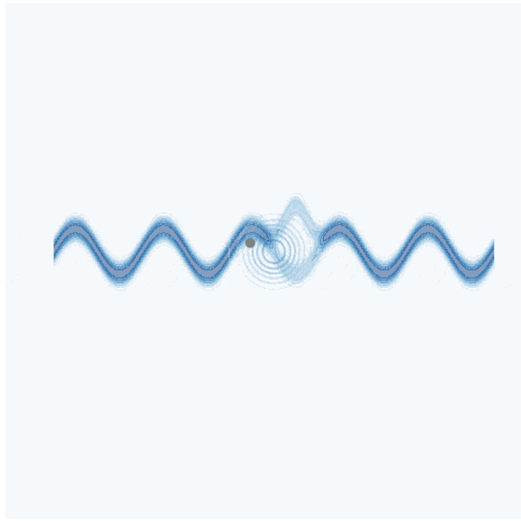
- **Target**: true posterior density (blue).

- Needed:

  1. Good initial **proposal** (encoder).

  2. Well-defined **hyperparameters**.





Method

Hamiltonian Monte Carlo

- Discrete trajectories (*chains*) of T updates, ending in

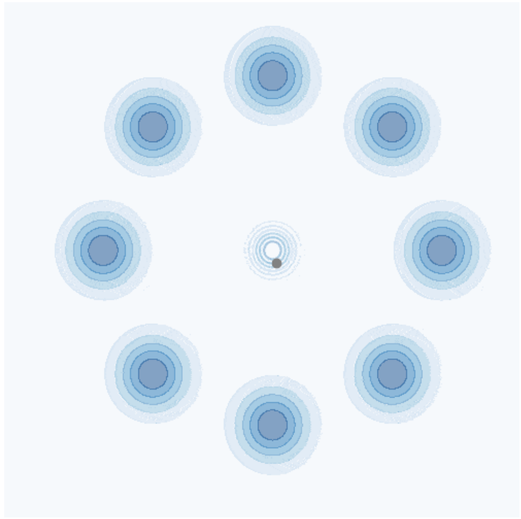
  ★ $\boldsymbol{z}^{(T)} \sim q^{(T)}(\boldsymbol{z} \mid \boldsymbol{x}) \approx p(\boldsymbol{z} \mid \boldsymbol{x})$

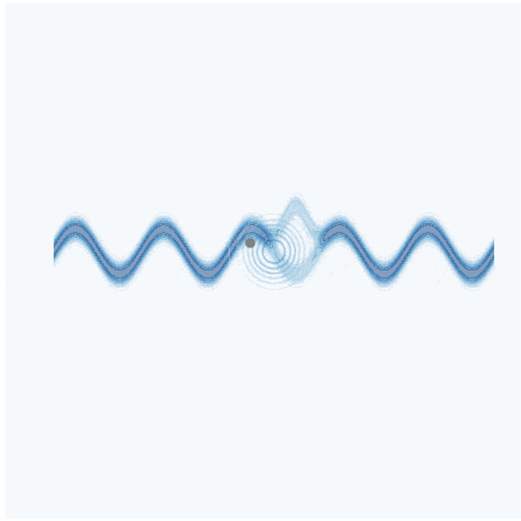
- **Target**: true posterior density (blue).

- Needed:

  1. Good initial **proposal** (encoder).

  2. Well-defined **hyperparameters**.





Method

Hamiltonian Monte Carlo

- Discrete trajectories (*chains*) of T updates, ending in

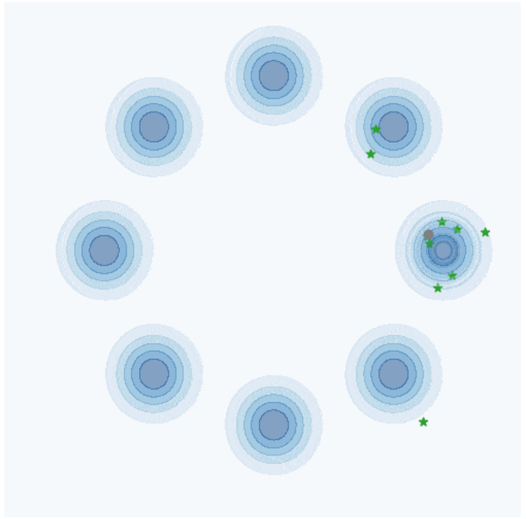
  ★ $\boldsymbol{z}^{(T)} \sim q^{(T)}(\boldsymbol{z} \mid \boldsymbol{x}) \approx p(\boldsymbol{z} \mid \boldsymbol{x})$

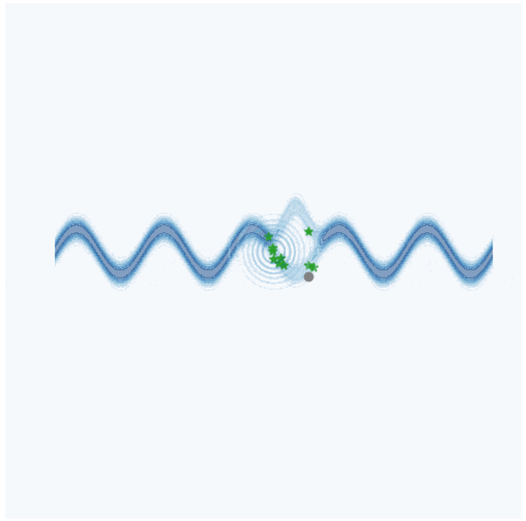
- **Target**: true posterior density (blue).

- Needed:

  1. Good initial **proposal** (encoder).

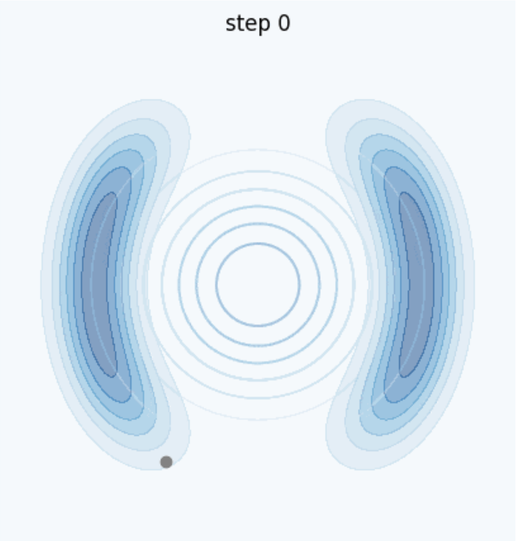
  2. Well-defined **hyperparameters**.

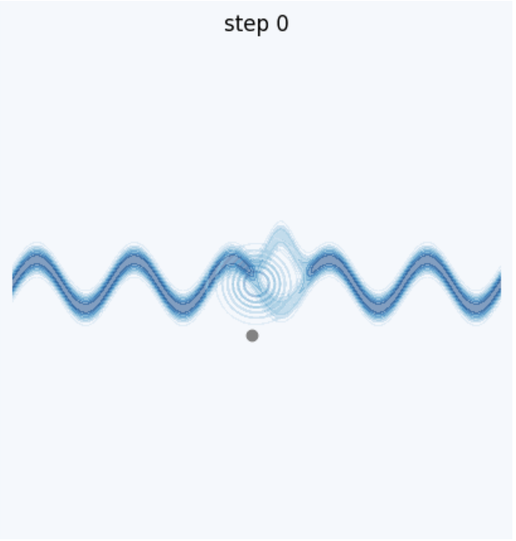




Method

HMC Hyperparameter tuning [9]





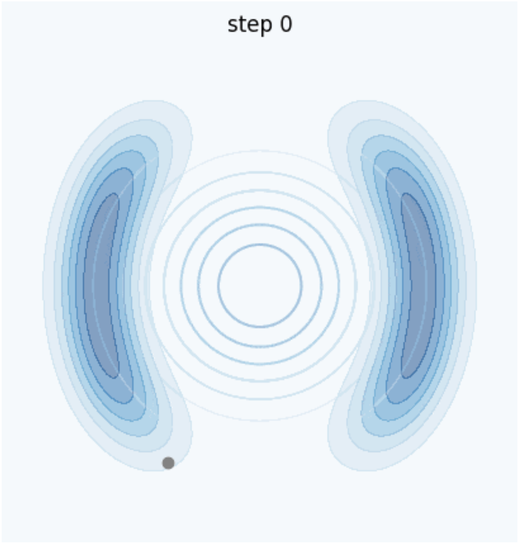
[9] (Campbell et at., 2021) [37] (Gong et at., 2020)

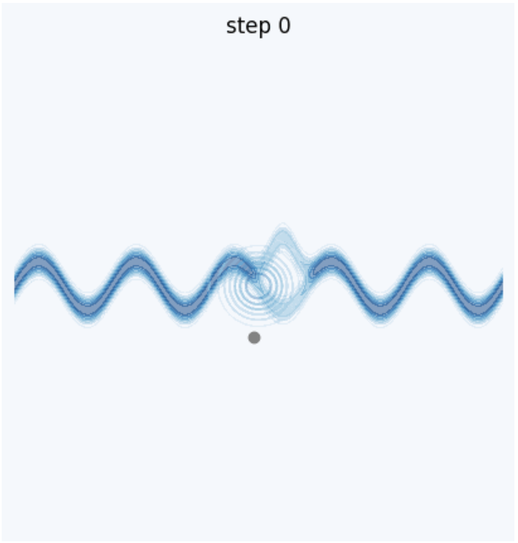
Method

HMC Hyperparameter tuning [9]

- Tuning the **hyperparams** via Variational Inference:

$$\phi^* = \underset{\phi}{\text{argmax}} \; \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{z})} \left[\log p^*(\boldsymbol{z})\right] + H\left[q_\phi^{(T)}(\boldsymbol{z})\right]$$





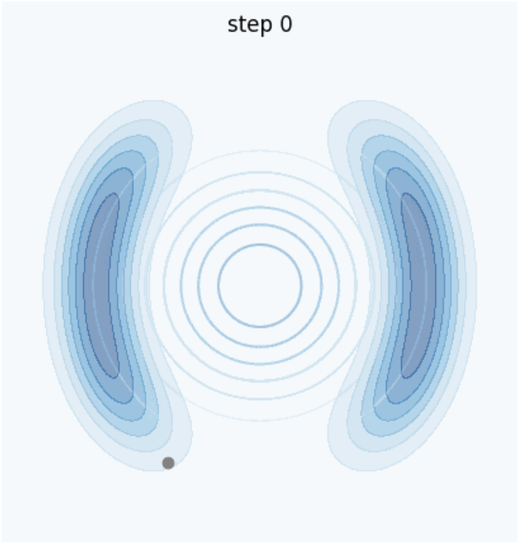
[9] (Campbell et at., 2021) [37] (Gong et at., 2020)

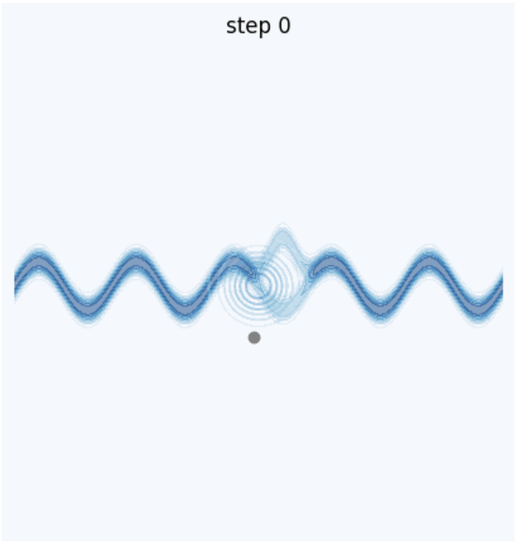
Method

HMC Hyperparameter tuning [9]

- Tuning the **hyperparams** via Variational Inference:

$$\phi^* = \underset{\phi}{\text{argmax}} \; \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{z})} \left[\log p^*(\boldsymbol{z})\right] + H\left[q_\phi^{(T)}(\boldsymbol{z})\right]$$





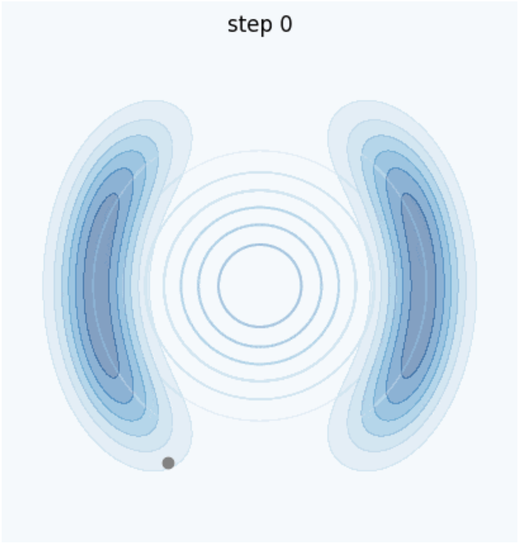
[9] (Campbell et at., 2021) [37] (Gong et at., 2020)

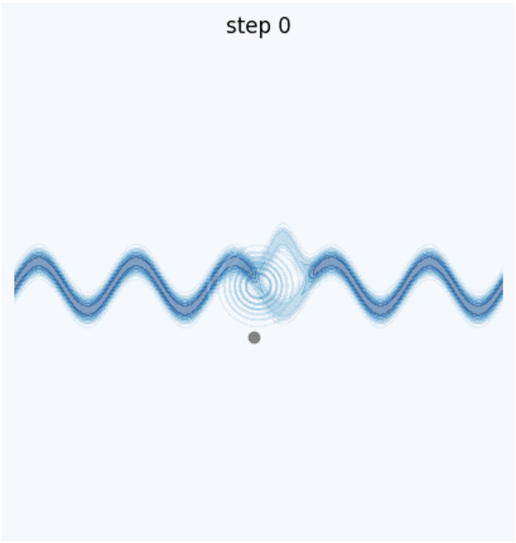
Method

HMC Hyperparameter tuning [9]

- Tuning the **hyperparams** via Variational Inference:

$$\phi^* = \underset{\phi}{\text{argmax}} \; \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{z})} \left[\log p^*(\boldsymbol{z}) \right]$$





[9] (Campbell et at., 2021) [37] (Gong et at., 2020)

Method

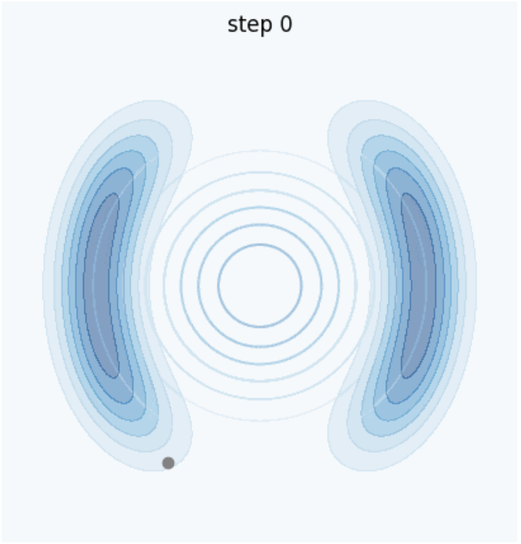
HMC Hyperparameter tuning [9]

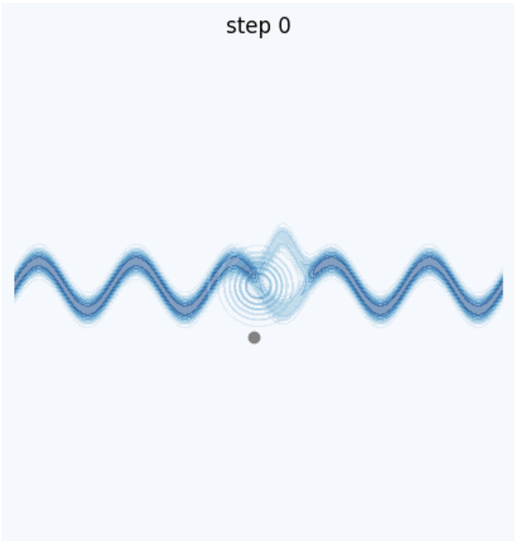
- Tuning the **hyperparams** via Variational Inference:

$$\phi^* = \underset{\phi}{\text{argmax}} \;\; \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{z})} \left[\log p^*(\boldsymbol{z})\right]$$

- Add an **inflation** parameter for scaling the proposal [37]

$$\boldsymbol{s}^* = \underset{s}{\text{argmin}} \;\; \text{SKSD}(\boldsymbol{z}^{(T)}, \nabla_z \log p^*(\boldsymbol{z}))$$





[9] (Campbell et at., 2021) [37] (Gong et at., 2020)

Method

HMC Hyperparameter tuning [9]

- Tuning the **hyperparams** via Variational Inference:

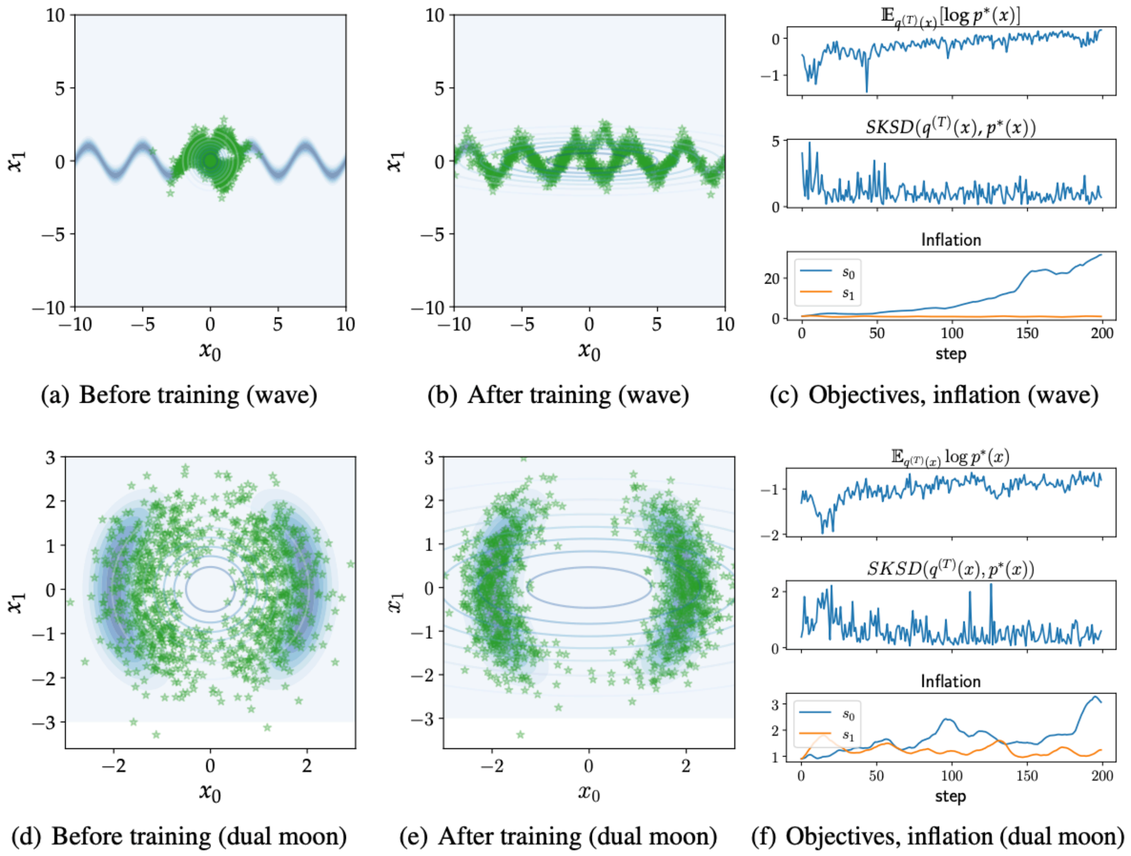
$$\phi^* = \underset{\phi}{\text{argmax}} \;\; \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{z})} \left[\log p^*(\boldsymbol{z})\right]$$

- Add an **inflation** parameter for scaling the proposal [37]

$$\boldsymbol{s}^* = \underset{s}{\text{argmin}} \;\; \text{SKSD}(\boldsymbol{z}^{(T)}, \nabla_z \log p^*(\boldsymbol{z}))$$

- Code available at:

https://github.com/ipeis/HMCTuning



(a) Before training (wave) (b) After training (wave) (c) Objectives, inflation (wave)

(d) Before training (dual moon) (e) After training (dual moon) (f) Objectives, inflation (dual moon)

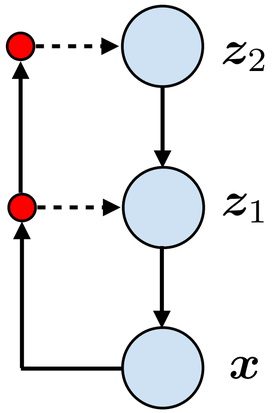
[9] (Campbell et at., 2021) [37] (Gong et at., 2020)

Method

Hierarchical VAEs

- Hierarchical VAEs are successful at increasing flexibility.

$$p(\boldsymbol{z}) = p\left(\boldsymbol{z}_L\right) \prod_{i=1}^{L-1} p\left(\boldsymbol{z}_i \mid \boldsymbol{z}_{i+1}\right)$$



- The hierarchy allows for modelling:
  - Abstract to specific generative factors.
  - Global to local generative factors.

[13]

[13] (Child, 2020)

Method

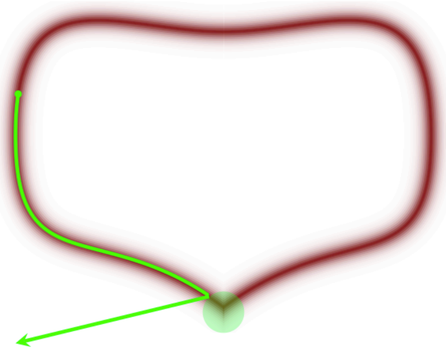
HMC is ill-posed for Hierarchical VAEs

- Hierarchical dependencies lead to **huge gradients** [36,38]

$$\boldsymbol{r}_{l+\frac{1}{2}} = \boldsymbol{r}_l + \frac{1}{2}\boldsymbol{\phi} \odot \nabla_{z_l} \log p^*(\boldsymbol{z}_l)\,,$$

$$\boldsymbol{z}_{l+1} = \boldsymbol{z}_k + \boldsymbol{r}_{l+\frac{1}{2}} \odot \boldsymbol{\phi} \odot \frac{1}{\boldsymbol{M}}\,,$$

$$\boldsymbol{r}_{l+1} = \boldsymbol{r}_{l+\frac{1}{2}} + \frac{1}{2}\boldsymbol{\phi} \odot \nabla_{z_{l+1}} \log p^*(\boldsymbol{z}_{l+1}),$$



- Samples can diverge due to integrator overflow issues.

[36] (Betancourt et at., 2017) [38] (Betancourt and Girolami, 2015)

Method

HMC is ill-posed for Hierarchical VAEs

Method

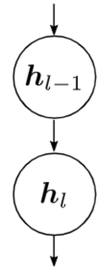
HMC is ill-posed for Hierarchical VAEs

- **Solution**:

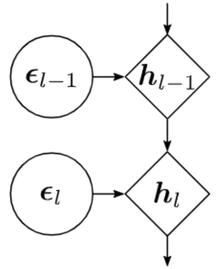
  ✓ Reparameterization for relaxed posterior:

$$\boldsymbol{h}_l = f_{\mu_l}(\boldsymbol{h}_{l+1}) + f_{\sigma_l}(\boldsymbol{h}_{l+1}) \cdot \boldsymbol{\epsilon}_l$$

  NNs with parameters $\theta_{\mu_l} \rightarrow f_{\mu_l}$, $\theta_{\sigma_l} \rightarrow f_{\sigma_l}$



(a) AR hierarchy



(b) Reparameterization

Method

HMC is ill-posed for Hierarchical VAEs

- **Solution**:

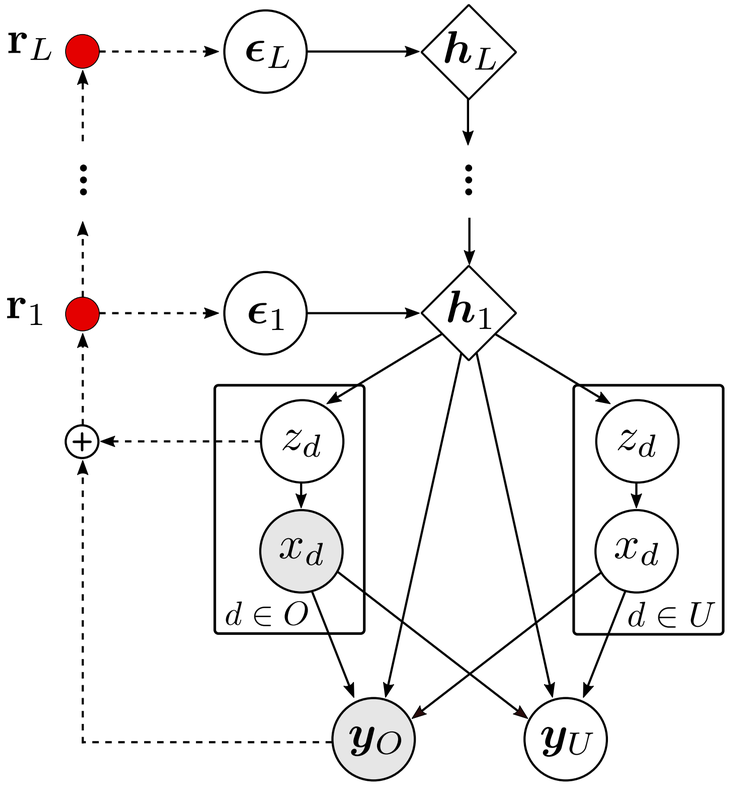
  ✓ Reparameterization for relaxed posterior:

$$\boldsymbol{h}_l = f_{\mu_l}(\boldsymbol{h}_{l+1}) + f_{\sigma_l}(\boldsymbol{h}_{l+1}) \cdot \boldsymbol{\epsilon}_l$$

  NNs with parameters $\theta_{\mu_l} \rightarrow f_{\mu_l}$, $\theta_{\sigma_l} \rightarrow f_{\sigma_l}$

  ✓ Perform inference on $\epsilon = \{\epsilon_1, \ldots, \epsilon_1\}$ with standard Gaussian prior.

  ✓ No need to increase complexity of the HMC method.



Method

HMC is ill-posed for Hierarchical VAEs

- **Solution**:

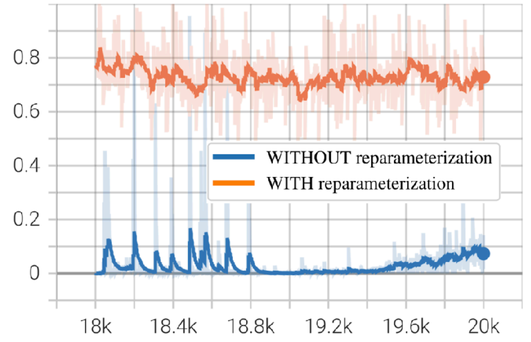
  ✓ Reparameterization for relaxed posterior:

$$\boldsymbol{h}_l = f_{\mu_l}(\boldsymbol{h}_{l+1}) + f_{\sigma_l}(\boldsymbol{h}_{l+1}) \cdot \boldsymbol{\epsilon}_l$$

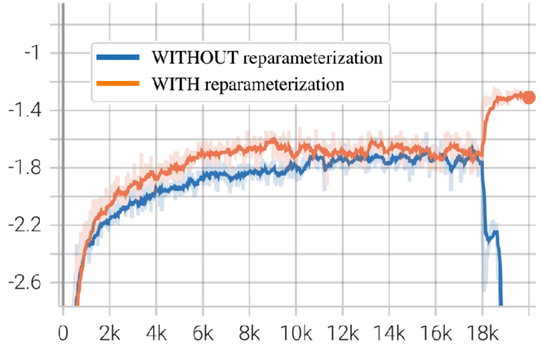
  NNs with parameters $\theta_{\mu_l} \rightarrow f_{\mu_l}$, $\theta_{\sigma_l} \rightarrow f_{\sigma_l}$

  ✓ Perform inference on $\epsilon = \{\epsilon_1, \ldots, \epsilon_1\}$ with standard Gaussian prior.

  ✓ No need to increase complexity of the HMC method.



(a) Mean acceptance rate $\bar{p}_a$



(b) $\log p(\boldsymbol{x}_U|\boldsymbol{x}_O)$

Method

Optimization algorithm

Algorithm 1 Training algorithm for HH-VAEM

Input: data $\left(\boldsymbol{x}_O^{(1:N)}, \boldsymbol{y}_O^{(1:N)}\right)$, steps: T_d, T_{VI}, T_{HMC}
Parameters: $\gamma, \theta, \psi, \phi, s$
STAGE 1: MARGINAL VAEs
for $d = 1$ **to** D **do**
  Initialize marginal VAE $\{\theta_d, \gamma_d\}$
  for $t = 1$ **to** T_d **do**
    $\gamma_d^{t+1}, \theta_d^{t+1} \leftarrow \text{Adam}_{\gamma_d^t, \theta_d^t}(\mathcal{L}_d)$
  end for
end for

1. Train marginal VAEs using:

$$\mathcal{L}_d\left(x_d; \{\theta_d, \gamma_d\}\right) = \mathbb{I}\left(x_d \in \boldsymbol{x}_O\right) \mathbb{E}_{q_{\gamma_d}(z_d|x_d)} \log \frac{p_{\theta_d}\left(x_d, z_d\right)}{q_{\gamma_d}\left(z_d \mid x_d\right)}$$

Method

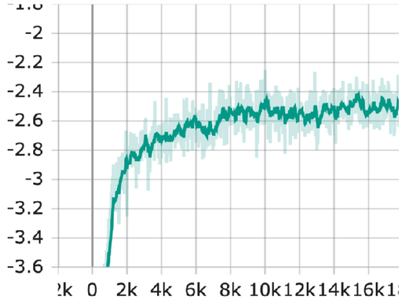
Optimization algorithm

Algorithm 1 Training algorithm for HH-VAEM

Input: data $\left(\boldsymbol{x}_O^{(1:N)}, \boldsymbol{y}_O^{(1:N)}\right)$, steps: T_d, T_{VI}, T_{HMC}
Parameters: $\gamma, \theta, \psi, \phi, s$
STAGE 1: MARGINAL VAEs
for $d = 1$ **to** D **do**
  Initialize marginal VAE $\{\theta_d, \gamma_d\}$
  for $t = 1$ **to** T_d **do**
    $\gamma_d^{t+1}, \theta_d^{t+1} \leftarrow \text{Adam}_{\gamma_d^t, \theta_d^t}(\mathcal{L}_d)$
  end for
end for
STAGE 2: DEPENDENCY VAE
for $t = 1$ **to** T_{VAE} **do**
  $\theta^{t+1}, \psi^{t+1} \leftarrow \text{Adam}_{\theta^t, \psi^t}(\mathcal{L}_{VI})$
end for

2. Training Hierarchical VAE using the ELBO:

$$\mathcal{L}_{VI}\left(\boldsymbol{x}_O, \boldsymbol{y}_O; \{\theta, \psi\}\right) = \mathbb{E}_{q_\psi}\left[\log p_\theta\left(\boldsymbol{z}_O \mid \boldsymbol{h}_1\right) + \log p_\theta\left(\boldsymbol{y}_O \mid \hat{\boldsymbol{x}}, \boldsymbol{h}_1\right)\right] - \sum_{l=1}^{L} D_{\text{KL}}\left(q_\psi\left(\boldsymbol{\epsilon}_l \mid \boldsymbol{x}_O, \boldsymbol{y}_O\right) \| p\left(\boldsymbol{\epsilon}_l\right)\right)$$



(a) $\log p(\boldsymbol{x}_U | \boldsymbol{x}_O)$

Method

Optimization algorithm

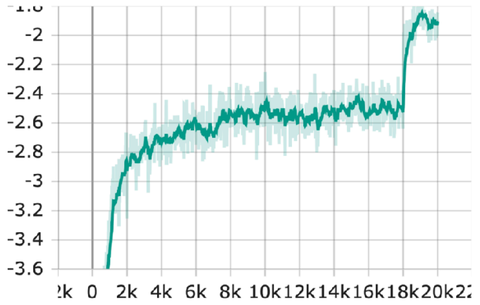
Algorithm 1 Training algorithm for HH-VAEM

Input: data $\left(\boldsymbol{x}_O^{(1:N)}, \boldsymbol{y}_O^{(1:N)}\right)$, steps: T_d, T_{VI}, T_{HMC}
Parameters: $\gamma, \theta, \psi, \phi, s$
STAGE 1: MARGINAL VAEs
for $d = 1$ **to** D **do**
  Initialize marginal VAE $\{\theta_d, \gamma_d\}$
  for $t = 1$ **to** T_d **do**
    $\gamma_d^{t+1}, \theta_d^{t+1} \leftarrow \text{Adam}_{\gamma_d^t, \theta_d^t}(\mathcal{L}_d)$
  end for
end for
STAGE 2: DEPENDENCY VAE
for $t = 1$ **to** T_{VAE} **do**
  $\theta^{t+1}, \psi^{t+1} \leftarrow \text{Adam}_{\theta^t, \psi^t}(\mathcal{L}_{VI})$
end for
STAGE 3: JOINTLY OPTIMIZING VAE + HMC
for $t = 1$ **to** T_{HMC} **do**
  $\psi^{t+1} \leftarrow \text{Adam}_{\psi^t}(\mathcal{L}_{VI})$
  $\theta^{t+1}, \phi^{t+1} \leftarrow \text{Adam}_{\theta^t, \phi^t}(\mathcal{L}_{HMC})$
  $s^{t+1} \leftarrow \text{Adam}_{s^t}(\mathcal{L}_{SKSD})$
end for

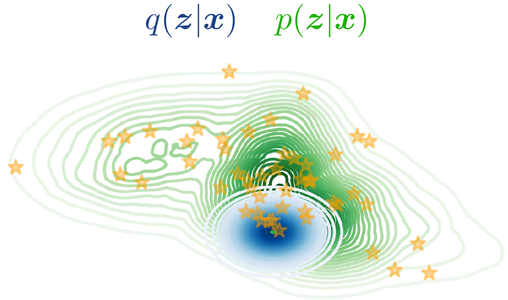
3. Train a) encoder using ELBO, b) HMC hyperparams, decoder and predictor parameters using HMC objective and c) scale using SKSD.

$$\mathcal{L}_{HMC}\left(\boldsymbol{z}_O, \boldsymbol{y}_O; \{\theta, \psi, \phi\}\right) = \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{\epsilon})}[\log p_\theta\left(\boldsymbol{z}_O \mid \boldsymbol{h}_1\right) + \log p_\theta\left(\boldsymbol{y}_O \mid \hat{\boldsymbol{x}}, \boldsymbol{h}_1\right) + \sum_{l=1}^{L} p(\boldsymbol{\epsilon}_l^{(T)})]$$

$$\mathcal{L}_{SKSD}(\boldsymbol{x}_O, \boldsymbol{y}_O; \boldsymbol{s}) = \text{SKSD}\left(q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O; \boldsymbol{s}),\, p(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O)\right)$$



(a) $\log p(\boldsymbol{x}_U|\boldsymbol{x}_O)$



[2]HMC samples (orange)

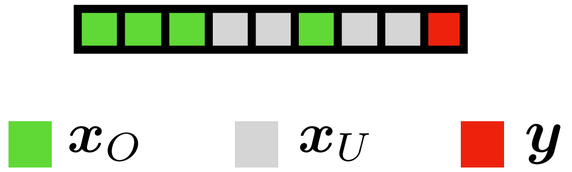
Experiments

Set up

Model	HMC	Hierarchical*
VAEM	✗	✗
MIWAEM	✗	✗
HMC-VAEM	✓	✗
H-VAEM	✗	✓
HH-VAEM	✓	✓

* 2 layers of latent variables.

- **Training**: missing features and target with a probability sampled from $U(0.01, 0.99)$ for each batch.
- **Test**: fixed 50% missing features, fully-missing target.



Experiments

Missing data imputation and target prediction

	Bank	Insurance	Avocado	Naval	Yatch	Diabetes	Concrete	Wine	Energy	Boston
VAEM	2.84 ± 0.07	1.81 ± 0.03	1.89 ± 0.01	0.55 ± 0.05	3.15 ± 0.28	2.78 ± 0.16	2.45 ± 0.26	3.01 ± 0.61	2.09 ± 0.10	2.01 ± 0.23
MIWAEM	2.74 ± 0.05	1.88 ± 0.04	1.92 ± 0.04	0.57 ± 0.03	2.66 ± 0.11	2.55 ± 0.09	2.34 ± 0.51	2.76 ± 0.48	2.06 ± 0.14	1.94 ± 0.23
H-VAEM	2.82 ± 0.06	1.80 ± 0.04	1.89 ± 0.01	0.48 ± 0.06	3.06 ± 0.31	2.74 ± 0.09	2.42 ± 0.21	2.85 ± 0.56	1.72 ± 0.11	1.89 ± 0.24
HMC-VAEM	2.69 ± 0.05	1.77 ± 0.06	1.89 ± 0.02	0.49 ± 0.07	$\mathbf{2.21 \pm 0.24}$	2.72 ± 0.20	2.28 ± 0.29	2.83 ± 0.46	1.73 ± 0.05	1.83 ± 0.16
HH-VAEM	$\mathbf{2.63 \pm 0.04}$	$\mathbf{1.75 \pm 0.03}$	$\mathbf{1.88 \pm 0.05}$	$\mathbf{0.40 \pm 0.05}$	2.47 ± 0.27	$\mathbf{2.54 \pm 0.13}$	$\mathbf{2.28 \pm 0.09}$	$\mathbf{1.90 \pm 0.17}$	$\mathbf{1.71 \pm 0.04}$	$\mathbf{1.83 \pm 0.11}$

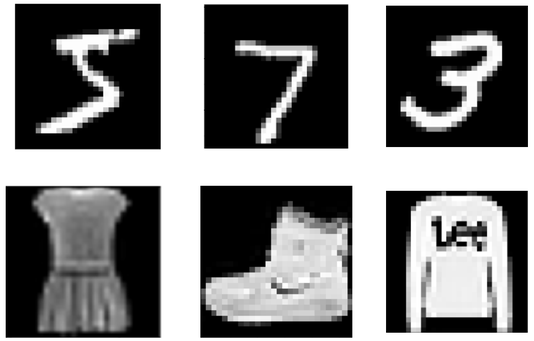
Negative log-likelihood of **imputed missing data**

	Bank	Insurance	Avocado	Naval	Yatch	Diabetes	Concrete	Wine	Energy	Boston
VAEM	0.56 ± 0.06	1.20 ± 0.03	1.18 ± 0.02	2.69 ± 0.01	0.61 ± 0.02	1.59 ± 0.19	1.07 ± 0.09	0.28 ± 0.09	0.61 ± 0.14	0.85 ± 0.21
MIWAEM	0.51 ± 0.03	1.15 ± 0.03	1.15 ± 0.03	2.70 ± 0.01	0.60 ± 0.03	$\mathbf{1.36 \pm 0.10}$	0.95 ± 0.22	0.28 ± 0.13	0.54 ± 0.12	0.80 ± 0.21
H-VAEM	0.50 ± 0.03	1.06 ± 0.02	1.18 ± 0.02	2.68 ± 0.01	0.60 ± 0.02	1.71 ± 0.14	1.02 ± 0.09	0.26 ± 0.11	0.46 ± 0.14	0.90 ± 0.22
HMC-VAEM	0.52 ± 0.02	1.00 ± 0.03	1.12 ± 0.03	2.71 ± 0.01	$\mathbf{0.52 \pm 0.15}$	1.55 ± 0.29	0.95 ± 0.26	0.28 ± 0.09	0.41 ± 0.07	0.71 ± 0.13
HH-VAEM	$\mathbf{0.49 \pm 0.03}$	$\mathbf{0.93 \pm 0.06}$	$\mathbf{1.10 \pm 0.01}$	$\mathbf{2.62 \pm 0.01}$	0.56 ± 0.02	1.38 ± 0.18	$\mathbf{0.95 \pm 0.08}$	$\mathbf{0.20 \pm 0.04}$	$\mathbf{0.32 \pm 0.05}$	$\mathbf{0.55 \pm 0.04}$

Negative log-likelihood of **predicted target**

Experiments

Missing data imputation and target prediction (MNIST datasets)



	VAE	MIWAE	H-VAE	HMC-VAE	**HH-VAE**
MNIST	0.124 ± 0.001	0.121 ± 0.001	0.119 ± 0.001	0.101 ± 0.004	$\mathbf{0.094 \pm 0.003}$
F-MNIST	0.162 ± 0.002	0.160 ± 0.002	0.156 ± 0.002	0.150 ± 0.002	$\mathbf{0.144 \pm 0.002}$

Table 3: Test negative log likelihood of the unobserved features for the MNIST datasets.

	VAE	MIWAE	H-VAE	HMC-VAE	**HH-VAE**
MNIST	0.153 ± 0.009	0.151 ± 0.007	0.146 ± 0.006	0.067 ± 0.007	$\mathbf{0.056 \pm 0.019}$
F-MNIST	0.501 ± 0.012	0.496 ± 0.008	0.494 ± 0.007	0.357 ± 0.060	$\mathbf{0.337 \pm 0.069}$

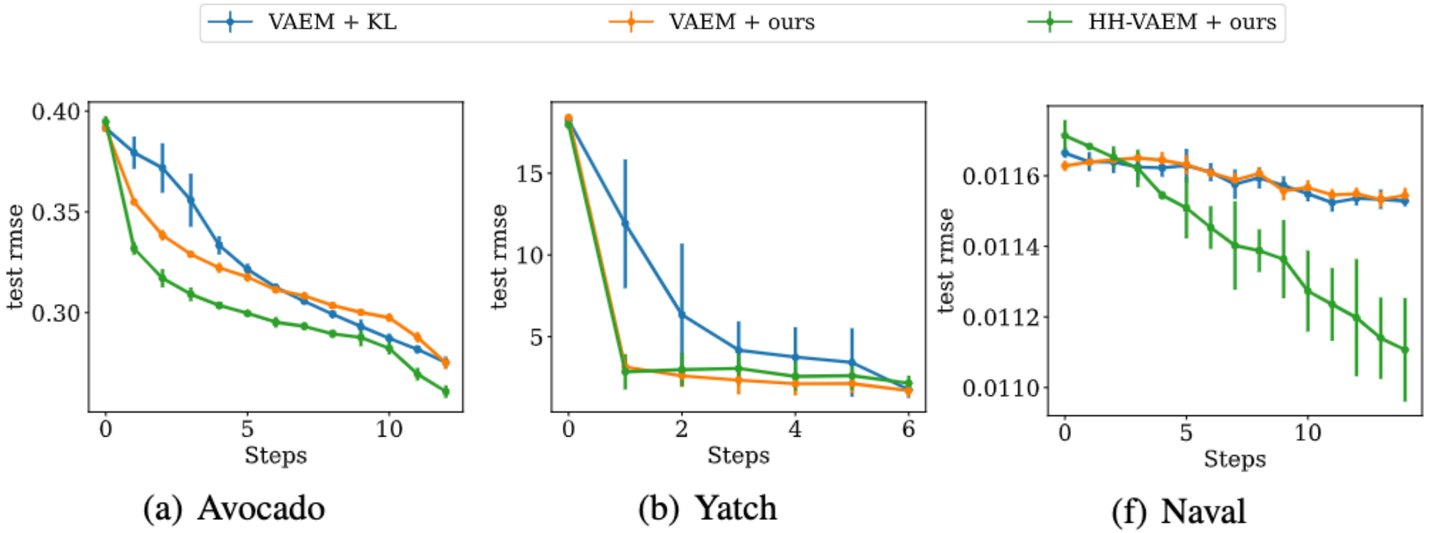
Table 4: Test negative log likelihood of the predicted target for the MNIST datasets.

	VAE	MIWAE	H-VAE	HMC-VAE	**HH-VAE**
MNIST	0.953 ± 0.004	0.953 ± 0.003	0.953 ± 0.003	0.978 ± 0.003	$\mathbf{0.981 \pm 0.005}$
F-MNIST	0.824 ± 0.005	0.824 ± 0.004	0.824 ± 0.004	0.869 ± 0.015	$\mathbf{0.876 \pm 0.017}$

Table 5: Test accuracy of the predicted digits for the MNIST datasets.

Experiments

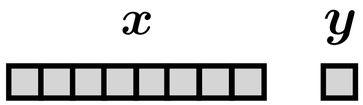
Sequential Active Information Acquisition (SAIA)

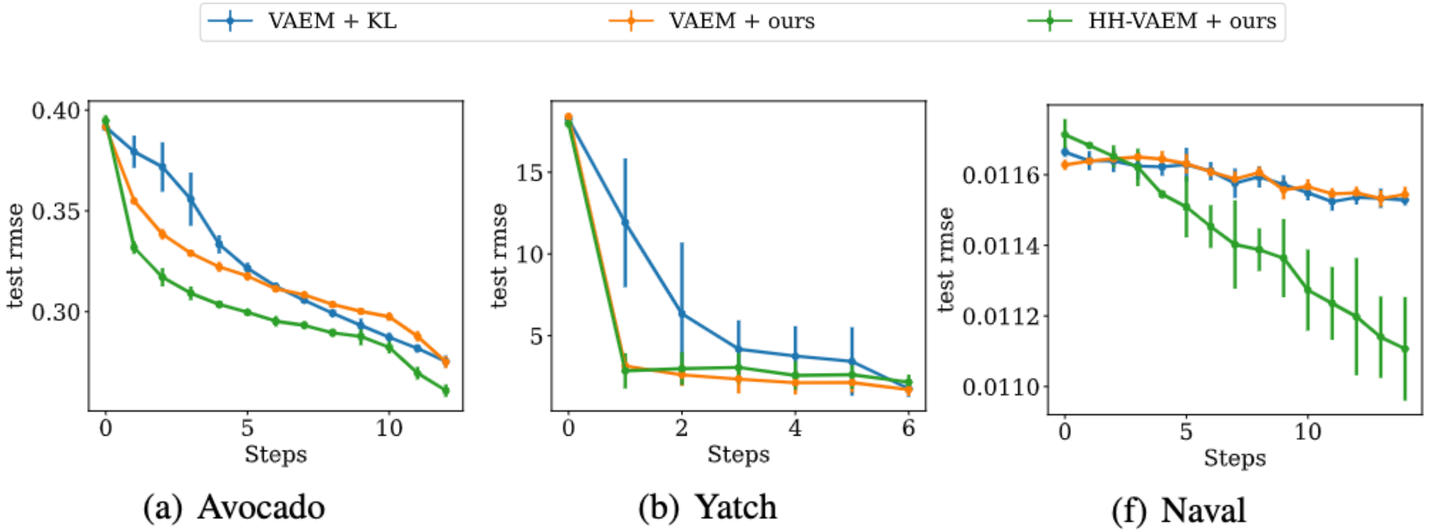


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

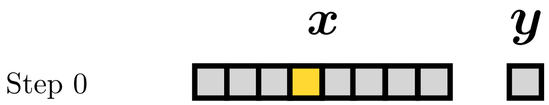


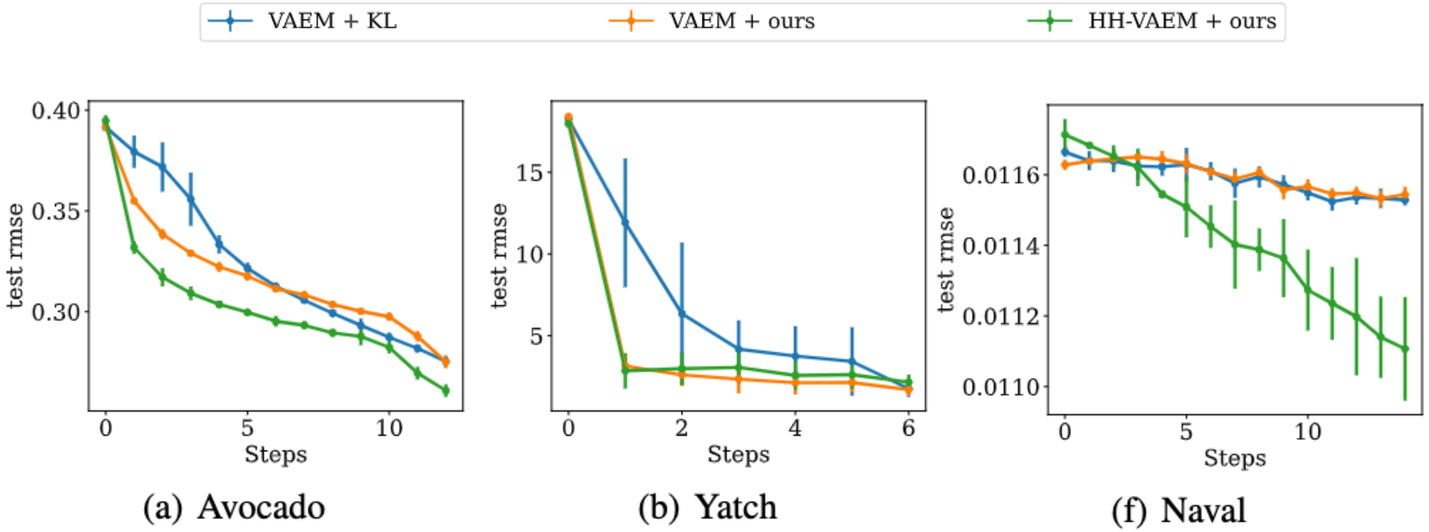


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

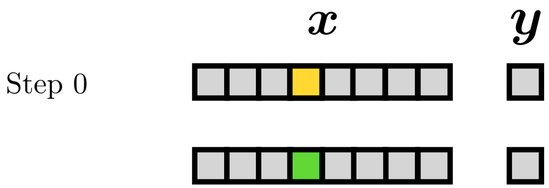


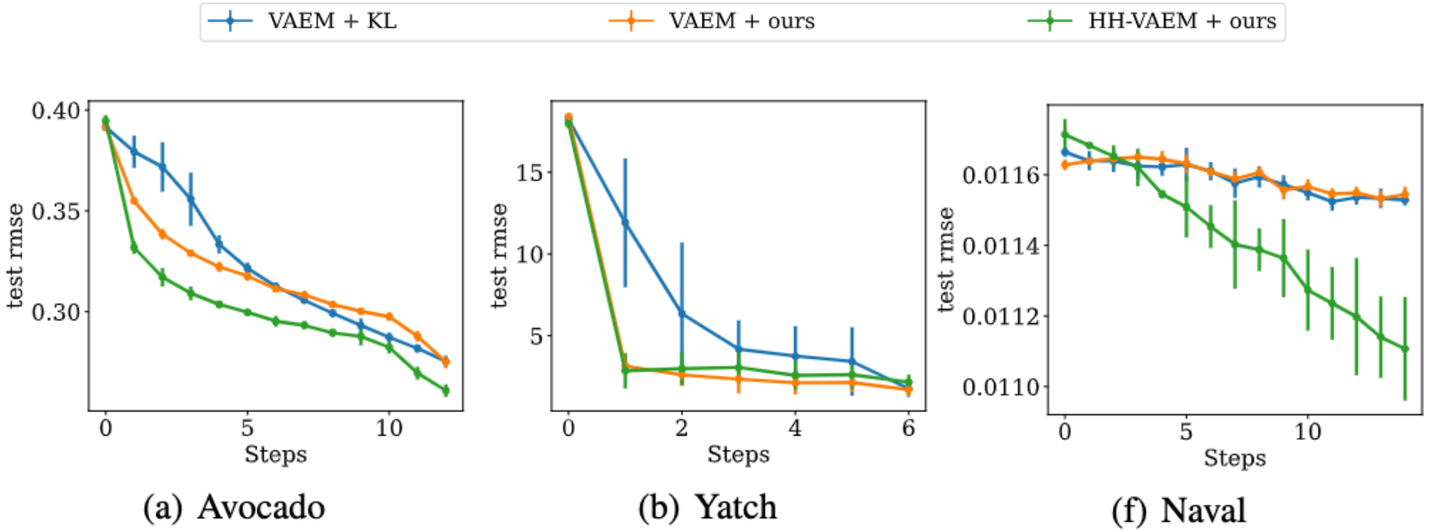


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

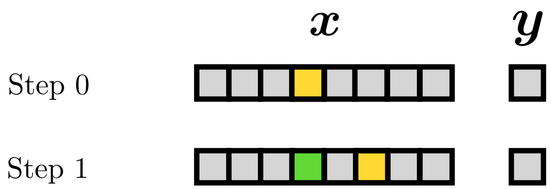


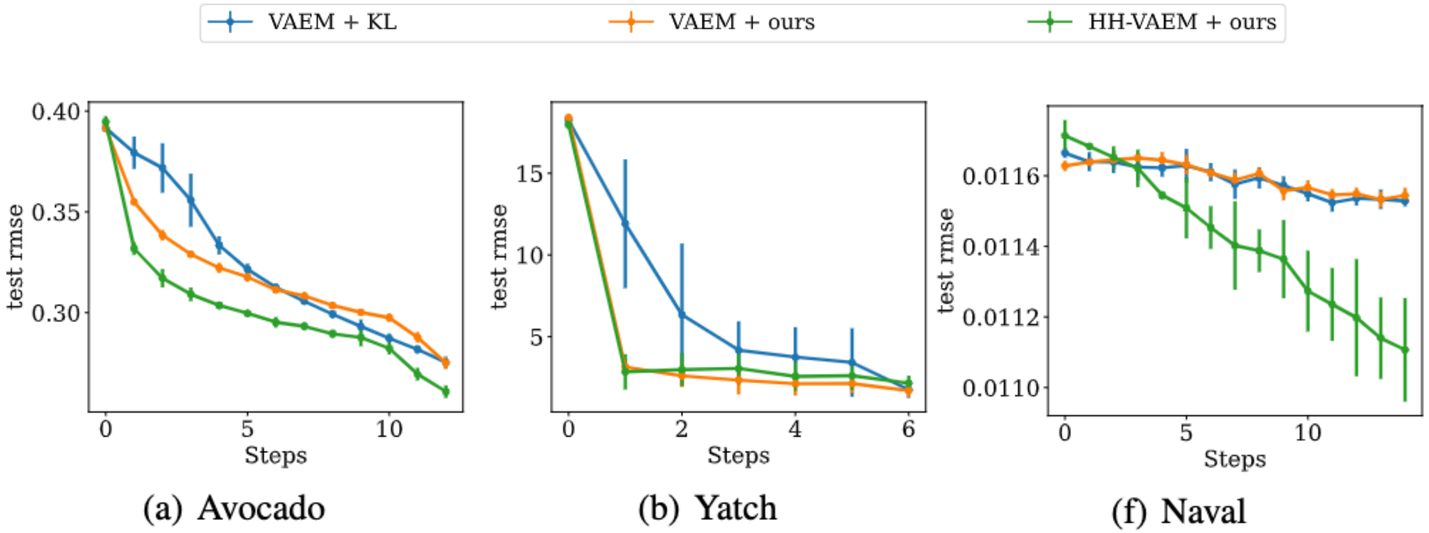


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

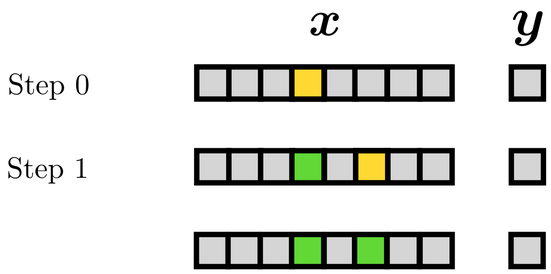


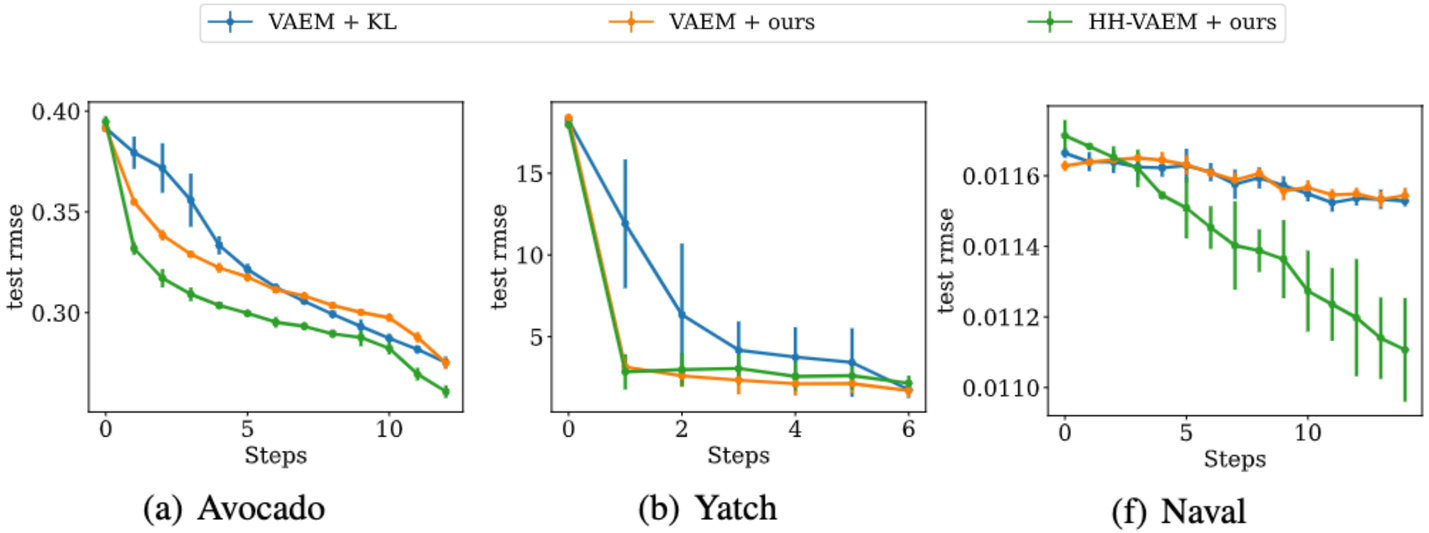


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

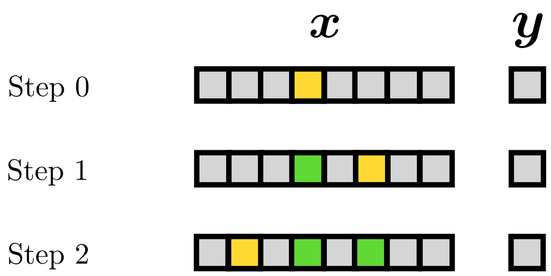


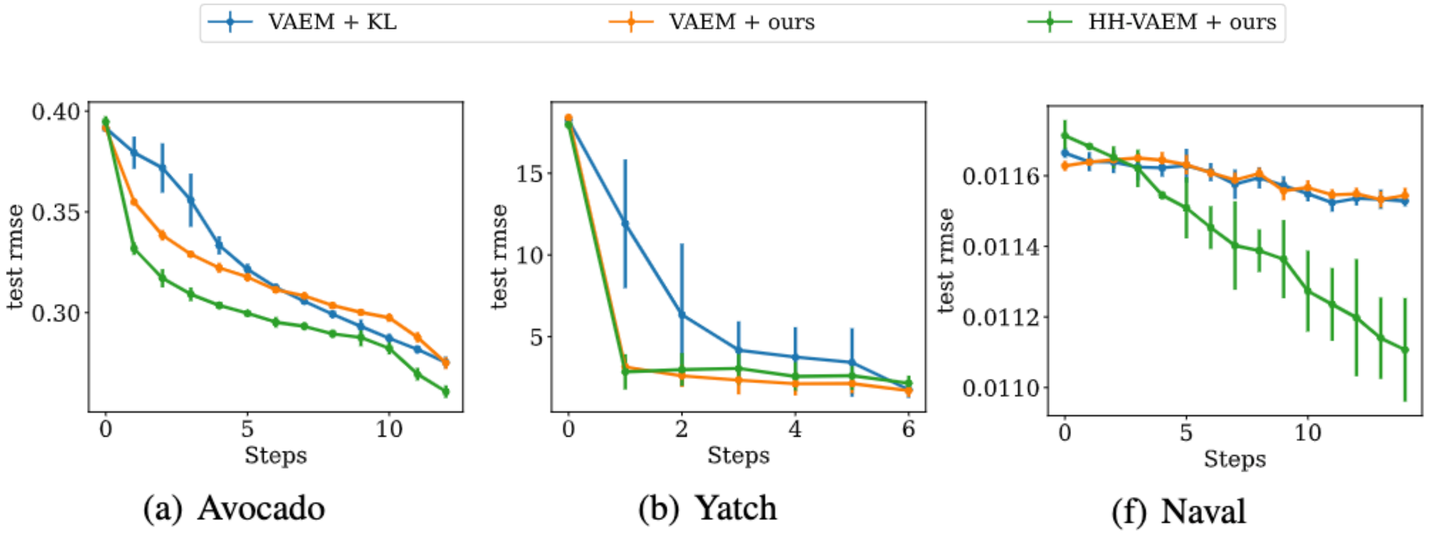


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

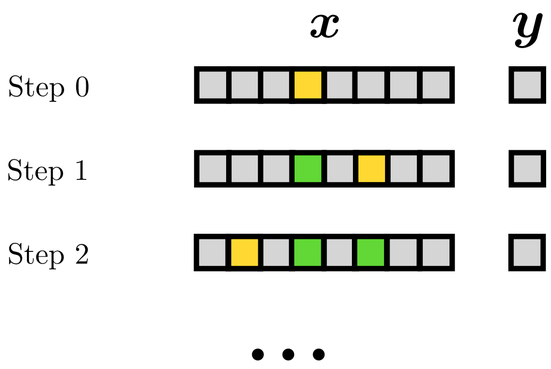


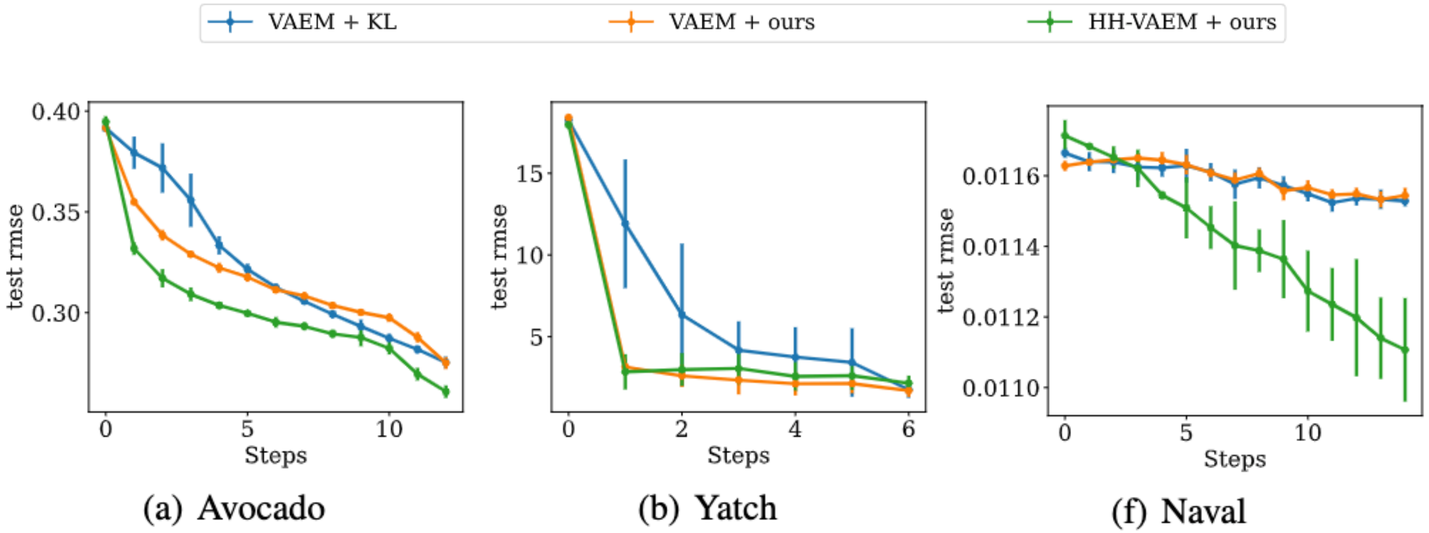


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

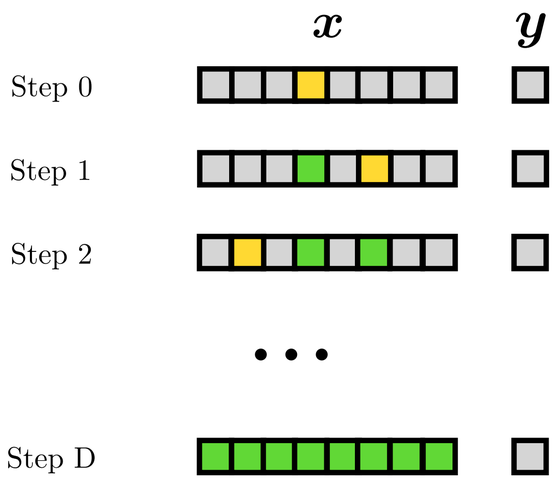


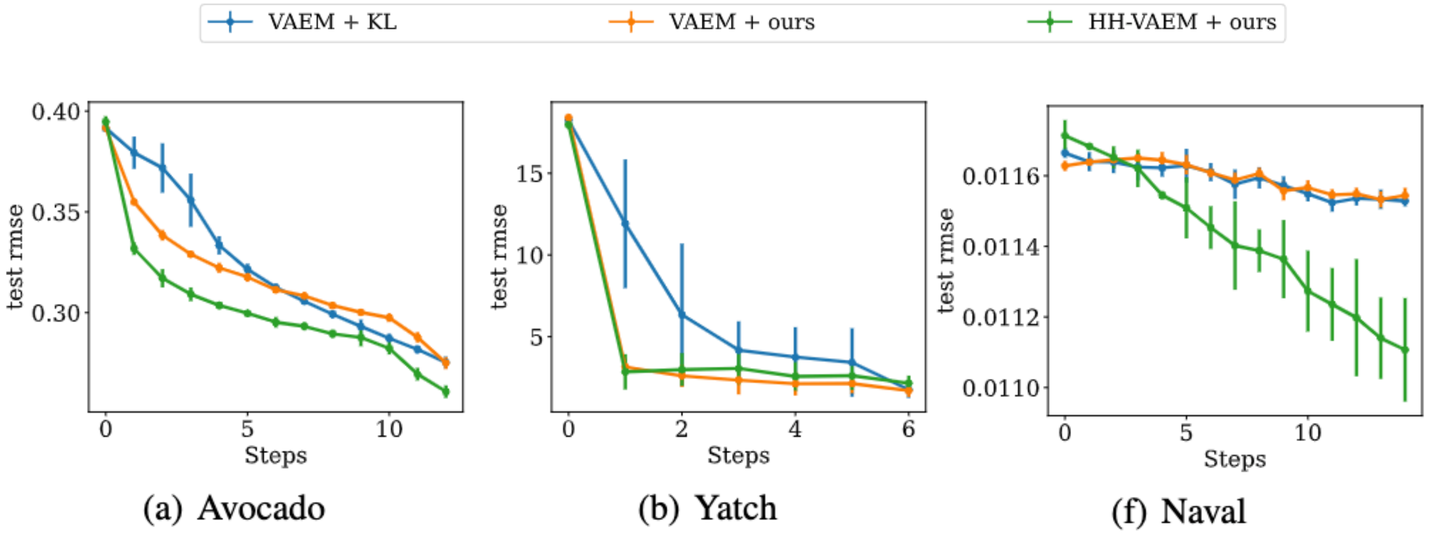


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)

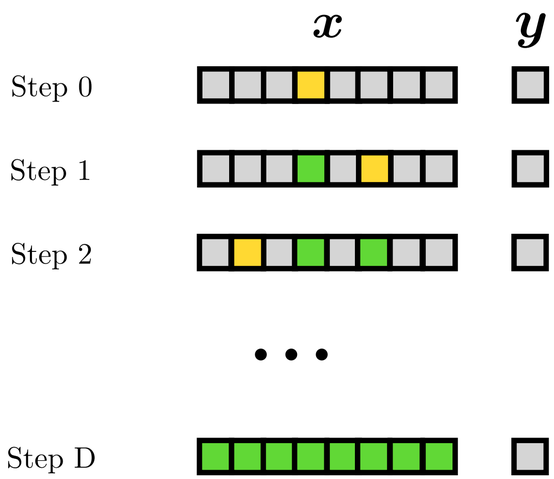


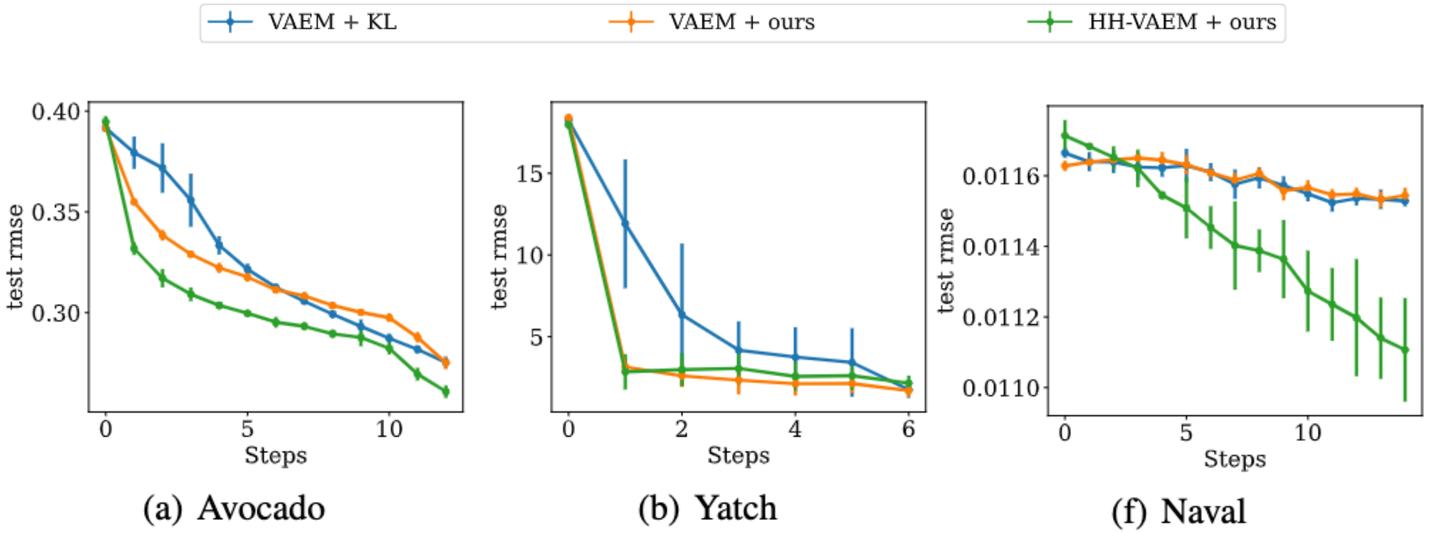


$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

Experiments

Sequential Active Information Acquisition (SAIA)



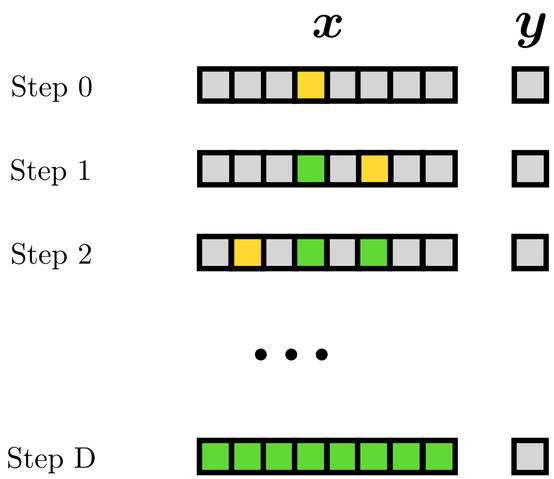


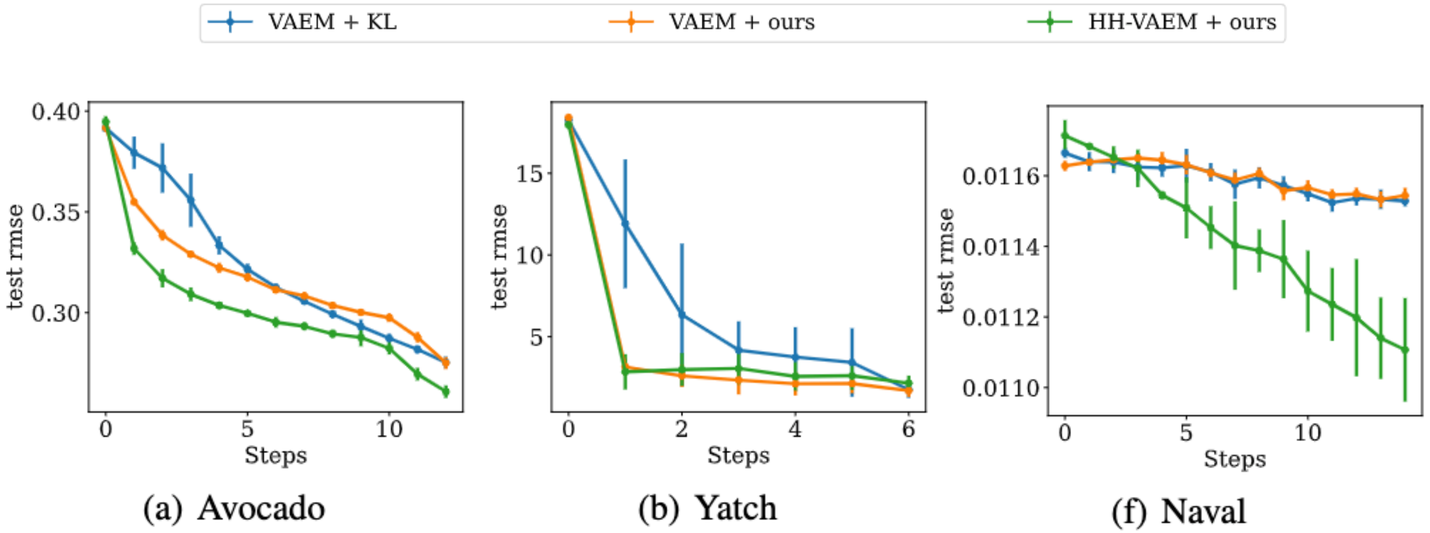
$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

$$\hat{R}\left(i, \boldsymbol{x}_o\right) = \mathbb{E}_{\hat{p}(\boldsymbol{x}_i|\boldsymbol{x}_o)} D_{KL}\left[q\left(\boldsymbol{z}|\boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z}|\boldsymbol{x}_o\right)\right] - \mathbb{E}_{\hat{p}(\boldsymbol{y}, \boldsymbol{x}_i|\boldsymbol{x}_o)} D_{KL}\left[q\left(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x}_o\right)\right]$$

Experiments

Sequential Active Information Acquisition (SAIA)





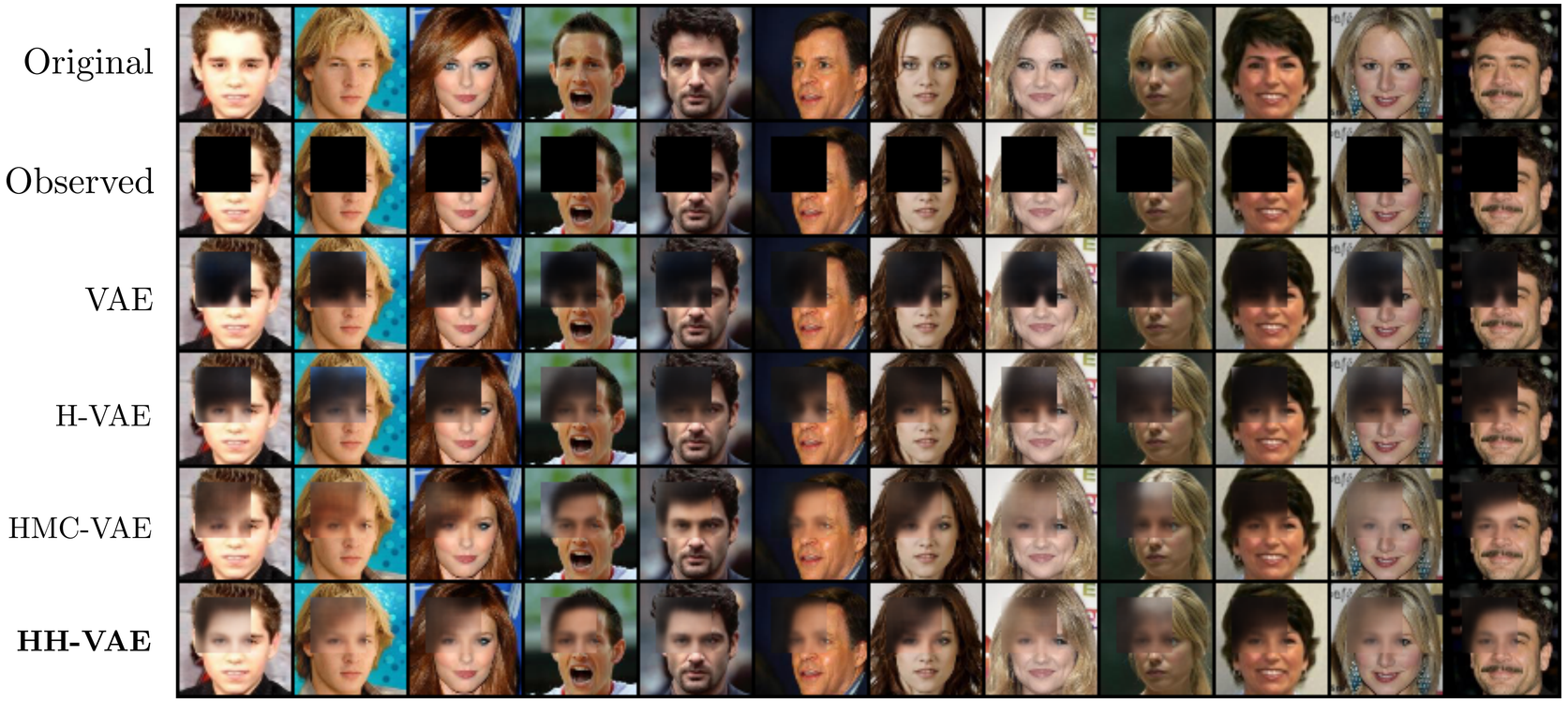
$$R(i, \boldsymbol{x}_O) = D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\mathbf{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$$

$$\hat{R}\left(i, \boldsymbol{x}_o\right) = \mathbb{E}_{\hat{p}(\boldsymbol{x}_i|\boldsymbol{x}_o)} D_{KL}\left[q\left(\boldsymbol{z}|\boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z}|\boldsymbol{x}_o\right)\right] - \mathbb{E}_{\hat{p}(\boldsymbol{y}, \boldsymbol{x}_i|\boldsymbol{x}_o)} D_{KL}\left[q\left(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x}_i, \boldsymbol{x}_o\right) || \, q\left(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x}_o\right)\right]$$

$$\hat{I}\left(\boldsymbol{y}; x_i \mid \boldsymbol{x}_O\right) \approx \sum_{ij} p_{x_i, \boldsymbol{y}|\boldsymbol{x}_O}(i, j) \log \frac{p_{x_i, \boldsymbol{y}|\boldsymbol{x}_O}(i, j)}{p_{x_i|\boldsymbol{x}_O}(i) p_{\boldsymbol{y}|\boldsymbol{x}_O}(j)}$$

Experiments

Conditional image inpainting



Outline

Contents

Conclusions

I. Peis, P. M. Olmos and A. Artés-Rodríguez. Unsupervised Learning of Global Factors in Deep Generative Models. In ***Pattern Recognition***, *134, 109130*, 2023.

I. Peis, C. Ma and J. M. Hernández-Lobato. Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (**NeurIPS**) 35*, 2022.

Conclusions

- ✓ VAEs can be used to learn global shared information in an unsupervised manner.
- ✓ The learned global representations are interpretable, leading to potential applications with real data.

I. Peis, P. M. Olmos and A. Artés-Rodríguez. Unsupervised Learning of Global Factors in Deep Generative Models. In ***Pattern Recognition***, *134, 109130*, 2023.

I. Peis, C. Ma and J. M. Hernández-Lobato. Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (**NeurIPS**) 35*, 2022.

Conclusions

- ✓ VAEs can be used to learn global shared information in an unsupervised manner.
- ✓ The learned global representations are interpretable, leading to potential applications with real data.
- ✓ We present the first Hierarchical VAE with HMC-based inference, outperforming Gaussian-based inference.
- ✓ This outperformance is also present in the following tasks:
  - Missing data imputation.
  - Target prediction.
  - Active information acquisition.

I. Peis, P. M. Olmos and A. Artés-Rodríguez. Unsupervised Learning of Global Factors in Deep Generative Models. In ***Pattern Recognition***, *134, 109130*, 2023.

I. Peis, C. Ma and J. M. Hernández-Lobato. Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (**NeurIPS**) 35*, 2022.

Future work

Technical research

- Global factors in sequential data [39,40].
- Hierarchical global latent spaces.
- Domain alignment with global factors [41,42].
- MCMC for enhancing disentanglement.
- MCMC sampling in other DGMs.

Applied research

- Learning global factors or efficient data acquisition in:
  - Clinical data.
  - Recommender systems.
  - Climate data.

[39] (Li and Mandt, 2018) [40] (Luo et al., 2020) [41] (Ilse et al., 2020) [42] (Nguyen et al., 2021)

Published Content

Conference papers

- **I. Peis**, C. Ma and J. M. Hernández-Lobato. Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (**NeurIPS**) 35*, 2022.
- B. Koyuncu, P. Sanchez-Martin, **I. Peis**, P. M. Olmos and I. Valera. Variational Mixture of HyperGenerators for Learning Distributions Over Functions. In *Proceedings of the 40th International Conference on Machine Learning (**ICML**)*, 2023.

Journal papers

- **I. Peis**, P. M. Olmos and A. Artés-Rodríguez. Unsupervised Learning of Global Factors in Deep Generative Models. In ***Pattern Recognition**, 134, 109130*, 2023.
- **I. Peis**, J. D. López-Moríñigo, M. M. Pérez-Rodríguez, M. L. Barrigón, M. Ruiz-Gómez, A. Artés-Rodríguez and E. Baca-García. Actigraphic recording of motor activity in depressed inpatients: a novel computational approach to prediction of clinical course and hospital discharge. In *Scientific Reports, 10. Nature*, 2020.
- **I. Peis**, P. M. Olmos, C. Vera-Varela, M. L. Barrigón, P. Courtet, E. Baca-García and A. Artés-Rodríguez. Deep Sequential Models for Suicidal Ideation from Multiple Source Data. In *IEEE Journal of Biomedical and Health Informatics, 23(6), 2286-2293*, 2020.

References

- [1] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. arXiv preprint arXiv:1312.6114, 2013.
- [2] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, & M. Chen, (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, *1*(2), 3.
- [3] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, ... & A. Aspuru-Guzik,(2018). Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, *4*(2), 268-276.
- [4] C. Cremer, X. Li, and D. Duvenaud. Inference Suboptimality in Variational Autoencoders. In International Conference on Machine Learning, pages 1078-1086. PMLR, 2018.
- [5] Y. Burda, R. Grosse, and R. Salakhutdinov. Importance Weighted Autoencoders. arXiv preprint arXiv:1509.00519, 2015.
- [6] T. Salimans, D. Kingma, and M. Welling. Markov Chain Monte Carlo and Variational Inference: Bridging the gap. In International Conference on Machine Learning, pages 1218-1226. PMLR, 2015.
- [7] F. J. Ruiz, M. K. Titsias, T. Cemgil, and A. Doucet. Unbiased Gradient Estimation for Variational Auto-Encoders using Coupled Markov Chains. In Uncertainty in Artificial Intelligence, pages 707-717. PMLR, 2021.
- [8] A. L. Caterini, A. Doucet, and D. Sejdinovic. Hamiltonian Variational Auto-Encoder. arXiv preprint arXiv:1805.11328, 2018.
- [9] A. Campbell, W. Chen, V. Stimper, J. M. Hernandez-Lobato, and Y. Zhang. A Gradient- Based Strategy for Hamiltonian Monte Carlo Hyperparameter optimization. In International Conference on Machine Learning, pages 1238-1248. PMLR, 2021.
- [10] N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan. Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders. arXiv preprint arXiv:1611.02648, 2016.
- [11] J. Tomczak and M. Welling. VAE with a VampPrior. In International Conference on Arti cial Intelligence and Statistics, pages 1214-1223. PMLR, 2018.
- [12] A. Vahdat and J. Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. arXiv preprint arXiv:2007.03898, 2020.
- [13] R. Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. arXiv preprint arXiv:2011.10650, 2020.
- [14] L. Maaløe, M. Fraccaro, V. Li evin, and O. Winther. BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. arXiv preprint arXiv:1902.02102, 2019.
- [15] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. IEEE transactions on pattern analysis and machine intelligence, 35(8):1798-1828, 2013.
- [16] C. Eastwood and C. K. Williams. A Framework for the Quantitative Evaluation of Disentangled Representations. In International Conference on Learning Representations, 2018.

References

- [17] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. 2016.
- [18] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Scholkopf, and O. Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In international conference on machine learning, pages 4114{4124, 2019b.
- [19] E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh. Disentangling Disentanglement in Variational Autoencoders. In International Conference on Machine Learning, pages 4402-4412, 2019b.
- [20] D. Bouchacourt, R. Tomioka, and S. Nowozin. Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
- [21] I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. MIT press, 2016.
- [22] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556, 2014.
- [23] I. Peis, P. M. Olmos, C. Vera-Varela, M. L. Barrigón, P. Courtet, E. Baca-García, and A. Artés-Rodriguez. Deep Sequential Models for Suicidal Ideation from Multiple Source Data. IEEE journal of biomedical and health informatics, 23(6):2286-2293, 2019.
- [24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. Communications of the ACM, 63(11):139-144, 2020.
- [25] J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. Advances in Neural Information Processing Systems, 33:6840-6851, 2020.
- [26] B. Koyuncu, P. Sanchez-Martin, I. Peis, P. M. Olmos, and I. Valera. Proceedings of the 40th International Conference on Machine Learning, 2023.
- [27] M. J. Vowels, N. C. Camgoz, and R. Bowden. NestedVAE: Isolating Common Factors via Weak Supervision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9202-9212, 2020.
- [28] P. Bromiley. Products and Convolutions of Gaussian Probability Density Functions. Tina-Vision Memo, 3(4):1, 2003.
- [29] C. Ma, S. Tschiatschek, K. Palla, J. M. Hern andez-Lobato, S. Nowozin, and C. Zhang. EdDI: Efficient Dynamic Discovery of High-Value Information with Partial VAE. arXiv preprint arXiv:1809.11142, 2018.
- [30] C. Ma, S. Tschiatschek, J. M. Hern andez-Lobato, R. Turner, and C. Zhang. VAEM: a Deep Generative Model for Heterogeneous Mixed Type Data. arXiv preprint arXiv:2006.11941, 2020.

References

- [31] A. Nazabal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling Incomplete Heterogeneous Data Using VAEs. Pattern Recognition, page 107501, 2020.
- [32] P.-A. Mattei and J. Frellsen. MIWAE: Deep Generative Modelling and Imputation of Incomplete Data Sets. In International Conference on Machine Learning, pages 4413-4423. PMLR, 2019.
- [33] J. M. Bernardo and A. F. Smith. Bayesian Theory, volume 405. John Wiley & Sons, 2009.
- [34] A. Kraskov, H. Stogbauer, and P. Grassberger. Estimating Mutual Information. Physical review E, 69(6):066138, 2004.
- [35] R. M. Neal. Hamiltonian Importance Sampling. In talk presented at the Ban International Research Station (BIRS) workshop on Mathematical Issues in Molecular Dynamics, 2005.
- [36] M. Betancourt. A conceptual introduction to Hamiltonian Monte Carlo. arXiv preprint arXiv:1701.02434, 2017.
- [37] W. Gong, Y. Li, and J. M. Hernández-Lobato. Sliced Kernelized Stein Discrepancy. arXiv preprint arXiv:2006.16531, 2020.
- [38] M. Betancourt and M. Girolami. Hamiltonian Monte Carlo for Hierarchical Models. Current trends in Bayesian methodology with applications, 79(30):2-4, 2015.
- [39] Y. Li and S. Mandt. Disentangled Sequential Autoencoder. arXiv preprint arXiv:1803.02991, 2018.
- [40] Y.-J. Luo, K. W. Cheuk, T. Nakano, M. Goto, and D. Herremans. Unsupervised Disentanglement of Pitch and Timbre for Isolated Musical Instrument Sounds. In ISMIR, pages 700-707, 2020.
- [41] M. Ilse, J. M. Tomczak, C. Louizos, and M. Welling. DIVA: Domain Invariant Variational Autoencoders. In Medical Imaging with Deep Learning, pages 322-348. PMLR, 2020.
- [42] A. T. Nguyen, T. Tran, Y. Gal, and A. G. Baydin. Domain Invariant Representation Learning with Domain Density Transformations. Advances in Neural Information Processing Systems, 34:5264-5275, 2021.

Thank you!