PHYS 139/239: Machine Learning in Physics

Lecture 14: More Autoencoders

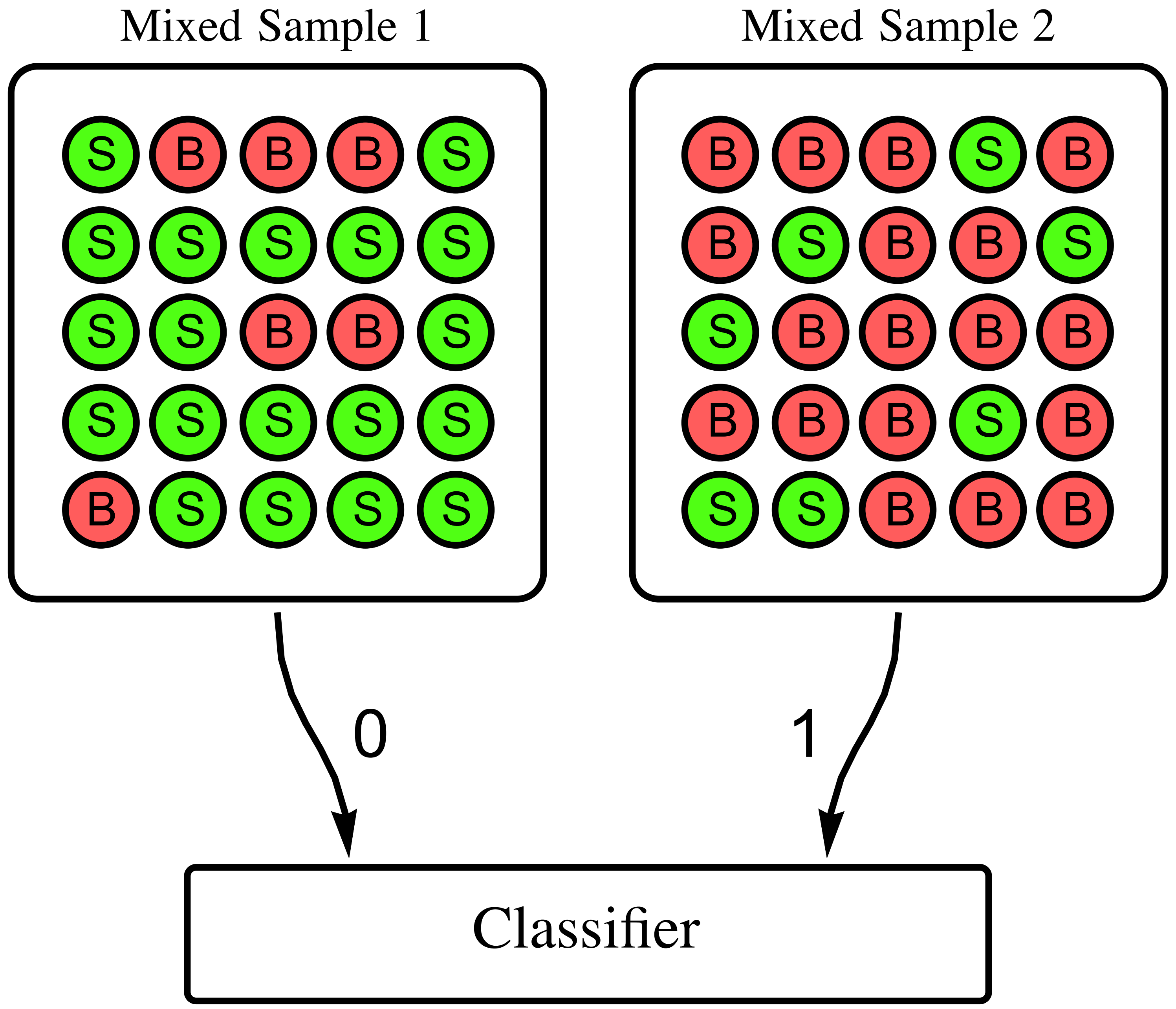
Javier Duarte — February 23, 2023

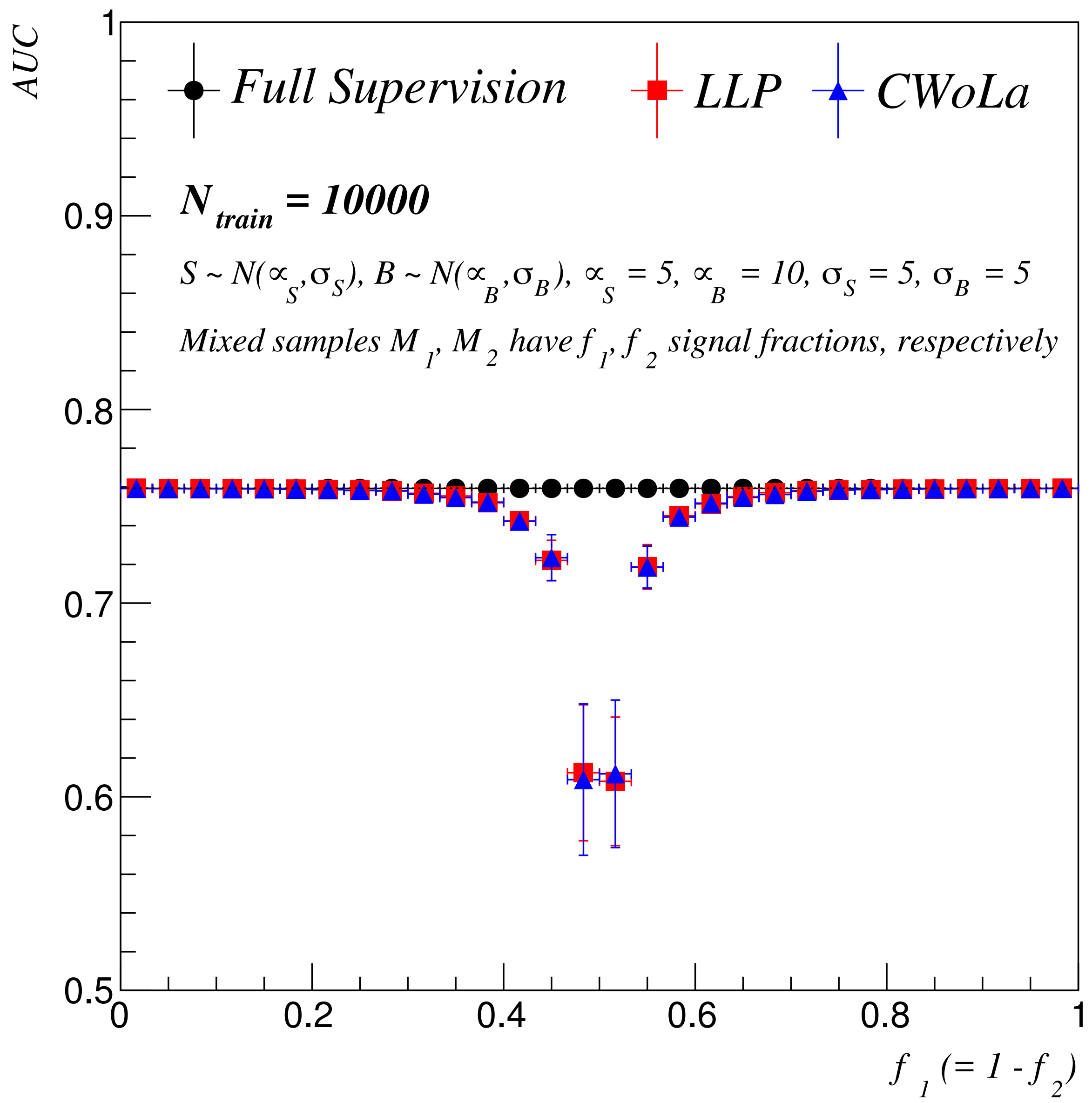
Recap: Types of learning

- Supervised learning: labels known for each data sample
- Unsupervised learning: only features known; no labels!
- Weakly supervised learning: features paired with noisy labels
- Semi-supervised learning: features paired with partial (incomplete) labels
- …

Classification witout labels (CWoLA)

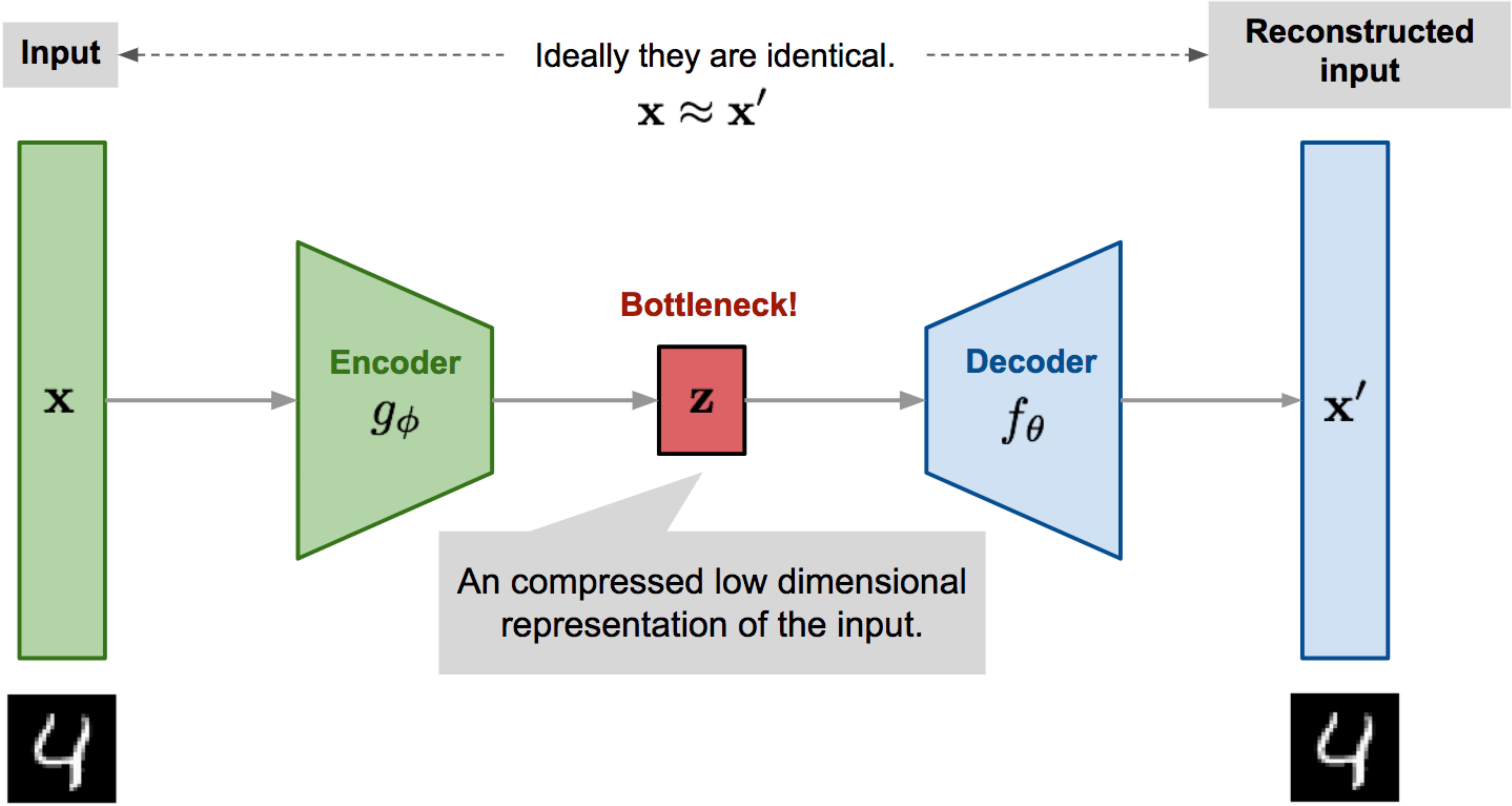
- Classification without labels (CWoLA): https://arxiv.org/abs/1708.02949
- Example of a weakly supervised framework
- Useful when you know you can isolate two samples with different fractions of signal and background
- Quite robust: works even if you don't exactly know the fractions!
- Limitation: performs drops off rapidly near $f_1 = f_2 = 0.5$
  - i.e. when the two samples don't actually have different fractions!





Recap: Autoencoders

- Feedforward neural networks with same input and output shapes
  - Goal: reconstruct original input by minimizing mean-squared error loss (or similar)

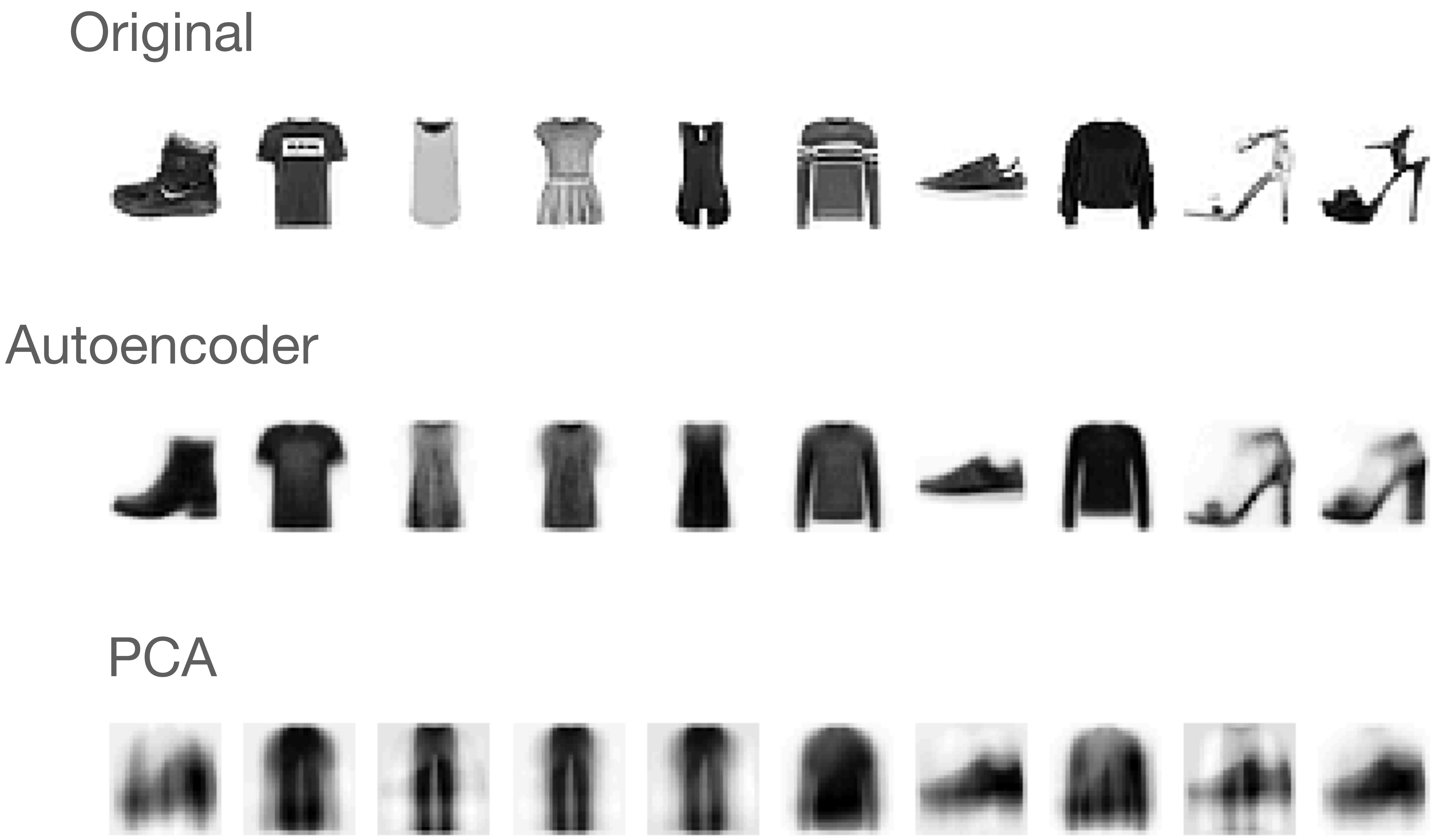


Example: fashion MNIST

https://rittikghosh.com/autoencoder.html

- Autoencoder in Keras:

```
m = Sequential()
m.add(Dense(1000, activation='relu',
            input_shape=(784,)))
m.add(Dense(500, activation='relu'))
m.add(Dense(250, activation='relu'))
m.add(Dense(32, activation='relu'))
m.add(Dense(2, activation='linear',
            name="bottleneck"))
m.add(Dense(32, activation='relu'))
m.add(Dense(250, activation='relu'))
m.add(Dense(500, activation='relu'))
m.add(Dense(1000, activation='relu'))
m.add(Dense(784, activation='sigmoid'))
```



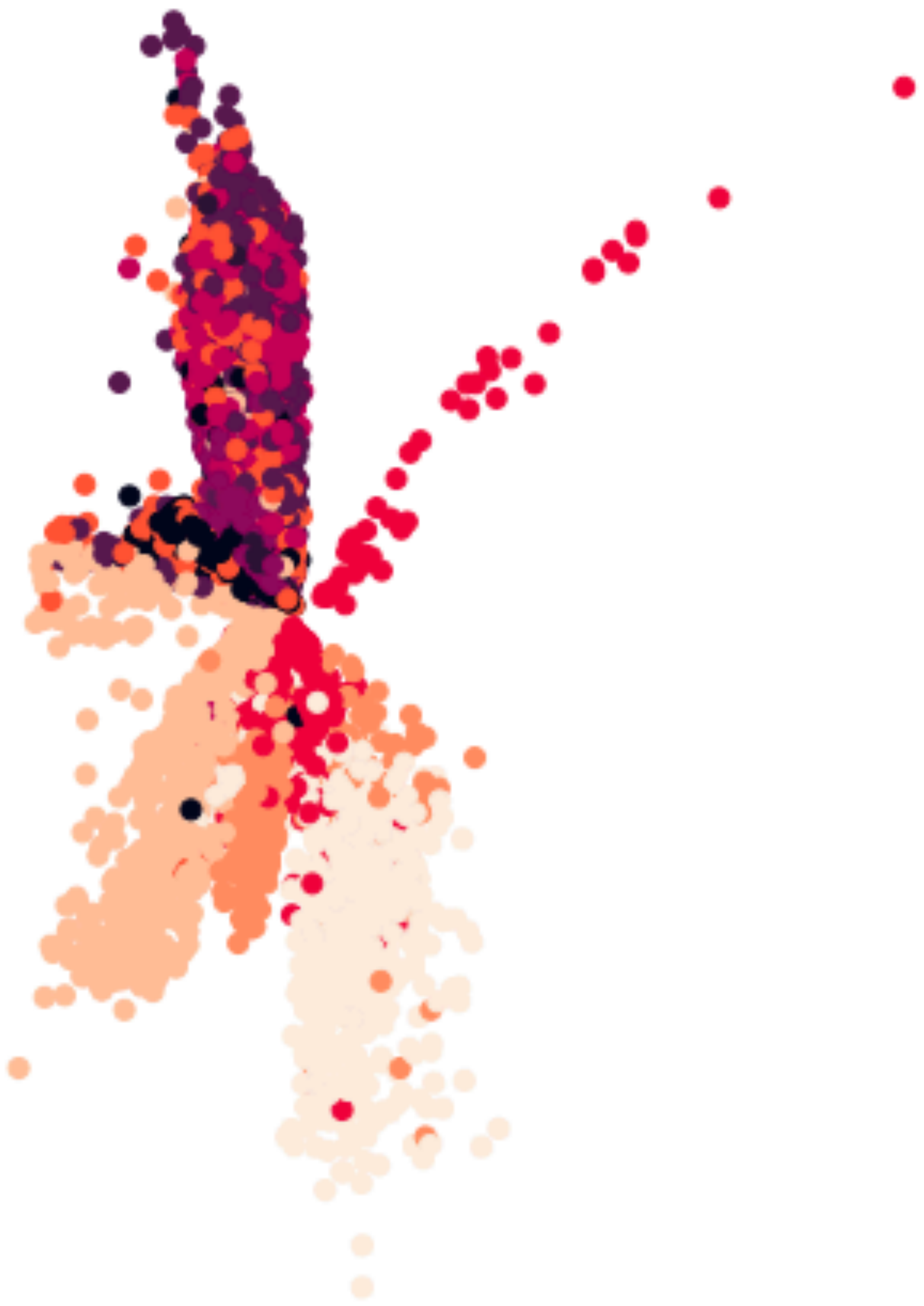
Example: fashion MNIST

https://rittikghosh.com/autoencoder.html

- Autoencoder in Keras:

```
m = Sequential()
m.add(Dense(1000, activation='relu',
            input_shape=(784,)))
m.add(Dense(500, activation='relu'))
m.add(Dense(250, activation='relu'))
m.add(Dense(32, activation='relu'))
m.add(Dense(2, activation='linear',
            name="bottleneck"))
m.add(Dense(32, activation='relu'))
m.add(Dense(250, activation='relu'))
m.add(Dense(500, activation='relu'))
m.add(Dense(1000, activation='relu'))
m.add(Dense(784, activation='sigmoid'))
```

Autoencoder

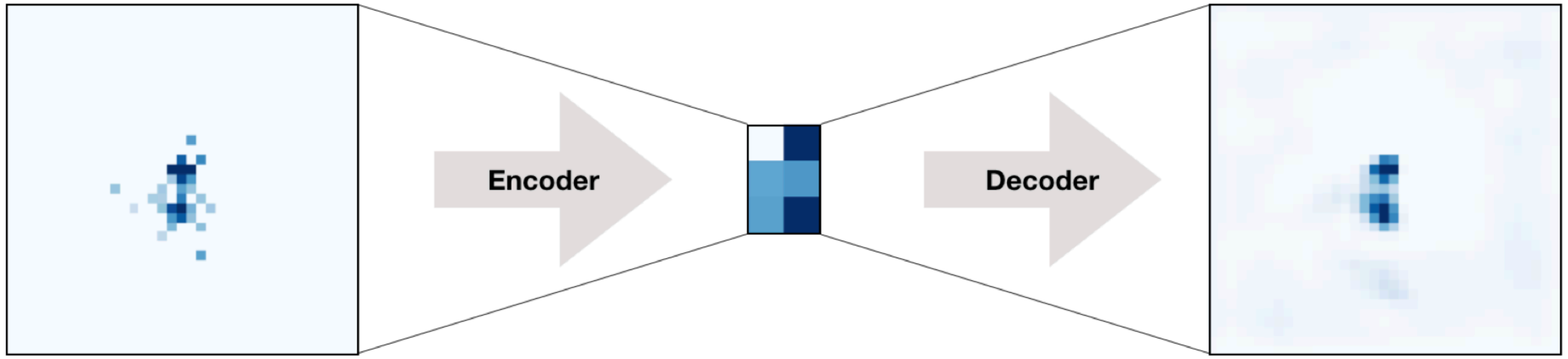


PCA

Application: outlier (anomaly) detection

Credit: B. Nachman
https://indico.cern.ch/event/1188153/

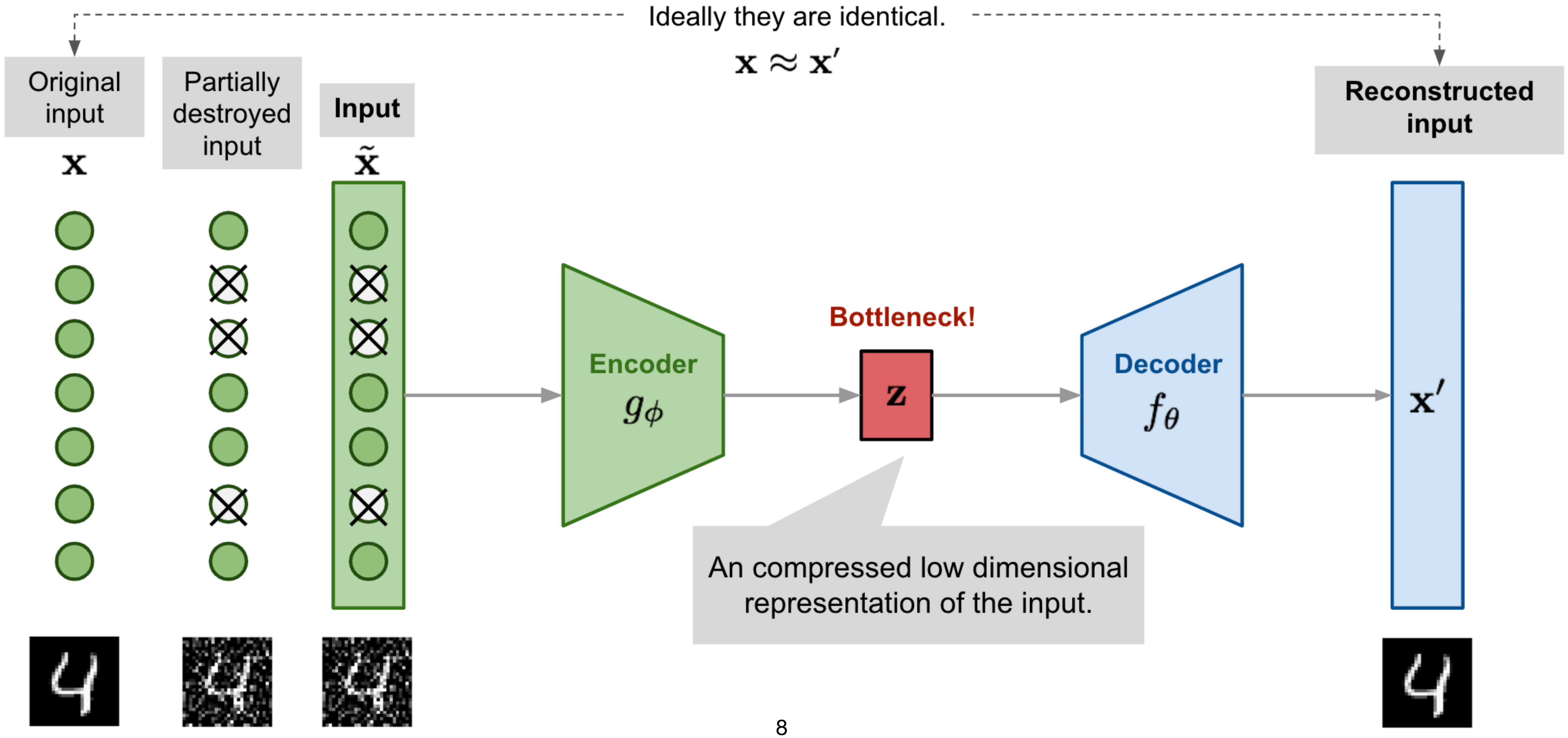
- Autoencoders compress data and then uncompress it
  - Autoencoder is trained on background (and has good reconstruction performance)
  - If x is far from $\mathrm{AE}(x) = \mathrm{Decoder}(\mathrm{Encoder}(x))$, then x has low $p_{\mathrm{bkgd}}(x)$
- Directly use "reconstruction loss" $L(x, \mathrm{AE}(x))$ as an anomaly score



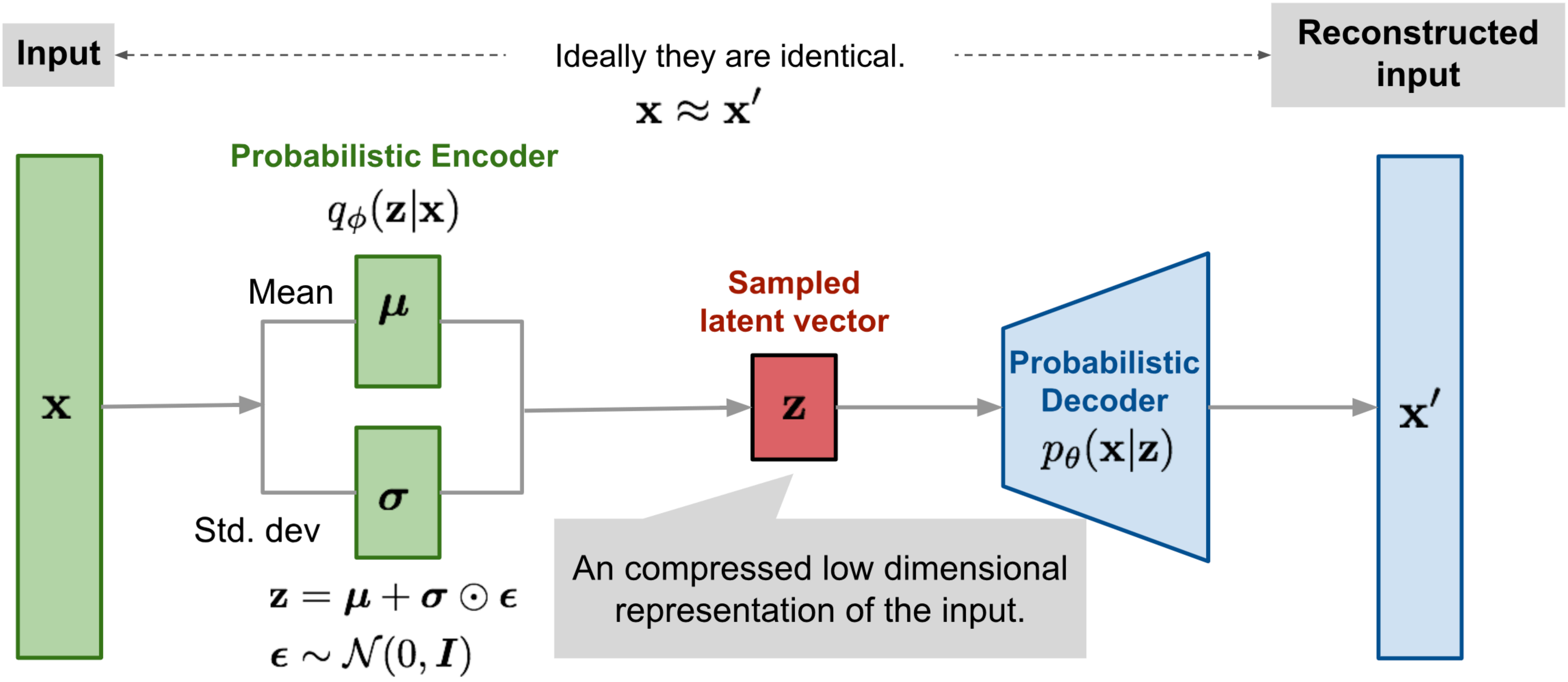
Denoising autoencoers

- We are at risk of “overfitting” (or simply learning the as useful identity function) when the latent space is overcomplete

- Denoising autoencoders (Vincent et al. 2008) corrupt the input by adding noise or masking values stochastically then the model is trained to recover the original input (not the corrupted one)



Variational autoencoders



Variational autoencoders

- Variational autoencoder or VAEs (Kingma & Welling, 2014) rooted in variational Bayesian methods
- Instead of mapping the input into a *fixed* vector, we want to map it into a *distribution* p_θ parameterized by θ
- Input data $\mathbf{x}$ and encoding vector $\mathbf{z}$ are related by
  - Prior $p_\theta(\mathbf{z})$
  - Likelihood $p_\theta(\mathbf{x} \mid \mathbf{z})$
  - Posterior $p_\theta(\mathbf{z} \mid \mathbf{x})$

Variational autoencoders

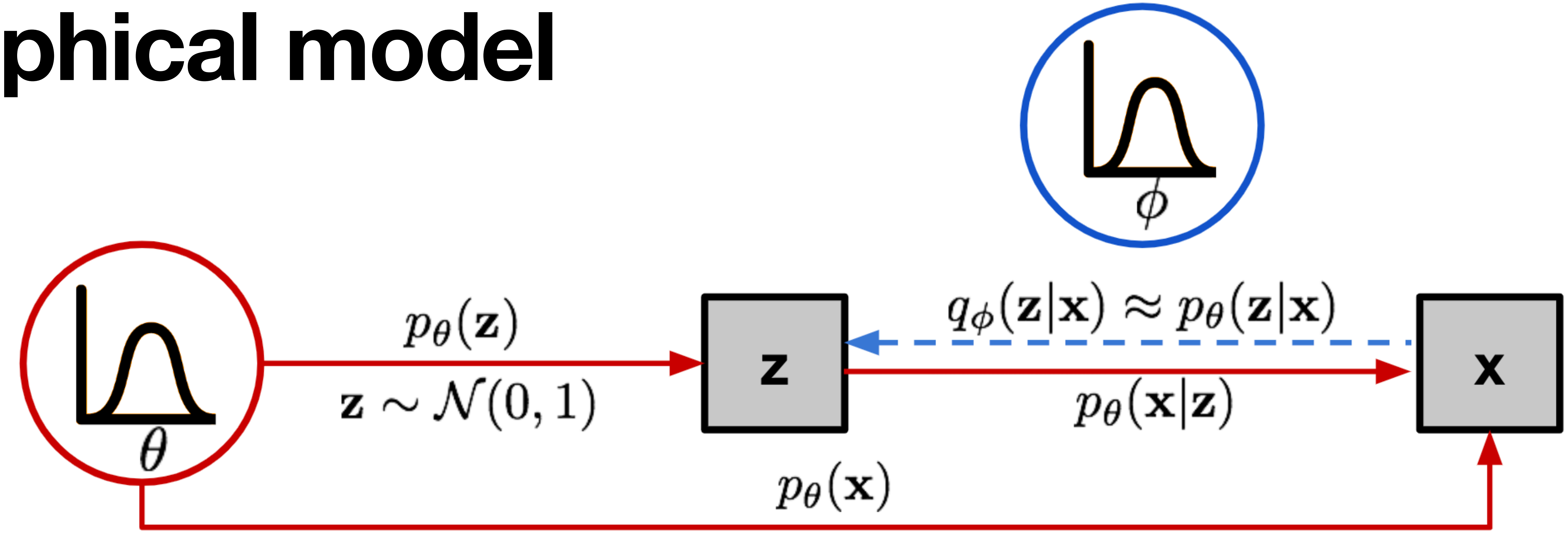
- Given θ, we can generate a sample like a real data point by following
    1. Sample $\mathbf{z}^{(i)}$ from a prior distribution $p_\theta(\mathbf{z})$
    2. Generate a value $\mathbf{x}^{(i)}$ from the conditional distribution $p_\theta(\mathbf{x} \mid \mathbf{z} = \mathbf{z}^{(i)})$
- The optimal parameter $\theta*$ maximizes the probability of generating real data samples or minimizes the loss:

$$\theta* = \arg\min_\theta \left(-\sum_{i=1}^{n} \log p_\theta(\mathbf{x}^{(i)}) \right)$$

where

$$p_\theta(\mathbf{x}^{(i)}) = \int p_\theta(\mathbf{x}^{(i)} \mid \mathbf{z}) p_\theta(\mathbf{z}) d\mathbf{z}$$

Graphical model



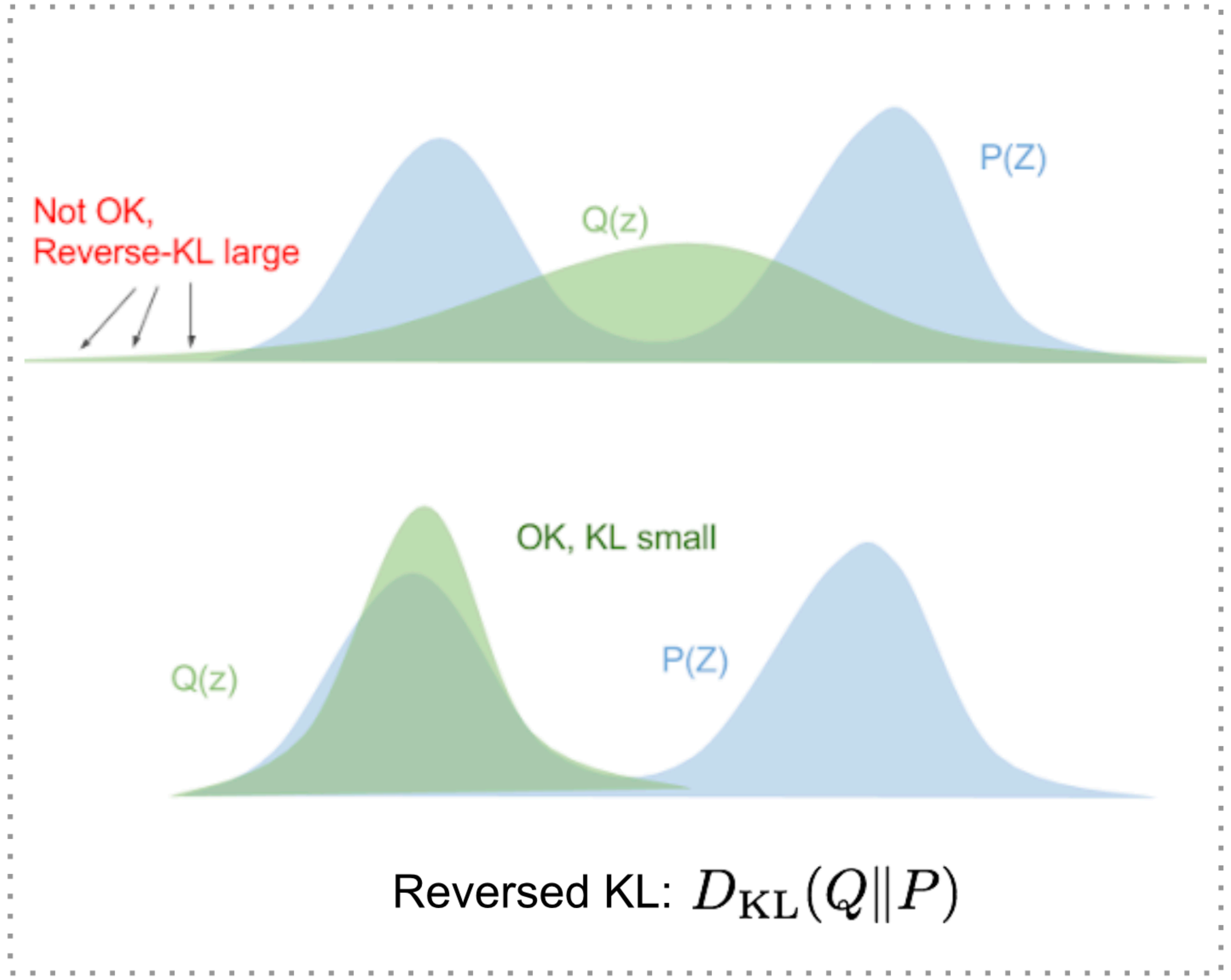
- Not easy to compute $p_\theta(\mathbf{x}^{(i)})$ because of integration over all values of $\mathbf{z}$
- To narrow down the space, introduce *approximation function* to output a likely code given an input $\mathbf{x}$, $q_\phi(\mathbf{z} \mid \mathbf{x})$ parametrized by ϕ
    - Conditional probability $p_\theta(\mathbf{x} \mid \mathbf{z})$ defines a generative model, known as the *probabilistic decoder*
    - Approximation function $q_\phi(\mathbf{z} \mid \mathbf{x})$ is the *probabilistic encoder*

VAE loss function: ELBO

- Estimated posterior $q_\phi(\mathbf{z}\,|\,\mathbf{x})$ should be close to real one $p_\theta(\mathbf{z}\,|\,\mathbf{x})$

- Use Kullback-Leilbler (KL) divergence $D_{\mathrm{KL}}(q_\phi(\mathbf{z}\,|\,\mathbf{x})||p_\theta(\mathbf{z}\,|\,\mathbf{x}))$ to quantify distance

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}\,|\,\mathbf{x})||p_\theta(\mathbf{z}\,|\,\mathbf{x})) = \int q_\phi(\mathbf{z}\,|\,\mathbf{x})\log\frac{q_\phi(\mathbf{z}\,|\,\mathbf{x})}{p_\theta(\mathbf{z}\,|\,\mathbf{x})}d\mathbf{z}$$



- Total loss:

$$L = -\log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}\,|\,\mathbf{x})||p_\theta(\mathbf{z}\,|\,\mathbf{x}))$$

- Known as evidence lower bound (ELBO) because by minimizing the loss, we are maximizing the lower bound of the probability of generating real data samples

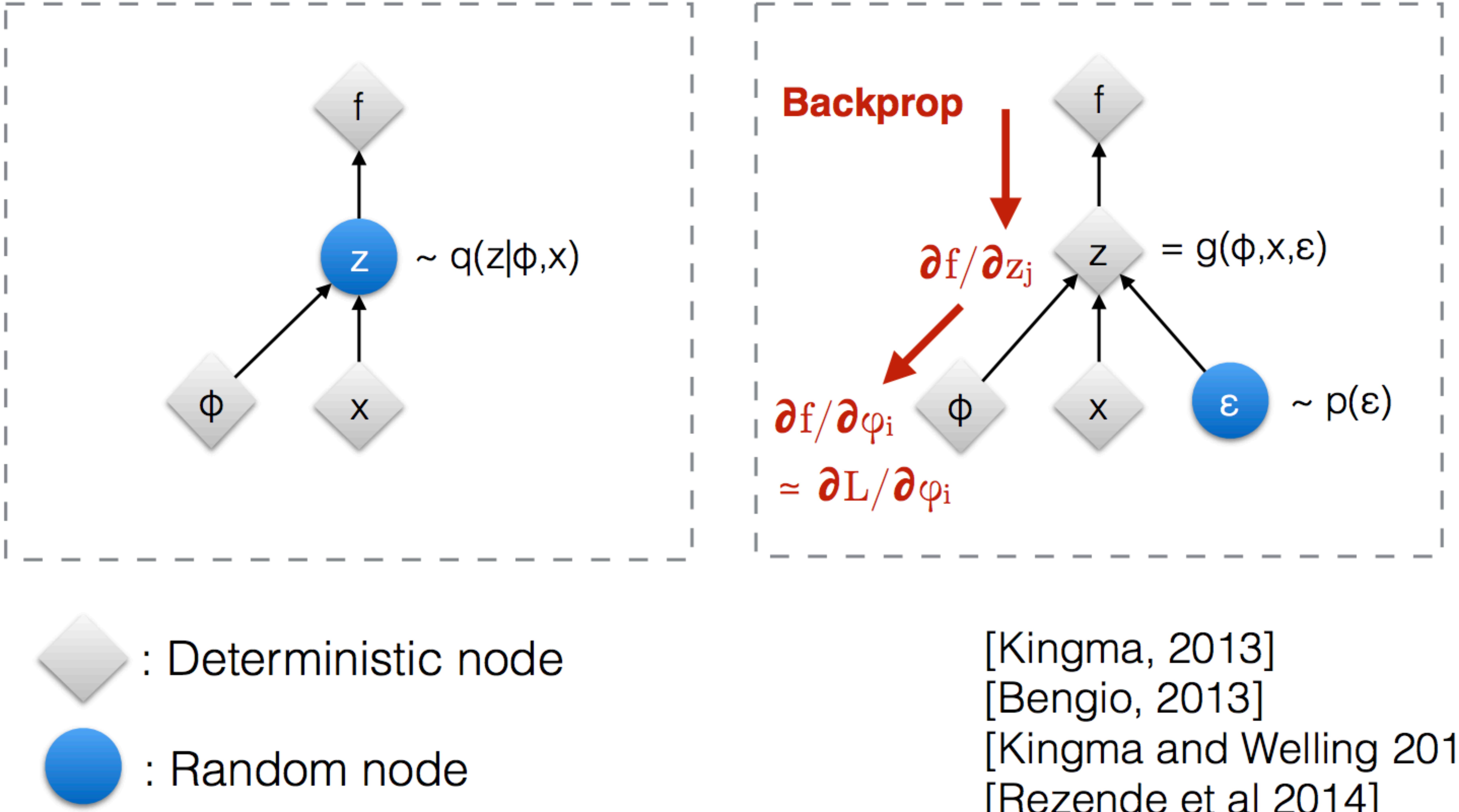
Reparametrization trick

Original form:

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

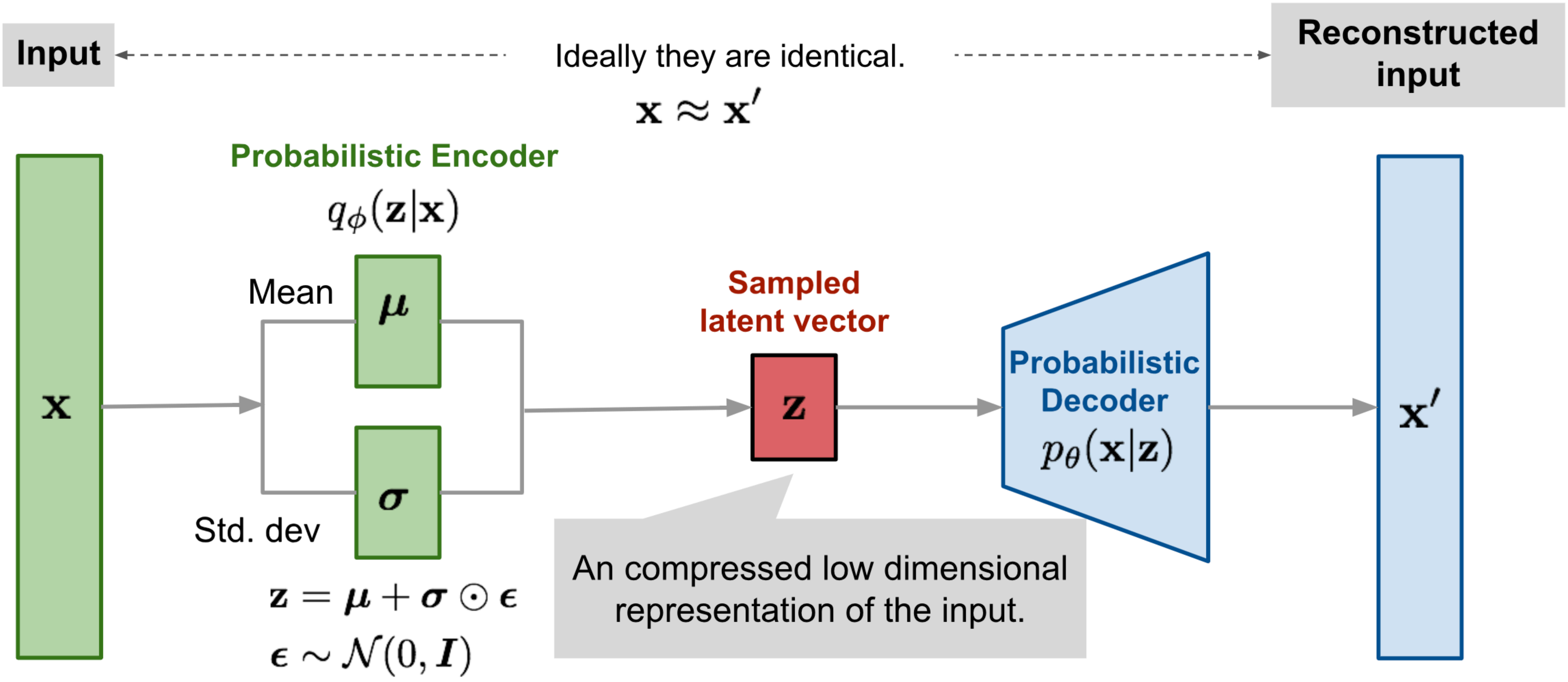
Reparametrized form:

$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0,\boldsymbol{I})$



[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

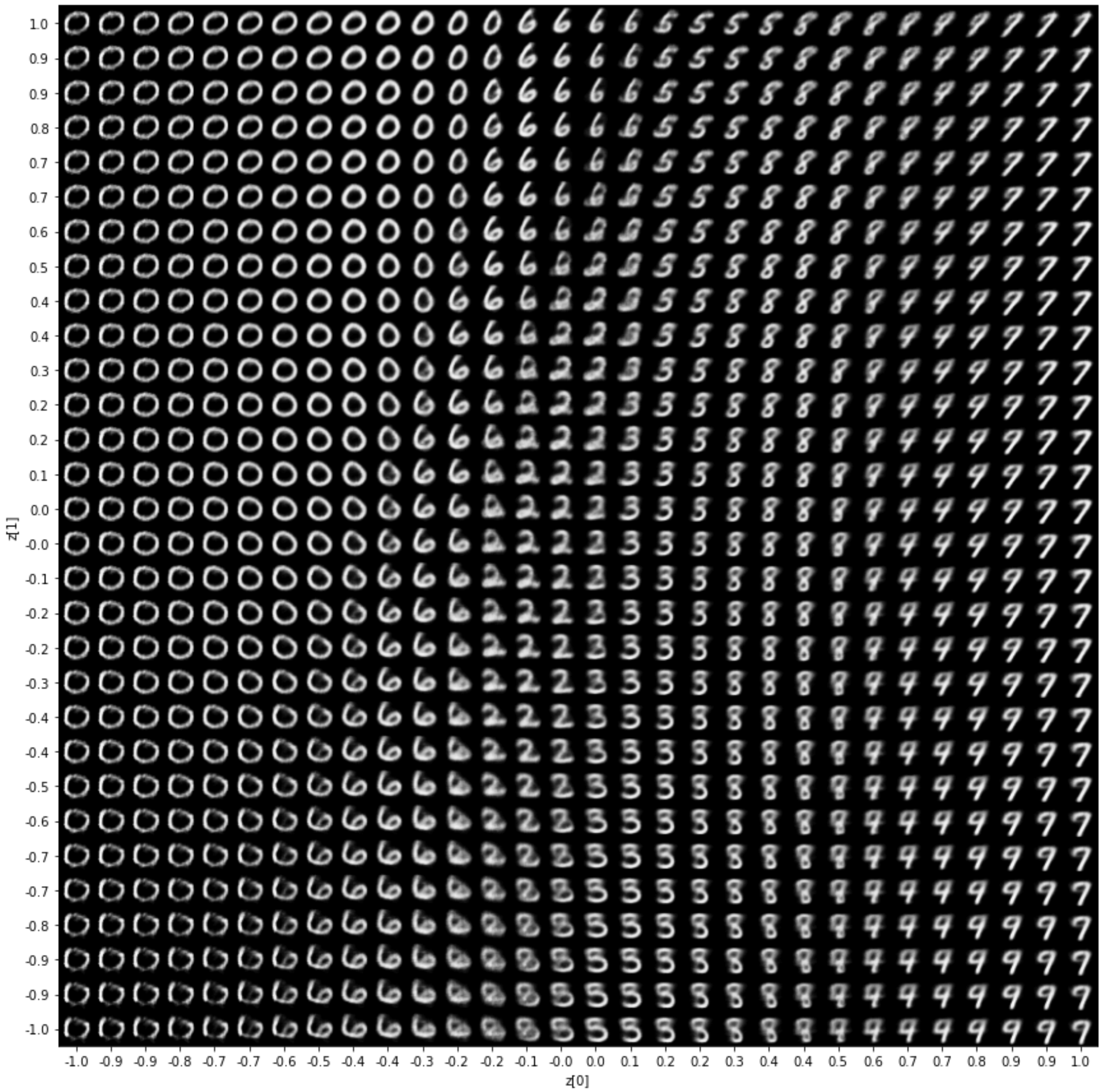
Variational autoencoders

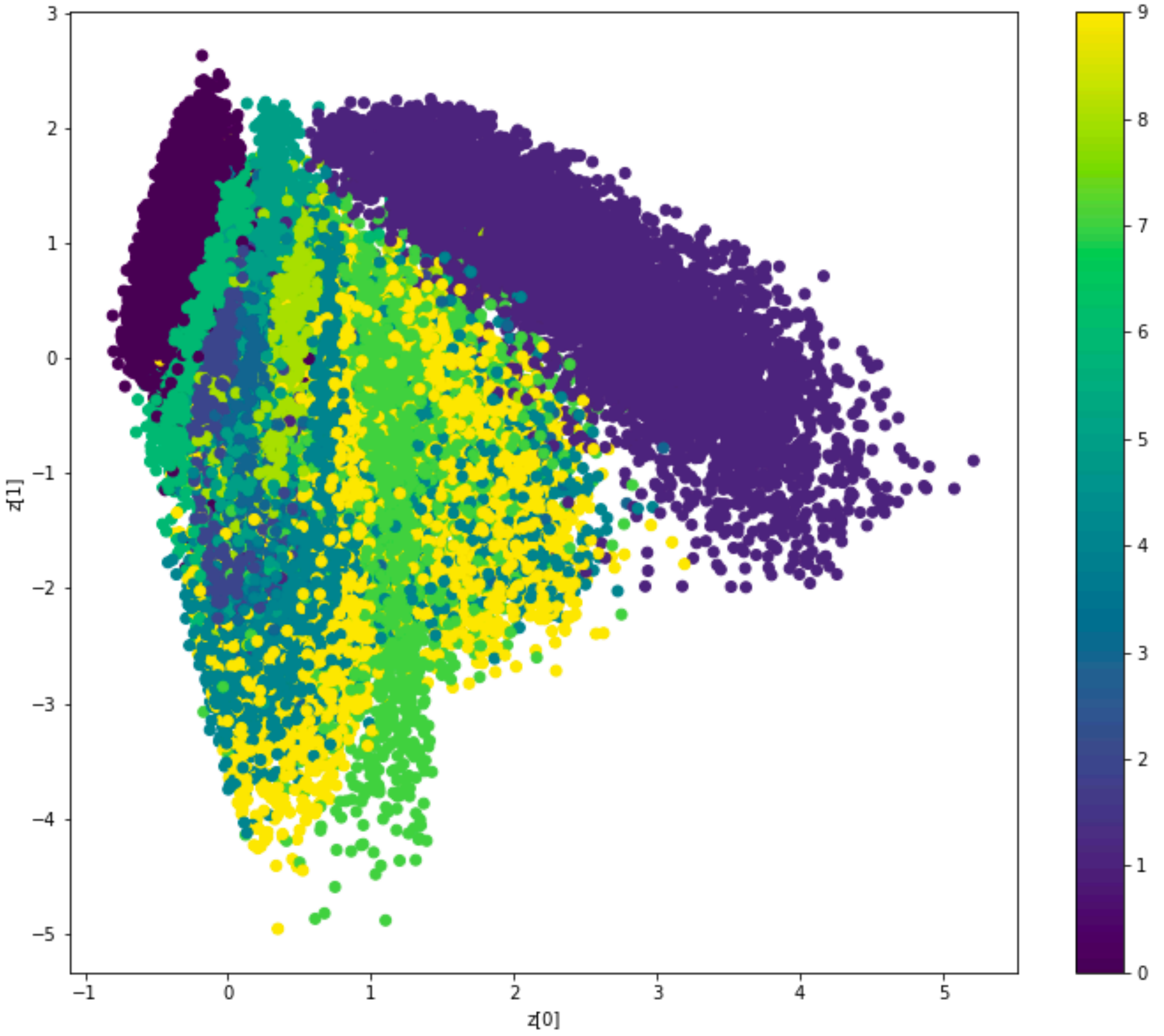


Implementation

- Implementation in Keras for handwritten MNIST digits: https://keras.io/examples/generative/vae/

- Two-dimensional (probabilistic) latent space





Next time

- Model compression