

# Variational Autoencoders

## Advanced Topics in Machine Learning

---

Cesare Alippi, Andrea Cini

September 22, 2021

Università della Svizzera italiana

# Table of contents

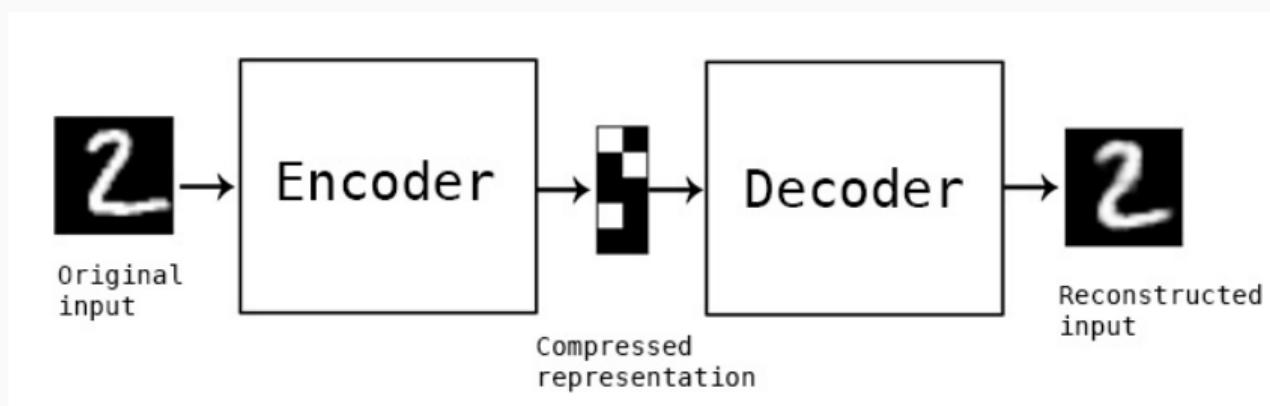
1. Preliminaries
2. Variational Autoencoders
3. Demo

## Preliminaries

---

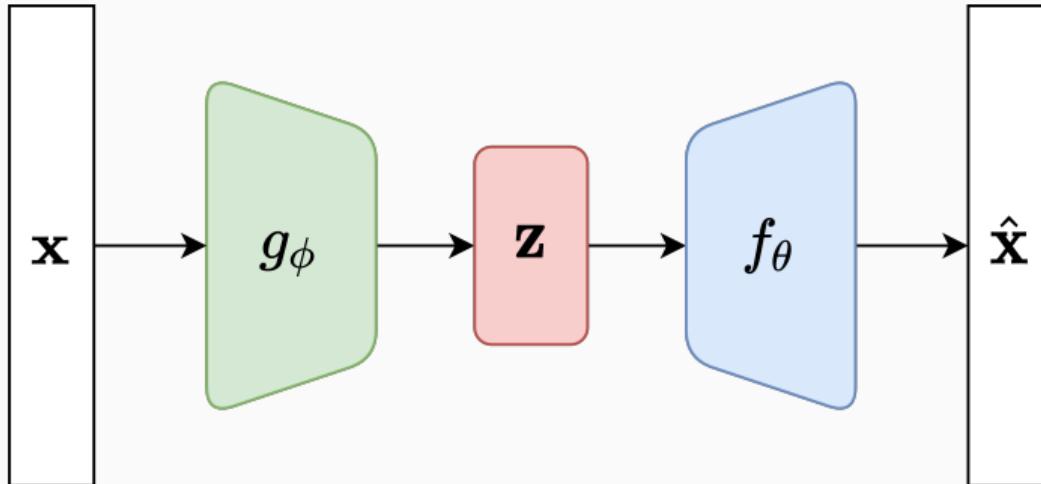
## Autoencoders (I)

General idea behind autoencoders:



- We want to learn a **low-dimensional representation** of our data.
- We want to be able go back and forth from the **encoding space** and the original space.

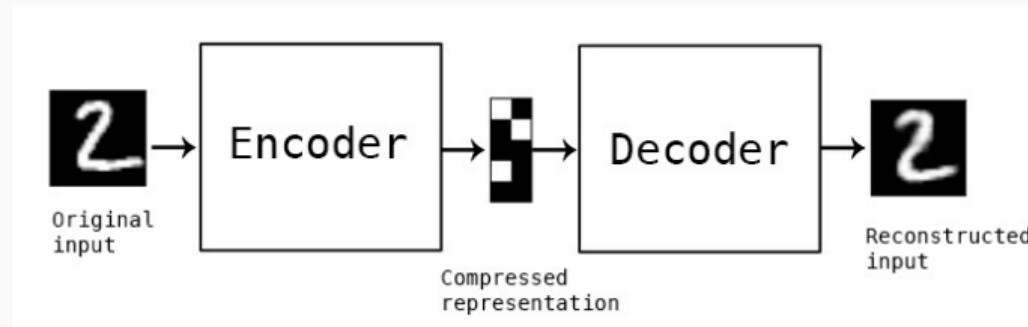
## Autoencoders (II)



**Figure 1:** A vanilla Autoencoder (AE)

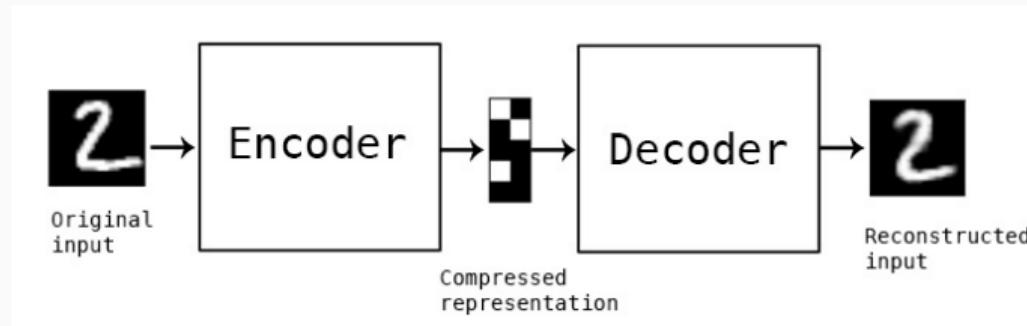
- Use a bottleneck Encoder-Decoder architecture to force a compact representation.
- Train a neural network to learn the identity function, i.e., to reconstruct the input.

## Autoencoders (III)



What are Autoencoders good for?

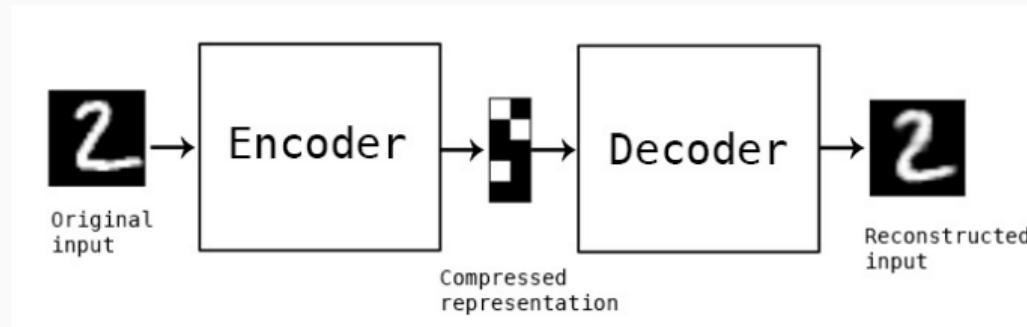
## Autoencoders (III)



What are Autoencoders good for?

- Dimensionality reduction

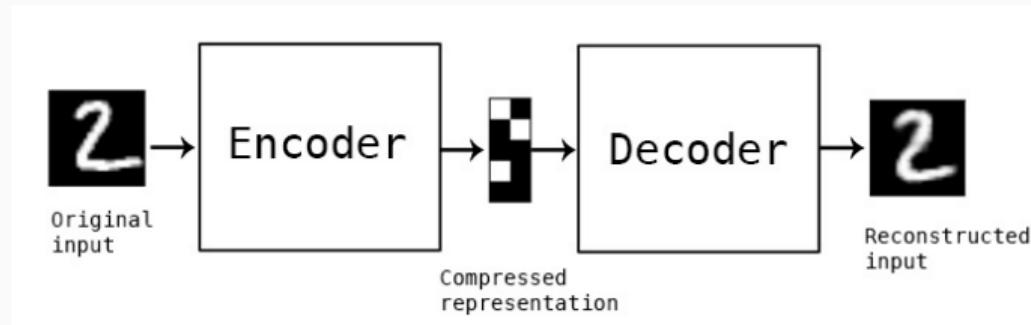
## Autoencoders (III)



What are Autoencoders good for?

- Dimensionality reduction
- Representation learning

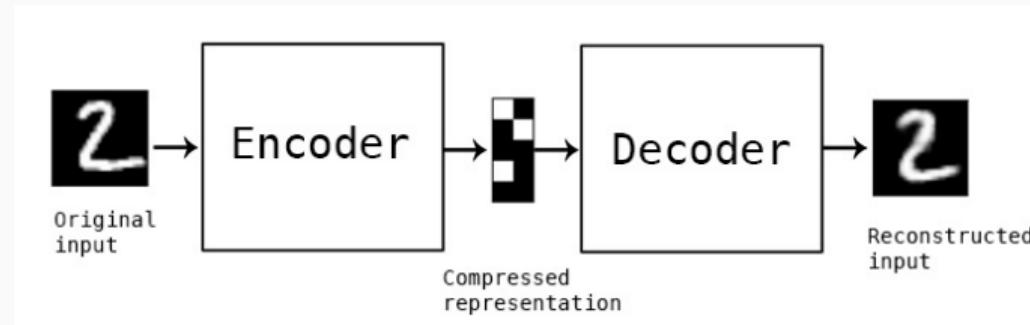
## Autoencoders (III)



What are Autoencoders good for?

- Dimensionality reduction
- Representation learning
- Removing noise from the input data

## Autoencoders (III)



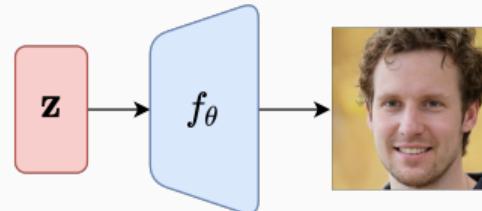
What are Autoencoders good for?

- Dimensionality reduction
- Representation learning
- Removing noise from the input data
- Generating new samples, but...

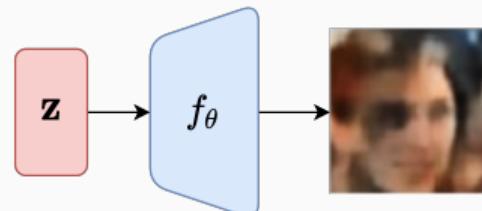
## Autoencoders (IV)

We can try to generate new samples sampling at random the encoding space:

What we want

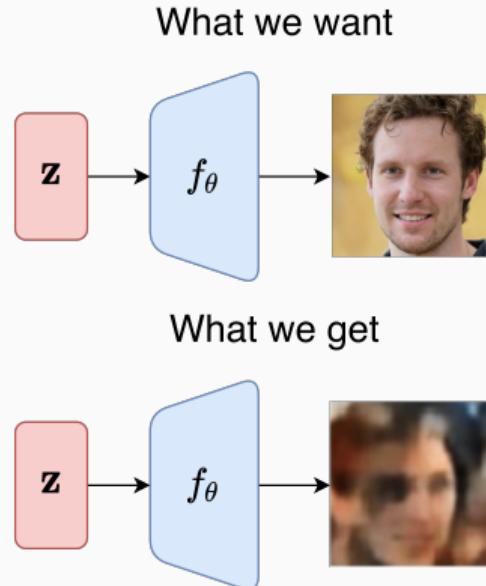


What we get



## Autoencoders (IV)

We can try to generate new samples sampling at random the encoding space:



- Why?

## Discriminative vs Generative models (I)

Standard AEs are **discriminative** models.

- Discriminative models learn to make predictions, i.e., learn a mapping from input to an output.
- AEs are trained to reconstruct the input, **not** to generate new data.
- It is hard to say something about the structure of the encoding space of a standard AE ...

## Discriminative vs Generative models (I)

Standard AEs are **discriminative** models.

- Discriminative models learn to make predictions, i.e., learn a mapping from input to an output.
- AEs are trained to reconstruct the input, **not** to generate new data.
- It is hard to say something about the structure of the encoding space of a standard AE ...
- ... it is not clear how to sample it to generate **new datapoints**: a point in the latent space does not necessarily correspond to a plausible observation.

## Discriminative vs Generative models (II)

Variational Autoencoders (VAEs)<sup>1</sup> are generative models.

- Generative approaches model the stochastic process generating the data and predictions are done using Bayes rule.
- Forcing the representations to be useful for generation often results in *better generalization*.

---

<sup>1</sup>Kingma and Welling 2013; Kingma and Welling 2019.

## Discriminative vs Generative models (II)

Variational Autoencoders (VAEs)<sup>1</sup> are generative models.

- Generative approaches model the stochastic process generating the data and predictions are done using Bayes rule.
- Forcing the representations to be useful for generation often results in *better generalization*.
- VAEs learn to (1) infer the latent representation behind each observation and (2) generate realistic synthetic observations from points in the latent space.

---

<sup>1</sup>Kingma and Welling 2013; Kingma and Welling 2019.

## Discriminative vs Generative models (II)

Variational Autoencoders (VAEs)<sup>1</sup> are generative models.

- Generative approaches model the stochastic process generating the data and predictions are done using Bayes rule.
- Forcing the representations to be useful for generation often results in *better generalization*.
- VAEs learn to (1) infer the latent representation behind each observation and (2) generate realistic synthetic observations from points in the latent space.

In order to understand VAEs we need to introduce the concept of Latent Variable Models.

---

<sup>1</sup>Kingma and Welling 2013; Kingma and Welling 2019.

## Latent Variable Models (I)

- Given a dataset  $\mathcal{D}$ , consider each sample  $\mathbf{x}$  as a realization of a random vector, i.e.,  $\mathbf{x} \sim p^*(\mathbf{x})$ .
- $p^*(\mathbf{x})$  is the unknown probability distribution underlying the process **generating** the data.
- We want to learn an approximating distribution, with a model parametrized by  $\theta$ , such that:

$$p_\theta(\mathbf{x}) \approx p^*(\mathbf{x});$$

$p_\theta(\mathbf{x})$  is known as **model evidence**. The notation  $p_\theta(\cdot)$  indicates that probability distribution  $p_\theta(\cdot)$  is a function of model parameters  $\theta$ .

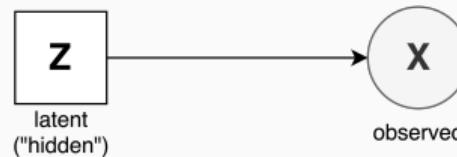
## Latent Variable Models (II)

A *Latent Variable Model (LVM)* uses an approximating distribution of the form:

$$p_{\theta}(x) = \int p_{\theta}(x, z) dz \quad (1)$$

$$= \int p_{\theta}(x|z)p_{\theta}(z) dz, \quad (\text{product rule}) \quad (2)$$

where  $p_{\theta}(x|z)$  is the **likelihood** of observation  $x$  given the latent variables  $z$  with **prior distribution**  $p_{\theta}(z)$ . In the following we refer to the case where  $z$  is a finite dimensional random vector and the likelihood is a known distribution, e.g., Gaussian.



## Latent Variable Models (III)

At this point you might have an intuition about what is going on...

---

## Latent Variable Models (III)

At this point you might have an intuition about what is going on...

- Sampling from  $p_{\theta}(x|z)$  we could generate new observations.

## Latent Variable Models (III)

At this point you might have an intuition about what is going on...

- Sampling from  $p_{\theta}(x|z)$  we could generate new observations.
- Symmetrically, modeling  $p_{\theta}(z|x)$ , known as the **posterior distribution**, we could make inference about the latent variables given an observation...

## Latent Variable Models (III)

At this point you might have an intuition about what is going on...

- Sampling from  $p_{\theta}(x|z)$  we could generate new observations.
- Symmetrically, modeling  $p_{\theta}(z|x)$ , known as the **posterior distribution**, we could make inference about the latent variables given an observation...
- ... unfortunately computing the posterior is **intractable** for most models<sup>2</sup>!

---

<sup>2</sup>To see why that is the case note that  $p_{\theta}(z|x) = \frac{p_{\theta}(x,z)}{p_{\theta}(x)}$  and that computing  $p_{\theta}(x)$  involves computing the integral in Eq. 1. See (Blei, Kucukelbir, and McAuliffe 2017) for a concrete example.

## Latent Variable Models (III)

At this point you might have an intuition about what is going on...

- Sampling from  $p_{\theta}(x|z)$  we could generate new observations.
- Symmetrically, modeling  $p_{\theta}(z|x)$ , known as the **posterior distribution**, we could make inference about the latent variables given an observation...
- ... unfortunately computing the posterior is **intractable** for most models<sup>2</sup>!
- **Solution:** we use an **approximate posterior**  $q_{\phi}(z|x) \approx p_{\theta}(z|x)$  (*we'll come back on this later*)

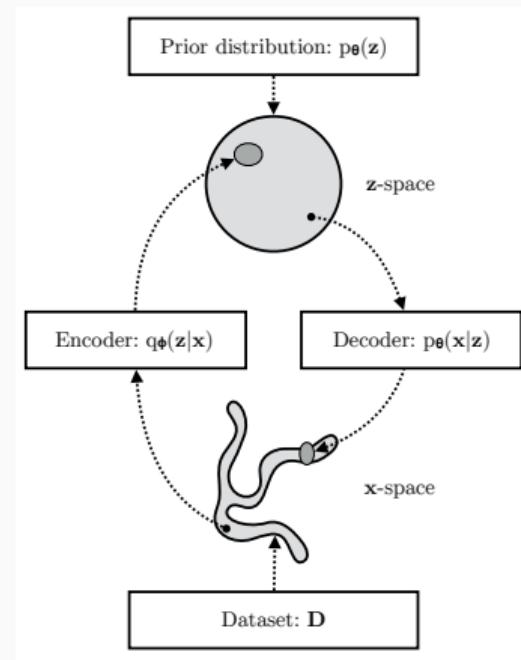
---

<sup>2</sup>To see why that is the case note that  $p_{\theta}(z|x) = \frac{p_{\theta}(x,z)}{p_{\theta}(x)}$  and that computing  $p_{\theta}(x)$  involves computing the integral in Eq. 1. See (Blei, Kucukelbir, and McAuliffe 2017) for a concrete example.

## Variational Autoencoders

---

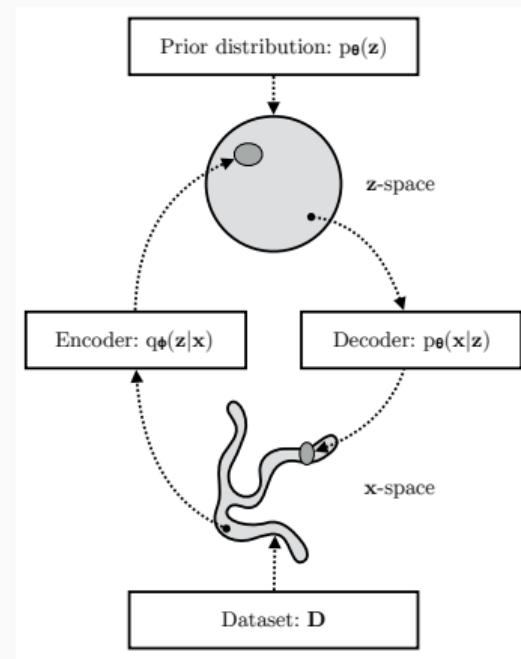
# Towards VAEs: the big picture



- The input space is a generic manifold.

<sup>3</sup>Kingma and Welling 2019.

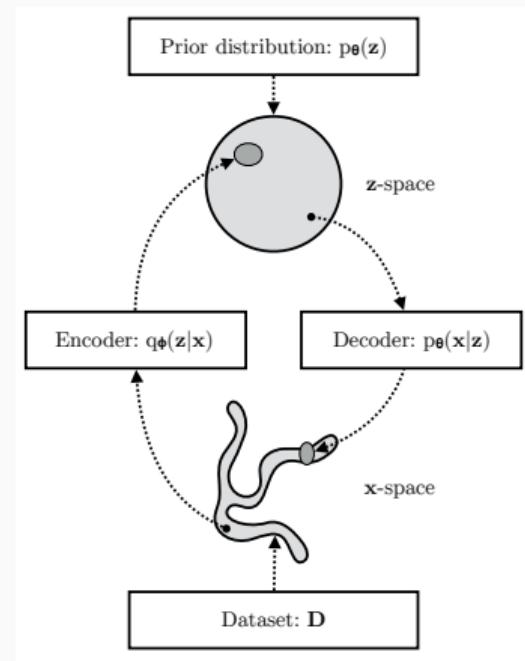
# Towards VAEs: the big picture



- The input space is a generic manifold.
- For the prior we usually use a Gaussian  $\mathbf{z} \sim \mathcal{N}(0, I)$

<sup>3</sup>Kingma and Welling 2019.

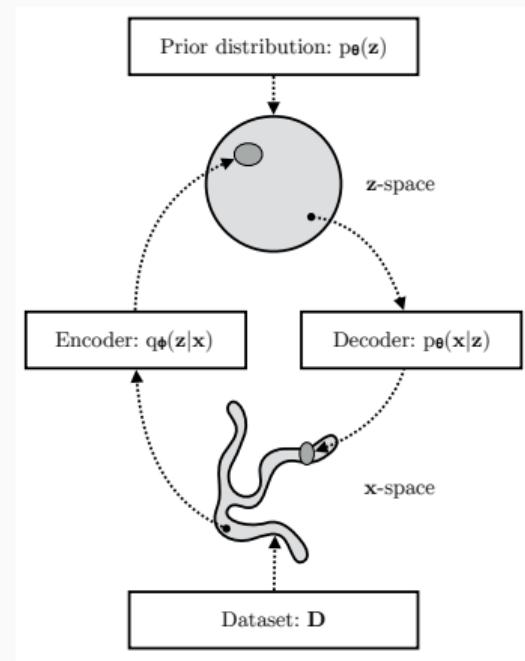
# Towards VAEs: the big picture



- The input space is a generic manifold.
- For the prior we usually use a Gaussian  $\mathbf{z} \sim \mathcal{N}(0, I)$
- Similarly for the approximate posterior:  $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}\left(\mathbf{z}; \mu_\phi(\mathbf{x}), I\sigma_\phi^2(\mathbf{x})\right)$

<sup>3</sup>Kingma and Welling 2019.

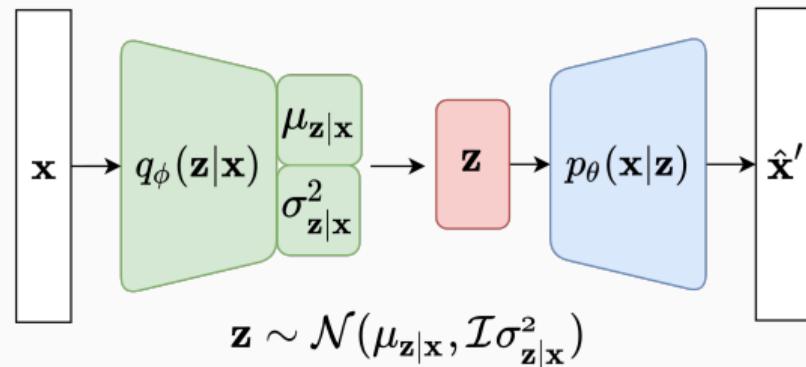
# Towards VAEs: the big picture



- The input space is a generic manifold.
- For the prior we usually use a Gaussian  $\mathbf{z} \sim \mathcal{N}(0, I)$
- Similarly for the approximate posterior:  $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}\left(\mathbf{z}; \mu_\phi(\mathbf{x}), I\sigma_\phi^2(\mathbf{x})\right)$
- The output of the probabilistic decoder depends on the original space (e.g., Bernoulli for binary outputs)

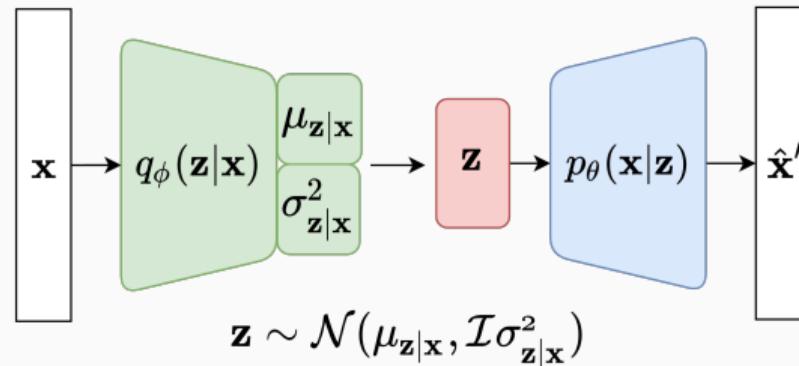
<sup>3</sup>Kingma and Welling 2019.

## A possible implementation



**Figure 2:** A Variational Autoencoder

## A possible implementation



**Figure 2:** A Variational Autoencoder

Let's take a step back and consider the approximate posterior  $q_\phi(\mathbf{z}|\mathbf{x})$

## Variational Inference (I)

Variational Inference<sup>4</sup> tackles the problem of approximating an intractable posterior from an optimization perspective.

$$q_{\phi^*}(z|x) = \operatorname{argmin}_{\phi} \mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z|x))$$

The KL-divergence  $\mathcal{D}_{KL}(\cdot \parallel \cdot)$  is intuitively a measure of *dissimilarity* between probability distributions:

$$\mathcal{D}_{KL}(p(x) \parallel g(x)) = \int p(x) \log \frac{p(x)}{g(x)} dx$$

---

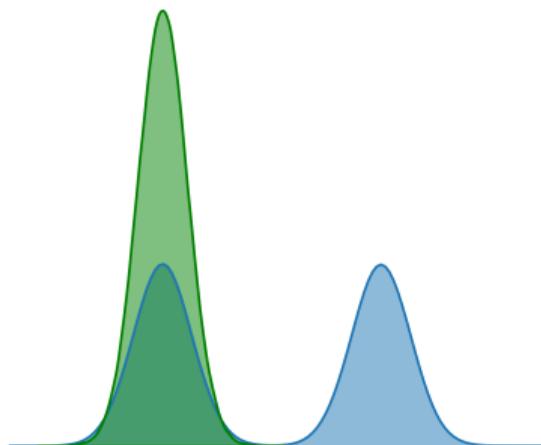
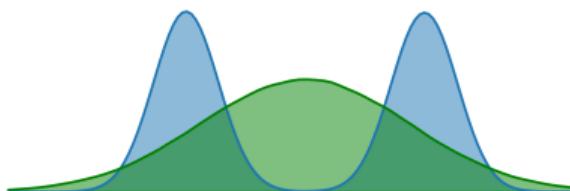
<sup>4</sup>Blei, Kucukelbir, and McAuliffe 2017.

## Remark

Which approximation would you prefer? In which case  $\mathcal{D}_{KL}(q_\phi(z|x) \parallel p_\theta(z|x))$  is lower?

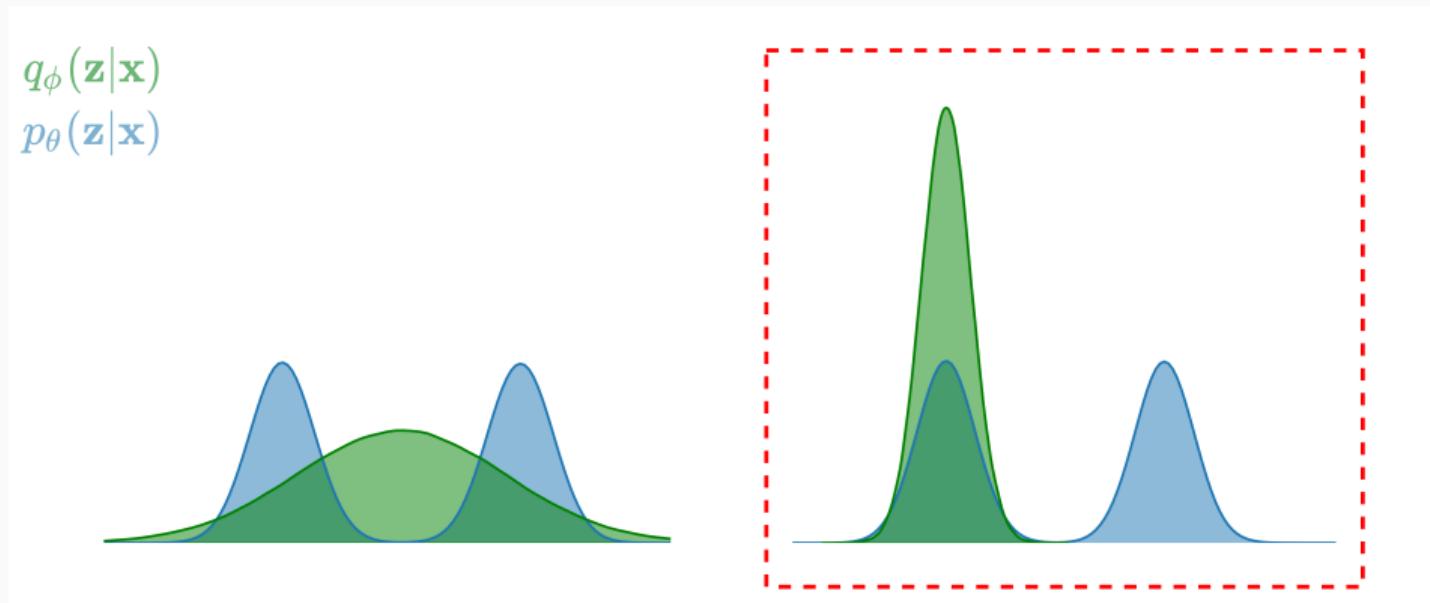
$$q_\phi(z|x)$$

$$p_\theta(z|x)$$



## Remark

Which approximation would you prefer? In which case  $\mathcal{D}_{KL}(q_\phi(z|x) \parallel p_\theta(z|x))$  is lower?



**NB:** The KL-divergence is not symmetric.

## Variational Inference (II)

What optimization objective should we use?

$$\mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z|x))$$

## Variational Inference (II)

What optimization objective should we use?

$$\begin{aligned} \mathcal{D}_{KL}(q_{\phi}(z|x) &\parallel p_{\theta}(z|x)) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} dz \end{aligned} \quad (\text{def.})$$

## Variational Inference (II)

What optimization objective should we use?

$$\mathcal{D}_{KL}(q_\phi(z|x) \parallel p_\theta(z|x))$$

$$= \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z|x)} dz \quad (def.)$$

$$= \int q_\phi(z|x) \log \frac{q_\phi(z|x)p_\theta(x)}{p_\theta(x, z)} dz \quad (\text{product rule})$$

## Variational Inference (II)

What optimization objective should we use?

$$\begin{aligned} & \mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z|x)) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} dz && (def.) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)p_{\theta}(x)}{p_{\theta}(x, z)} dz && (product\ rule) \\ &= \int q_{\phi}(z|x) \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(x, z)} + \log p_{\theta}(x) \right) dz \end{aligned}$$

## Variational Inference (II)

What optimization objective should we use?

$$\begin{aligned} & \mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z|x)) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} dz && (def.) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)p_{\theta}(x)}{p_{\theta}(x,z)} dz && (product\ rule) \\ &= \int q_{\phi}(z|x) \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(x,z)} + \log p_{\theta}(x) \right) dz \\ &= \log p_{\theta}(x) + \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(x,z)} dz && (\int q_{\phi}(z|x) dz = 1) \end{aligned}$$

## Variational Inference (II)

What optimization objective should we use?

$$\begin{aligned} & \mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z|x)) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} dz && (def.) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)p_{\theta}(x)}{p_{\theta}(x,z)} dz && (product\ rule) \\ &= \int q_{\phi}(z|x) \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(x,z)} + \log p_{\theta}(x) \right) dz \\ &= \log p_{\theta}(x) + \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(x,z)} dz && (\int q_{\phi}(z|x) dz = 1) \\ &= \log p_{\theta}(x) + \int q_{\phi}(z|x) \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(z)} - \log p_{\theta}(x|z) \right) dz && (product\ rule) \end{aligned}$$

## Variational Inference (II)

What optimization objective should we use?

$$\begin{aligned} & \mathcal{D}_{KL}(q_\phi(z|x) \parallel p_\theta(z|x)) \\ &= \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z|x)} dz && (def.) \\ &= \int q_\phi(z|x) \log \frac{q_\phi(z|x)p_\theta(x)}{p_\theta(x, z)} dz && (product\ rule) \\ &= \int q_\phi(z|x) \left( \log \frac{q_\phi(z|x)}{p_\theta(x, z)} + \log p_\theta(x) \right) dz \\ &= \log p_\theta(x) + \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(x, z)} dz && (\int q_\phi(z|x) dz = 1) \\ &= \log p_\theta(x) + \int q_\phi(z|x) \left( \log \frac{q_\phi(z|x)}{p_\theta(z)} - \log p_\theta(x|z) \right) dz && (product\ rule) \\ &= \log p_\theta(x) + \mathcal{D}_{KL}(q_\phi(z|x) \parallel p_\theta(z)) - \mathbb{E}_{z \sim q_\phi} [\log p_\theta(x|z)] \end{aligned}$$

## Variational Inference (II)

What optimization objective should we use?

$$\begin{aligned} & \mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z|x)) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} dz && (def.) \\ &= \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)p_{\theta}(x)}{p_{\theta}(x,z)} dz && (product\ rule) \\ &= \int q_{\phi}(z|x) \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(x,z)} + \log p_{\theta}(x) \right) dz \\ &= \log p_{\theta}(x) + \int q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(x,z)} dz && (\int q_{\phi}(z|x) dz = 1) \\ &= \log p_{\theta}(x) + \int q_{\phi}(z|x) \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(z)} - \log p_{\theta}(x|z) \right) dz && (product\ rule) \\ &= \log p_{\theta}(x) + \mathcal{D}_{KL}(q_{\phi}(z|x) \parallel p_{\theta}(z)) - \mathbb{E}_{z \sim q_{\phi}} [\log p_{\theta}(x|z)] \end{aligned}$$

## Variational Inference (III)

Rearranging the terms we get:

$$\underbrace{\log p_{\theta}(x)}_{\text{evidence}} - \underbrace{\mathcal{D}_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x))}_{\text{KL-divergence}} \\ = \mathbb{E}_{z \sim q_{\phi}} [\log p_{\theta}(x|z)] - \mathcal{D}_{KL}(q_{\phi}(z|x) || p_{\theta}(z))$$

## Variational Inference (III)

Rearranging the terms we get:

$$\begin{aligned}\text{ELBO} &= \underbrace{\log p_{\theta}(x)}_{\text{evidence}} - \underbrace{\mathcal{D}_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x))}_{\text{KL-divergence}} \\ &= \mathbb{E}_{z \sim q_{\phi}} [\log p_{\theta}(x|z)] - \mathcal{D}_{KL}(q_{\phi}(z|x) || p_{\theta}(z)) \leq \log p_{\theta}(x)\end{aligned}$$

## Variational Inference (III)

Rearranging the terms we get:

$$\begin{aligned}\underbrace{\text{ELBO}}_{\text{Evidence Lower BOund}} &= \underbrace{\log p_\theta(x)}_{\text{evidence}} - \underbrace{\mathcal{D}_{KL}(q_\phi(z|x) || p_\theta(z|x))}_{\text{KL-divergence}} \\ &= \mathbb{E}_{z \sim q_\phi} [\log p_\theta(x|z)] - \mathcal{D}_{KL}(q_\phi(z|x) || p_\theta(z)) \leq \log p_\theta(x)\end{aligned}$$

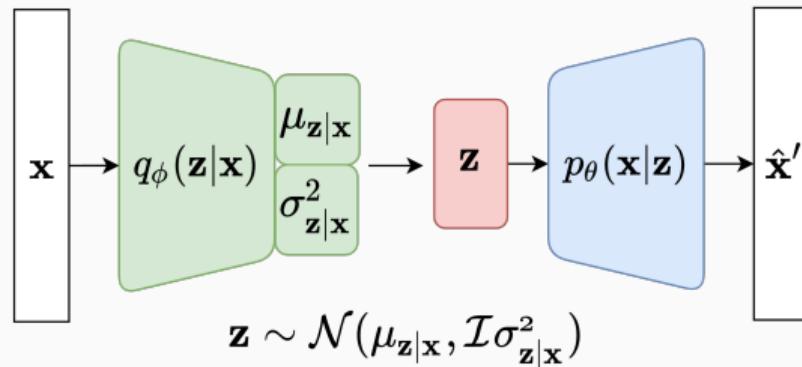
Intuitively, by maximizing the ELBO we maximize the probability of generating **plausible samples** while pushing the approximated posterior closer to the actual one.

## Variational Inference (IV)

$$\text{ELBO} = \underbrace{\mathbb{E}_{z \sim q_\phi} [\log p_\theta(x|z)] - \mathcal{D}_{KL}(q_\phi(z|x) \parallel p_\theta(z))}_{\mathcal{L}(x;\theta,\phi)}$$

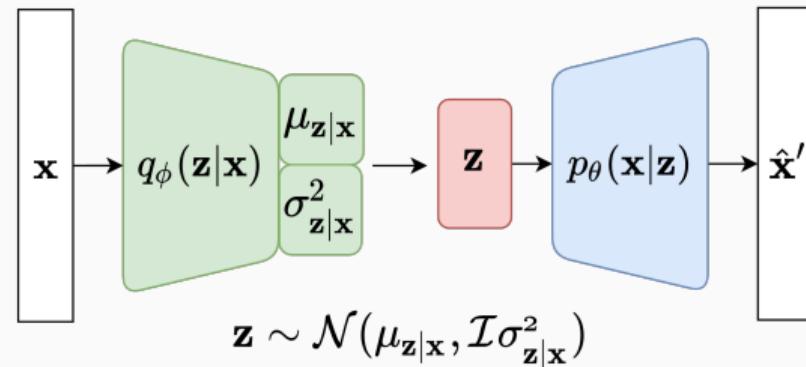
This expression is pretty easy to evaluate empirically by sampling the available data and the approximated posterior, so we are done... are we?

## Reparameterization trick (I)



**Problem:** Sampling from a distribution is not a differentiable operation!

## Reparameterization trick (I)



Problem: Sampling from a distribution is not a differentiable operation!

Solution: The reparameterization trick

## Reparameterization trick (II)

In general the problem requires to estimate the **gradient of an expectation**:

$$\nabla_{\theta} \mathbb{E}_{z \sim q_{\theta}} [f(z)]$$

With the reparameterization trick we *change the sampling distribution* so that it becomes independent from  $\theta$ :

$$\varepsilon \sim p(\varepsilon)$$

$$z = g(x, \varepsilon)$$

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_{z \sim q_{\theta}} [f(z)] &= \nabla_{\theta} \mathbb{E}_{\varepsilon \sim p(\varepsilon)} [f(g(x, \varepsilon))] \\ &= \mathbb{E}_{\varepsilon \sim p(\varepsilon)} [\nabla_{\theta} f(g(x, \varepsilon))]\end{aligned}$$

## Reparameterization trick (II)

In general the problem requires to estimate the **gradient of an expectation**:

$$\nabla_{\theta} \mathbb{E}_{z \sim q_{\theta}} [f(z)]$$

With the reparameterization trick we *change the sampling distribution* so that it becomes independent from  $\theta$ :

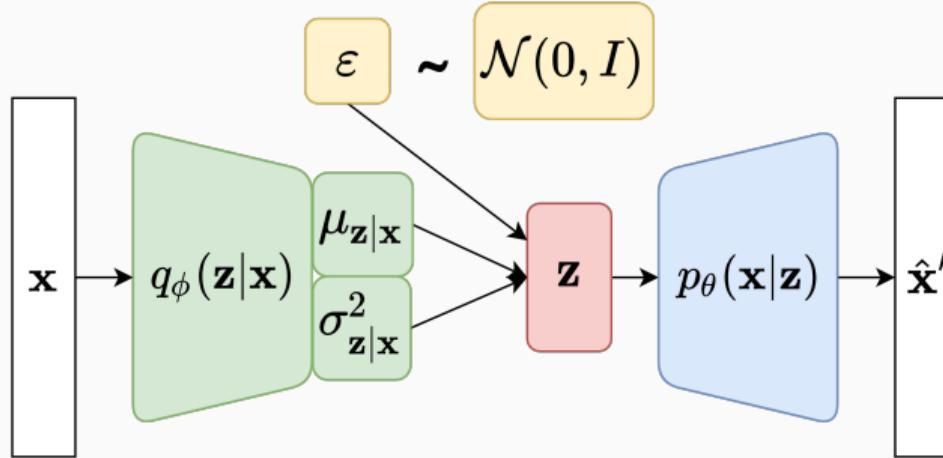
$$\varepsilon \sim p(\varepsilon)$$

$$z = g(x, \varepsilon)$$

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_{z \sim q_{\theta}} [f(z)] &= \nabla_{\theta} \mathbb{E}_{\varepsilon \sim p(\varepsilon)} [f(g(x, \varepsilon))] \\ &= \mathbb{E}_{\varepsilon \sim p(\varepsilon)} [\nabla_{\theta} f(g(x, \varepsilon))]\end{aligned}$$

Other alternatives exist (e.g., see the **REINFORCE** estimator from the RL literature).

## Reparameterization trick (III)



$$\varepsilon \sim \mathcal{N}(0, I)$$

$$\mu_{\mathbf{z}|\mathbf{x}}, \sigma_{\mathbf{z}|\mathbf{x}}^2 = \text{Encoder}(\mathbf{x})$$

$$\mathbf{z} = \mu_{\mathbf{z}|\mathbf{x}} + \varepsilon \odot \sigma_{\mathbf{z}|\mathbf{x}}$$

## Reparameterization trick (IV)

In the Gaussian case, assuming a  $J$ -dimensional latent space:

$$\begin{aligned}-\mathcal{D}_{KL}(q_{\phi}(\mathbf{z}|\mathbf{x}) \parallel p_{\theta}(\mathbf{z})) &= \int q_{\phi}(\mathbf{z}|\mathbf{x}) (\log p_{\theta}(\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x})) d\mathbf{z} \\&= \frac{1}{2} \sum_{j=1}^J \left( 1 + \log((\sigma_{\mathbf{z}|\mathbf{x}}^{(j)})^2) - (\mu_{\mathbf{z}|\mathbf{x}}^{(j)})^2 - (\sigma_{\mathbf{z}|\mathbf{x}}^{(j)})^2 \right).\end{aligned}$$

Considering the  $i$ -th available sample  $\mathbf{x}^{(i)}$  and  $L$  samples from the approximate posterior:

$$\begin{aligned}\tilde{\mathcal{L}}(\mathbf{x}^{(i)}; \boldsymbol{\theta}, \boldsymbol{\phi}) &= \frac{1}{2} \sum_{j=1}^J \left( 1 + \log((\sigma_{\mathbf{z}|\mathbf{x}^{(i)}}^{(j)})^2) - (\mu_{\mathbf{z}|\mathbf{x}^{(i)}}^{(j)})^2 - (\sigma_{\mathbf{z}|\mathbf{x}^{(i)}}^{(j)})^2 \right) + \frac{1}{L} \sum_{l=1}^L \log p_{\theta}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}),\end{aligned}$$

where  $\mathbf{z}^{(i,l)} = \mu_{\mathbf{z}|\mathbf{x}^{(i)}} + \boldsymbol{\varepsilon}^{(l)} \odot \boldsymbol{\sigma}_{\mathbf{z}|\mathbf{x}^{(i)}}$ . The final optimization objective is:

$$\tilde{\mathcal{L}}_N(\boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{i=1}^N \tilde{\mathcal{L}}(\mathbf{x}^{(i)}; \boldsymbol{\theta}, \boldsymbol{\phi})$$

## Examples of learned manifolds



<sup>5</sup>Kingma and Welling 2013.

## State of the art



<sup>6</sup>Razavi, Oord, and Vinyals 2019.

## Demo

---

## Demo



<https://cutt.ly/pfSOTKv>

**Questions?**

## Likelihood for a Bernoulli decoder

In the case where we model the output of the probabilistic decoder  $p_{\theta}(\mathbf{x}|\mathbf{z})$  as multivariate Bernoulli the log-likelihood can be compute as:

$$\log p_{\theta}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}) = \underbrace{\sum_{i=1}^N \mathbf{x}^{(i)} \log \hat{\mathbf{x}}^{(i)} + (1 - \mathbf{x}^{(i)}) \log (1 - \hat{\mathbf{x}}^{(i)})}_{\text{binary cross-entropy}}$$

where  $\hat{\mathbf{x}}^{(i)} = \text{Decoder}(\mathbf{z}^{(i,l)})$ .

## Likelihood for a Gaussian decoder

In the case of a multivariate diagonal Gaussian decoder:

$$\log p_{\theta}(\mathbf{x}^{(i)} | \mathbf{z}^{(i,l)}) = \log \mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{z}^{(i,l)}}, \boldsymbol{\sigma}_{\mathbf{x}|\mathbf{z}^{(i,l)}}^2)$$

where  $\boldsymbol{\mu}_{\mathbf{x}|\mathbf{z}^{(i,l)}}, \boldsymbol{\sigma}_{\mathbf{x}|\mathbf{z}^{(i,l)}} = \text{Decoder}(\mathbf{z}^{(i,l)})$ .

Note that:

- The output of the Decoder network is the mean and the variance of the Gaussian.
- No need of tricks to estimate the gradient since the expectation is over  $\mathbf{z}$  and not  $\mathbf{x}$ .

## Credits

Most of the material was inspired and adapted from:

- The tutorial on VAEs from the authors of the original paper:  
<https://arxiv.org/abs/1906.02691>
- This nice blogpost by Lilian Weng: <https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html>

## References i

-  Blei, David M., Alp Kucukelbir, and Jon D. McAuliffe (Apr. 2017). "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518, pp. 859–877. ISSN: 1537-274X. DOI: 10.1080/01621459.2017.1285773. URL: <http://dx.doi.org/10.1080/01621459.2017.1285773>.
-  Kingma, Diederik P. and Max Welling (2013). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114 [stat.ML].
-  – (2019). "An Introduction to Variational Autoencoders". In: *Foundations and Trends® in Machine Learning* 12.4, pp. 307–392. ISSN: 1935-8245. DOI: 10.1561/2200000056. URL: <http://dx.doi.org/10.1561/2200000056>.
-  Razavi, Ali, Aaron van den Oord, and Oriol Vinyals (2019). *Generating Diverse High-Fidelity Images with VQ-VAE-2*. arXiv: 1906.00446 [cs.LG].