Latent Space Oddity On The Curvature Of Deep Generative Models

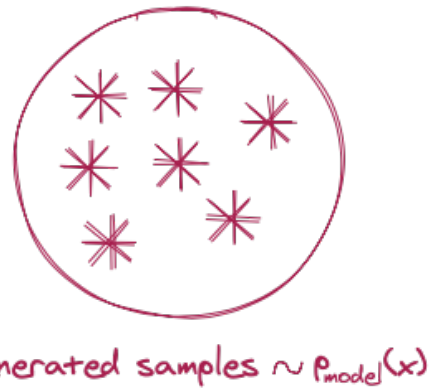
Clément GRISI, Timothée DARCET

6 janvier 2020

Generative Models

Goal

Given training data, generate new samples from the same distribution



learn $p_{\text{model}}(x)$ close to $p_{\text{data}}(x)$

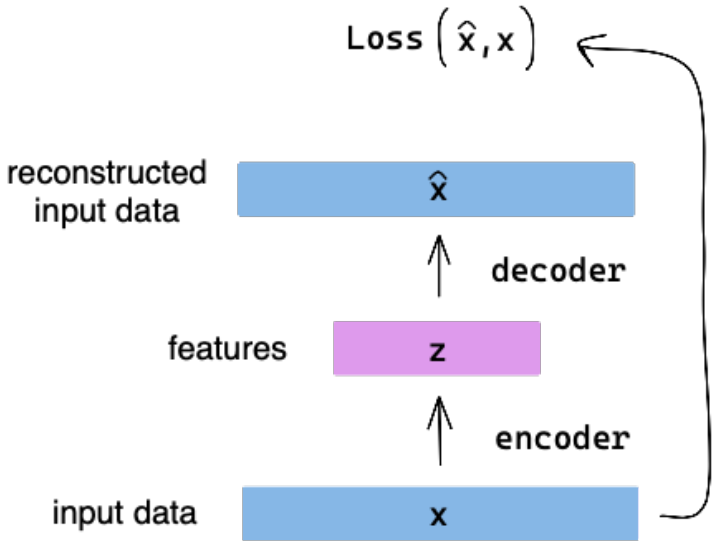
Density estimation

- core problem in the **unsupervised** learning setting
- explicit : explicitly define and solve $p_{\text{model}}(x)$
- implicit : learn a model that can sample from $p_{\text{model}}(x)$ without explicitly defining it

Variational Auto Encoders

Auto encoders

unsupervised approach for learning a lower dimensional feature representation **z** from unlabeled training data *x*



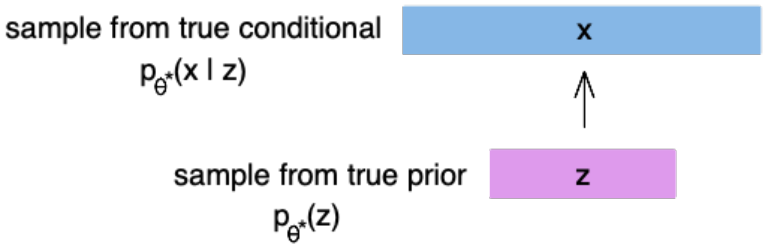
z captures meaningful factors of variation in the data

Question : can we generate new images from an auto encoder ?

Variational auto encoders

VAEs are a probabilistic spin on auto encoders that will let us sample from the model to generate data

Assumption : training data $\{x_i\}_{i\in[1,N]}$ is generated from underlying (latent) **unobserved** representation **z**



Goal : estimate the true parameters θ^* of this generative model

Question : how do we train the model ?

Intractability

Choose simple prior $p(z)$ (e.g. Gaussian)
Conditional $p(x|z)$ is **complex** : represent it with a neural network

Natural strategy : learn model parameters to maximize the likelihood of the training data

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

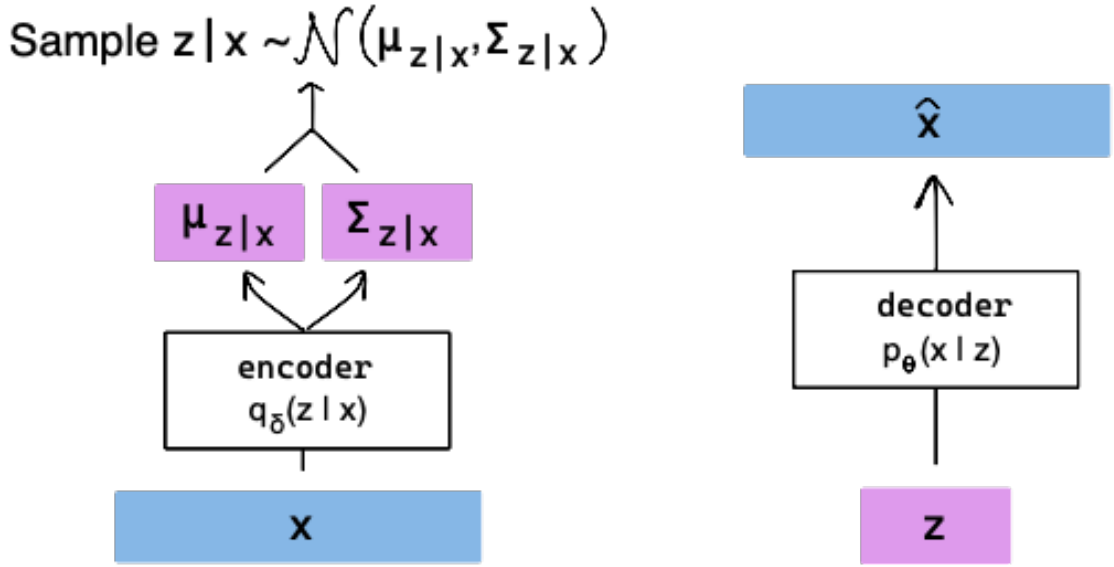
Though we know $p_\theta(x|\mathbf{z})$ and $p_\theta(\mathbf{z})$, it is **intractable** to compute $p_\theta(x|\mathbf{z})$ for every z !

Posterior density is also **intractable** :

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

Solution

In addition to the **decoder** network modeling $p_\theta(x|z)$, define an additional **encoder** network $q_\delta(z|x)$ that approximates $p_\theta(z|x)$.



This allows us to derive a **tractable** lower bound on the data likelihood

Tractable lower bound

$$
\begin{aligned}
\log p_\theta(x) &= \mathbb{E}_{z\sim q_\delta(\cdot|x)}\left[\log p_\theta(x)\right] \\
&= \mathbb{E}_z\left[\log \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(z|x)}\right] \\
&= \mathbb{E}_z\left[\log \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(z|x)}\frac{q_\delta(z|x)}{q_\delta(z|x)}\right] \\
&= \mathbb{E}_z\left[\log p_\theta(x|z)\right] - \mathbb{E}_z\left[\log \frac{q_\delta(z|x)}{p_\theta(z)}\right] + \mathbb{E}_z\left[\log \frac{q_\delta(z|x)}{p_\theta(z|x)}\right] \\
&= \mathbb{E}_z\left[\log p_\theta(x|z)\right] - d_{\mathrm{KL}}\left(q_\delta(z|x)||p_\theta(z)\right) + d_{\mathrm{KL}}\left(q_\delta(z|x)||p_\theta(z|x)\right)
\end{aligned}
$$

Tractable lower bound

- decoder network gives $p_\theta(x|z)$: we can estimate $\mathbb{E}_z[\log p_\theta(x|z)]$ through sampling (differentiable with the reparametrization trick [1])
- $d_{\mathrm{KL}}(q_\delta(z|x)||p_\theta(z))$ is the KL-div of two Gaussian distributions : it has a nice closed from solution (differentiable)
- though $p_\theta(z|x)$ is intractable, we know KL-div is always positive : $d_{\mathrm{KL}}(q_\delta(z|x)||p_\theta(z|x)) \geq 0$

Hence,

$$\begin{aligned}\log p_\theta(x) &\geq \mathbb{E}_z[\log p_\theta(x|z)] - d_{\mathrm{KL}}(q_\delta(z|x)||p_\theta(z)) \\ &\geq \mathscr{L}(x,\theta,\delta)\end{aligned}$$

We've identified a **tractable** + **differentiable** lower bound which we can take gradient of and optimize (ELBO, evidence lower bound)

Training

Optimal parameters will be the ones that maximize this lower bound :

$$\theta^*, \delta^* = \arg\max_{\theta,\delta} \sum_{i=1}^{N} \mathscr{L}(x_i, \theta, \delta)$$

In order to find them with stochastic gradient descent, we need to compute and backpropagate $\nabla_\theta \mathscr{L}(\theta, \delta)$ and $\nabla_\delta \mathscr{L}(\theta, \delta)$.

Computing $\nabla_\theta \mathscr{L}(x_i, \theta, \delta)$ for a given input data x_i is relatively straightforward. However, computing $\nabla_\delta \mathscr{L}(x_i, \theta, \delta)$ is harder as the parameter δ appears in the distribution with respect to which the expectation $\mathbb{E}_{z \sim q_\delta(\cdot|x_i)}[\log p_\theta(x_i|z)]$ is taken :

$$\nabla_\delta \mathbb{E}_{z \sim q_\delta(\cdot|x_i)}[\log p_\theta(x_i|z)] \quad \neq \quad \mathbb{E}_{z \sim q_\delta(\cdot|x_i)}[\nabla_\delta \log p_\theta(x_i|z)]$$

Reparametrization trick

Our goal is to rewrite this expectation in such a way that δ appears inside the expectation only :

$$\mathbb{E}_{z\sim q_\delta(\cdot|x_i)}\left[\log p_\theta(x_i|z)\right] = \mathbb{E}_{p_\varepsilon}\left[\log p_\theta\left(x_i|g_\delta(\varepsilon, x_i)\right)\right]$$

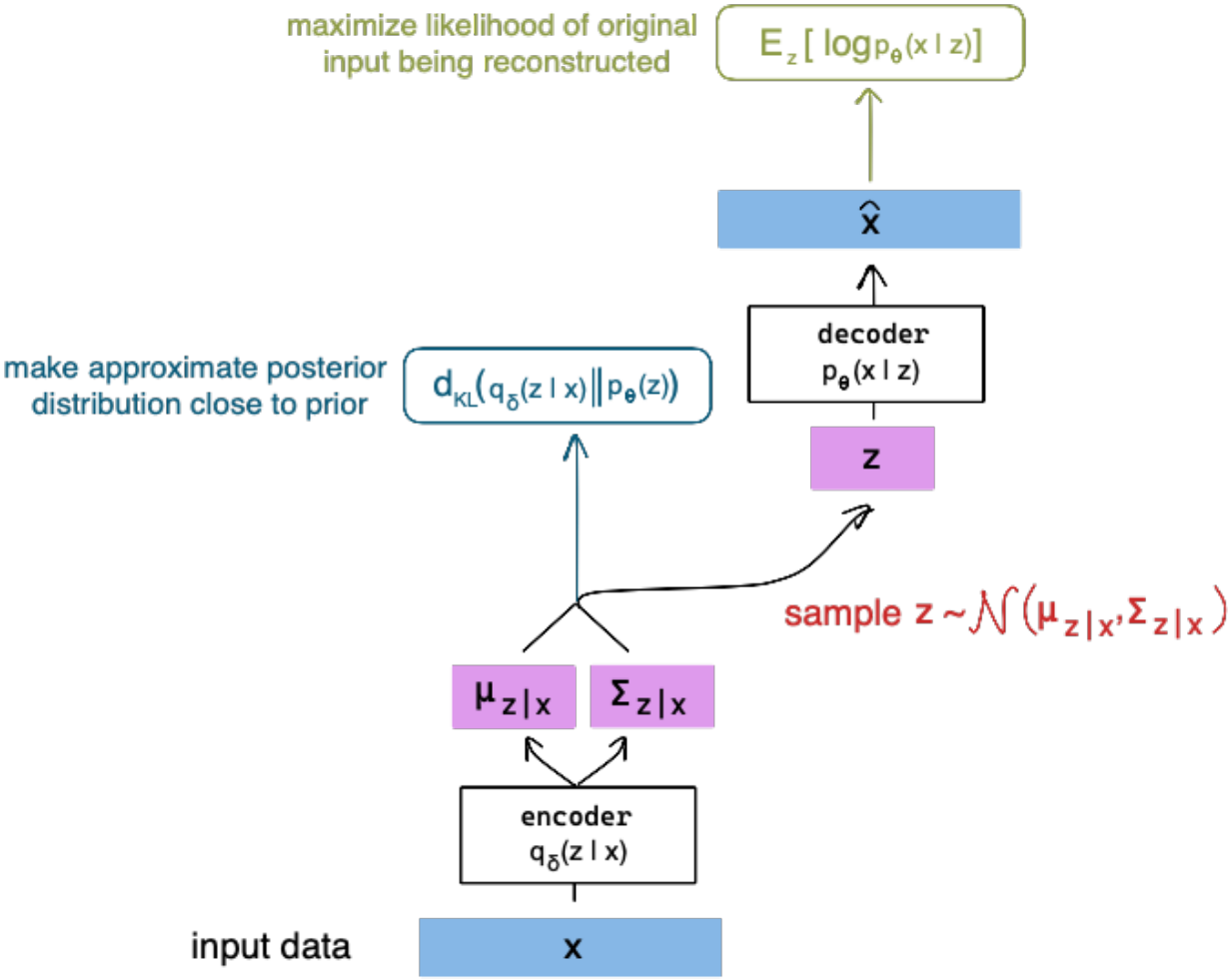
such that $g_\delta(\varepsilon, x_i) = z \sim \mathcal{N}\left(\mu_{z|x_i}, \Sigma_{z|x_i}\right)$

We can sample from $\mathcal{N}\left(\mu_{z|x_i}, \Sigma_{z|x_i}\right)$ by first sampling $\varepsilon \sim \mathcal{N}(0,1)$ and computing :

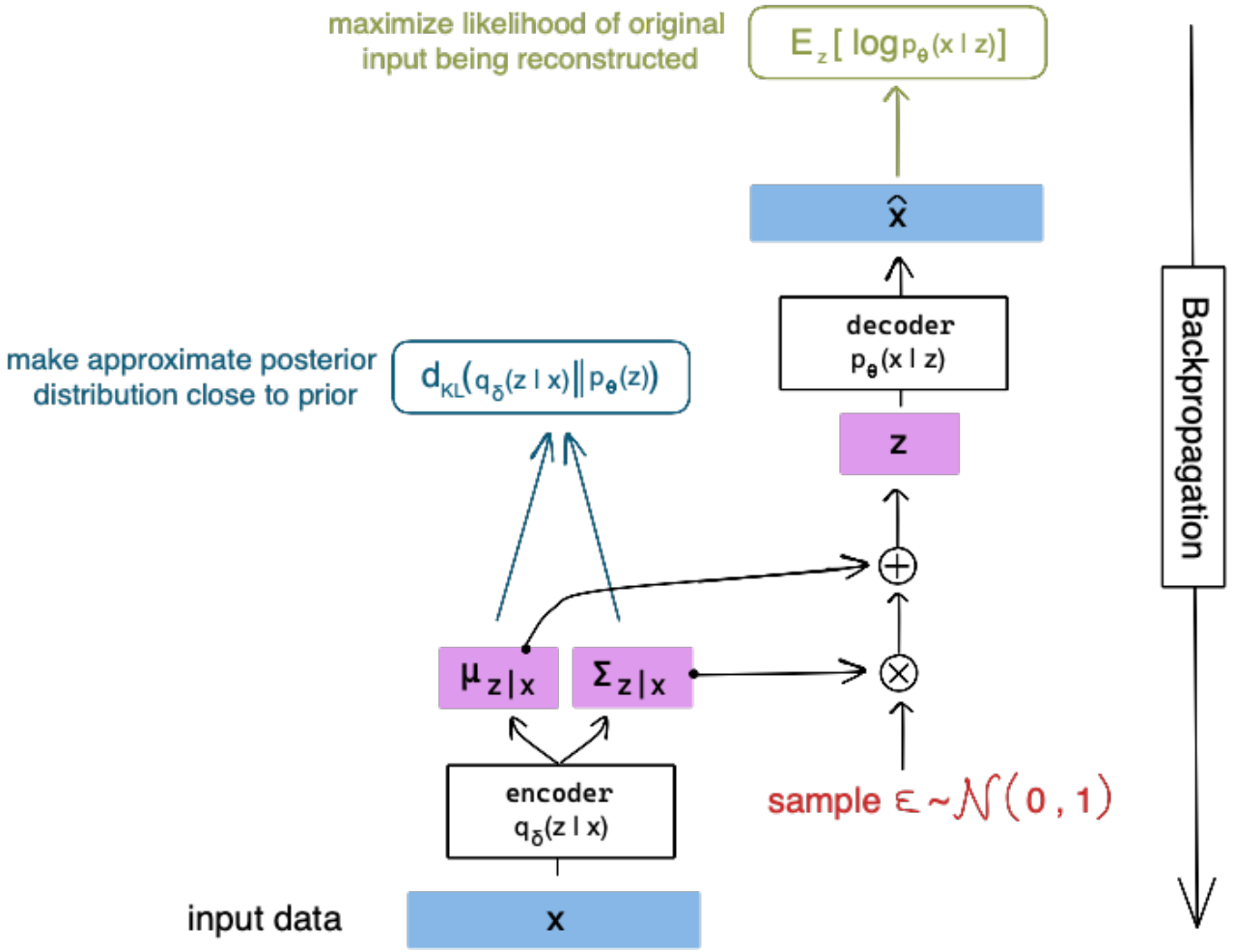
$$z = g_\delta(\varepsilon, x_i) = \mu_{z|x_i} + \left(\Sigma_{z|x_i}\right)^{\frac{1}{2}} \varepsilon$$

This function g is a simple linear transform (deterministic) and allows us to compute $\nabla_\phi \mathbf{L}(\theta, \phi)$.

Reparametrization trick

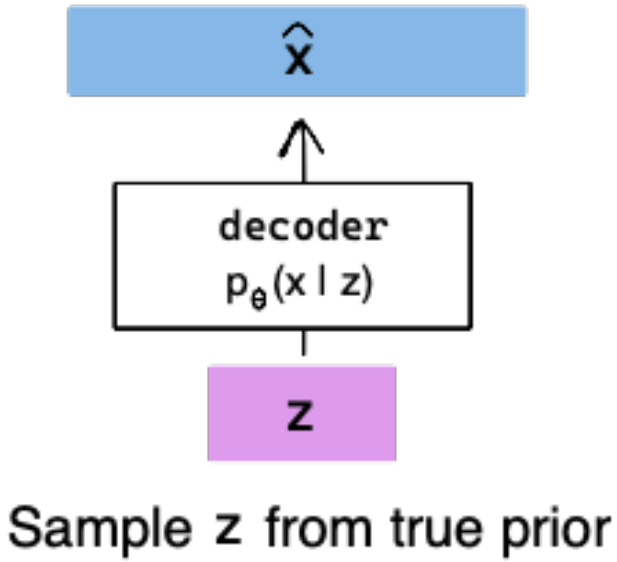


Reparametrization trick



Data generation

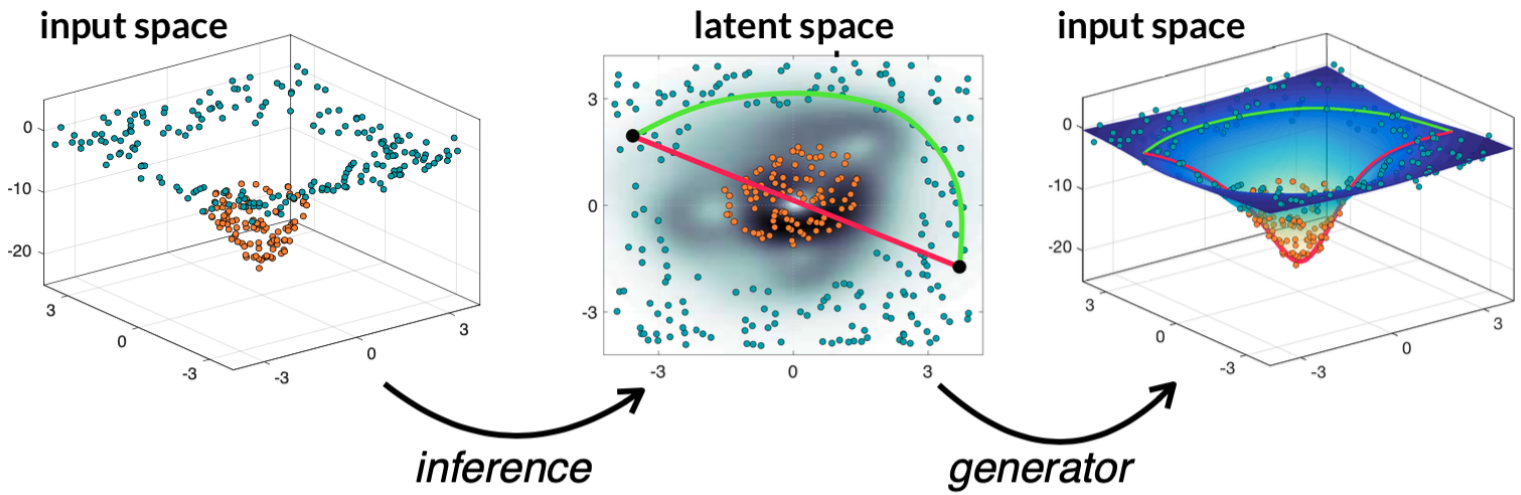
Once trained, we can sample **z** from prior and use the decoder network to generate new data :



Latent Space Oddity

A misinterpretation of the latent space

Goal : equiped with a good latent space, points from the same class should be clother to each other than to members of the other classes



Issue : this doesn't *seem* to be the case (green path is actually shorter than red path)

This *seemed* conclusion is incorrect
→ this is due to a misinterpretation of the latent space : it shouldn't be seen as a linear Euclidian space but rather as a **curved** space

This curvature induces a Riemannian metric which gives more meaningful distances than the usual Eulidian distance

Geometry of the decoder

VAE decoder

The decoder driving VAEs is **stochastic** :

$$f(z) = \mu(z) + \sigma(z) \odot \varepsilon \quad (\varepsilon \sim \mathcal{N}(0, \mathbb{I}_D))$$

→ this implies a **stochastic** Riemannian geometry

Theorem 1

if stochastic decoder is at least twice differentiable, the expected metric equals :

$$\overline{\mathrm{M}}_z = \mathbb{E}_{p(\varepsilon)}\left[\mathrm{M}_z\right] = \left(\mathrm{J}_z^{(\mu)}\right)^\top \left(\mathrm{J}_z^{(\mu)}\right) + \left(\mathrm{J}_z^{(\sigma)}\right)^\top \left(\mathrm{J}_z^{(\sigma)}\right)$$

where $\mathrm{J}_z^{(\mu)}$ and $\mathrm{J}_z^{(\sigma)}$ are the jacobian matrices of $\mu(\cdot)$ and $\sigma(\cdot)$

VAE decoder

Coupled with Theorem 2, we get that the (deterministic) expected metric $\overline{\mathrm{M}}_z$ is a good approximation to M_z when the data dimension is large : we can use the theory of deterministic decoders

Theorem 2

$$\lim_{D\to\infty} \mathrm{Var}\left(\mathrm{M}_z\right) = 0$$

The resulting induced distance will be large in regions of the latent space where the decoder is highly uncertain (high variance)
→ shortest paths will tend to avoid these regions

Ensuring proper geometry

The geometry of a VAE depends on both the mean and the variance of its decoder
→ successful training will provide good estimates of the geometry in regions near the data

The decoder is expected to have high variance in regions with no data
In practice, variance estimates are extrpolated to regions without data : as neural nets extrapolate poorly, practical variance estimates tend to be poor in regions with no data...

Issue : we need well-behaved variance functions to ensure a well-behaved geometry

Regularizing the geometry

Solution : model decoder's *precision* instead of its variance

add more details

The measure element : $\sqrt{\det(M_z)}$

add more details

In constrast to the standard model, the proposed model provides a clear geometrical structure that follows the trend of the data

Conclusion

- standard model assigns arbitrary density in regions of the input space where data has not been seen during training, while the proposed model assigns minimal density to such regions
- the variance term of the standard model behaves arbitrarily in regions with no latent codes, whereas the proposed model assigns high variance in those regions

Result : proposed model achieves better marginal likelihood on held-out data

Experiments

Experiments

Bibliography :


Carl Doersch.
Tutorial on variational autoencoders, 2021.


Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg.
Latent space oddity : on the curvature of deep generative models, 2018.