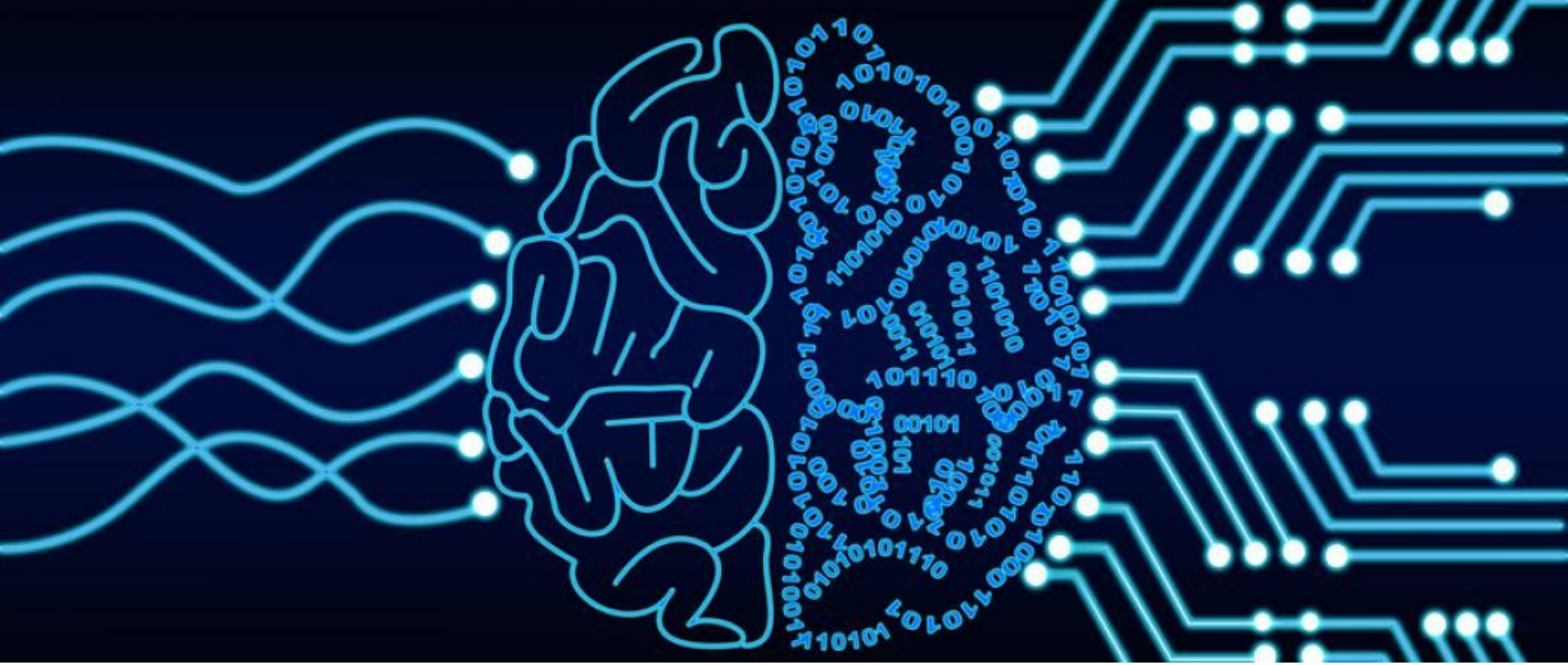


Deep generative methods

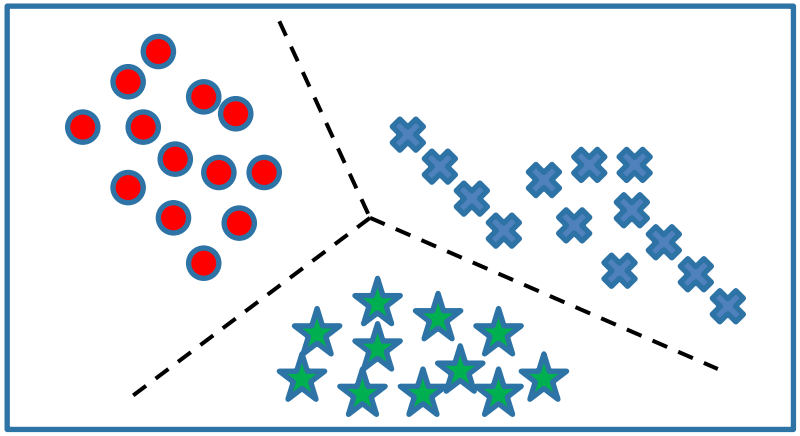
Stéphane Lathuilière
Assistant Professor
Télécom Paris

Introduction: generative networks and applications

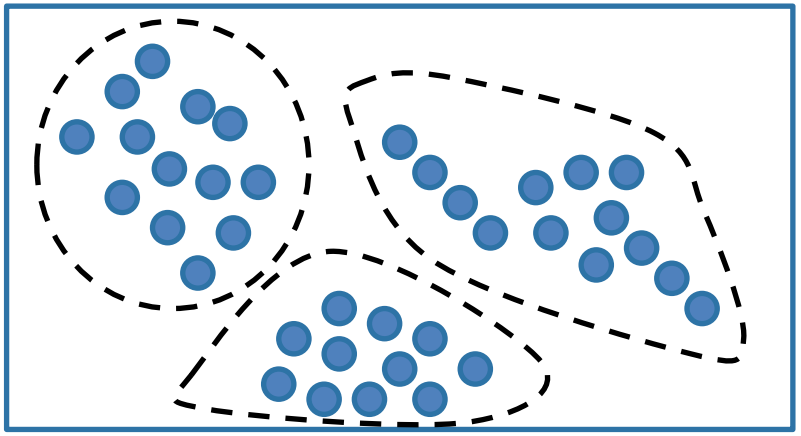
- Generative basics (GAN and VAE) (3h)
- Advanced Generative methods (2h)
- Domain adaptations (1h)
- Labs:
  - GAN
  - VAE

Types of Learning

- **Supervised (inductive)** learning
  - Given: training data + desired outputs (labels)
- **Unsupervised** learning
  - Given: training data (without desired outputs)



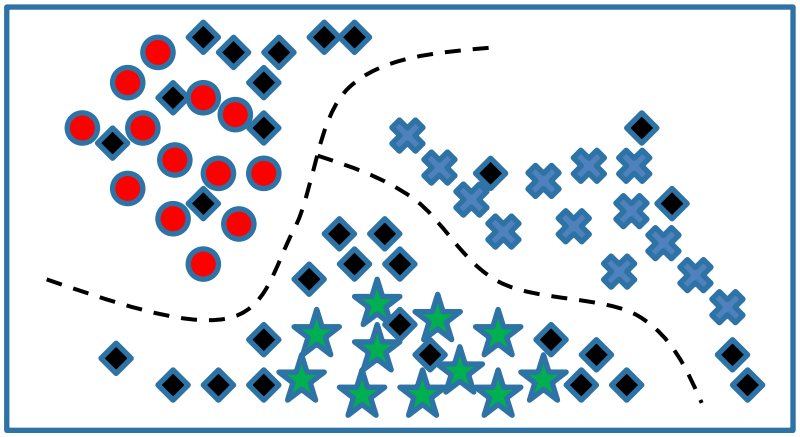
Supervised learning



Unsupervised learning

Types of Learning

- **Supervised (inductive)** learning
  - Given: training data + desired outputs (labels)
- **Unsupervised** learning
  - Given: training data (without desired outputs)
- **Semi-supervised** learning
  - Given: training data + a few desired outputs



Semi-supervised learning

Types of Learning

- **Supervised (inductive)** learning
  - Given: training data + desired outputs (labels)
- **Unsupervised** learning
  - Given: training data (withoutdesired outputs)
- **Semi-supervised** learning
  - Given: training data + a few desired outputs
- **Reinforcement** learning
  - Rewards from sequence of actions

Supervised vs Unsupervised Learning

Supervised Learning

Data: (x, y) x is data, y is label

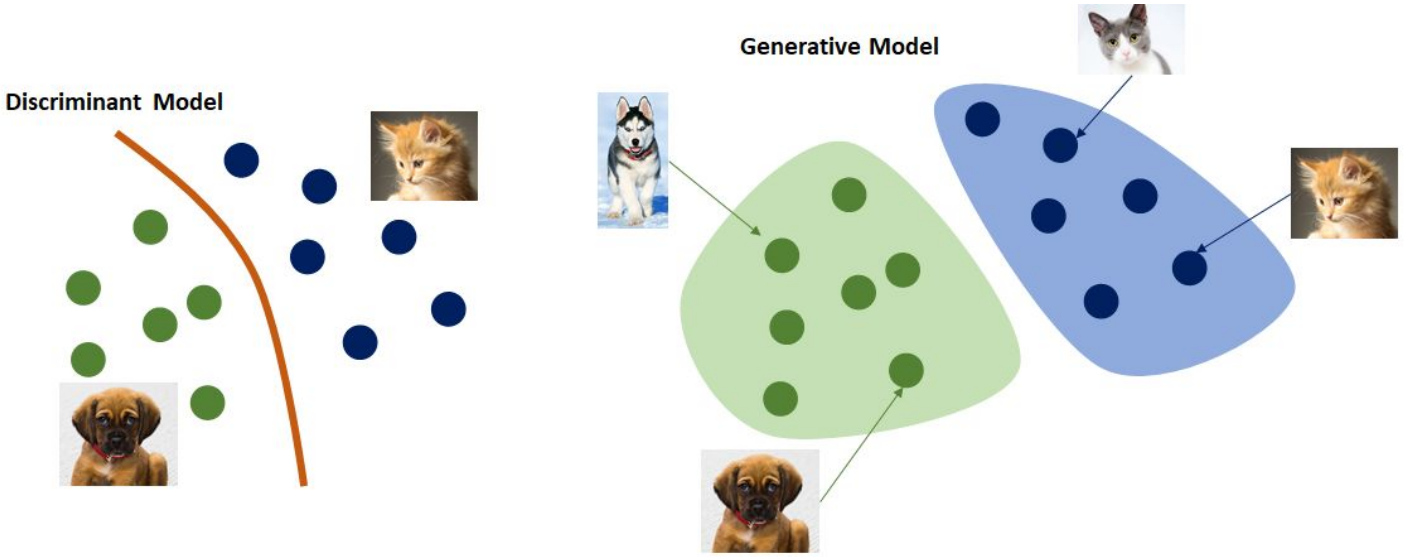
Goal: Learn a *function* to map x => y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Classification

Supervised Learning: Classification

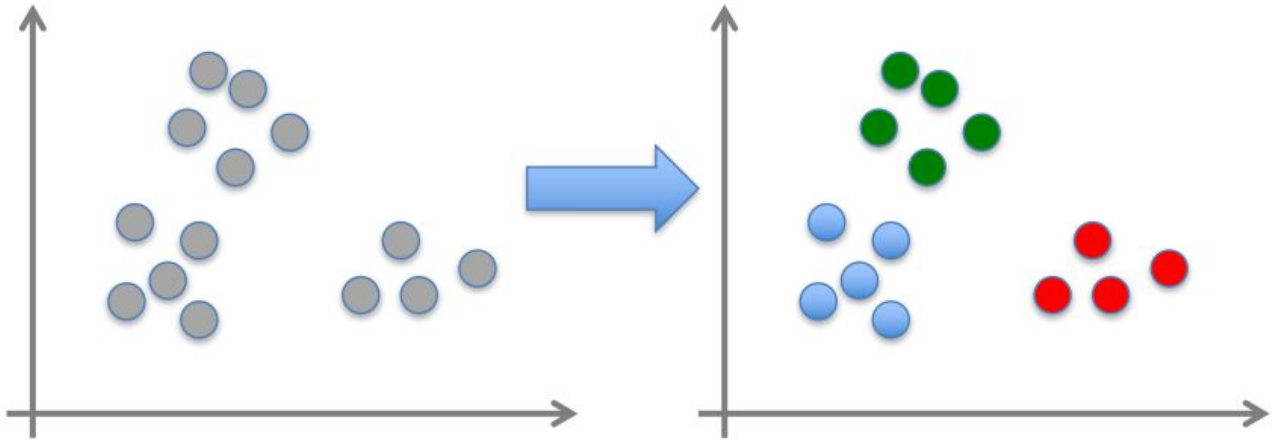
Discriminative vs Generative Algorithms



About Generative and Discriminative models - Jordi EsteveSorribas,medium

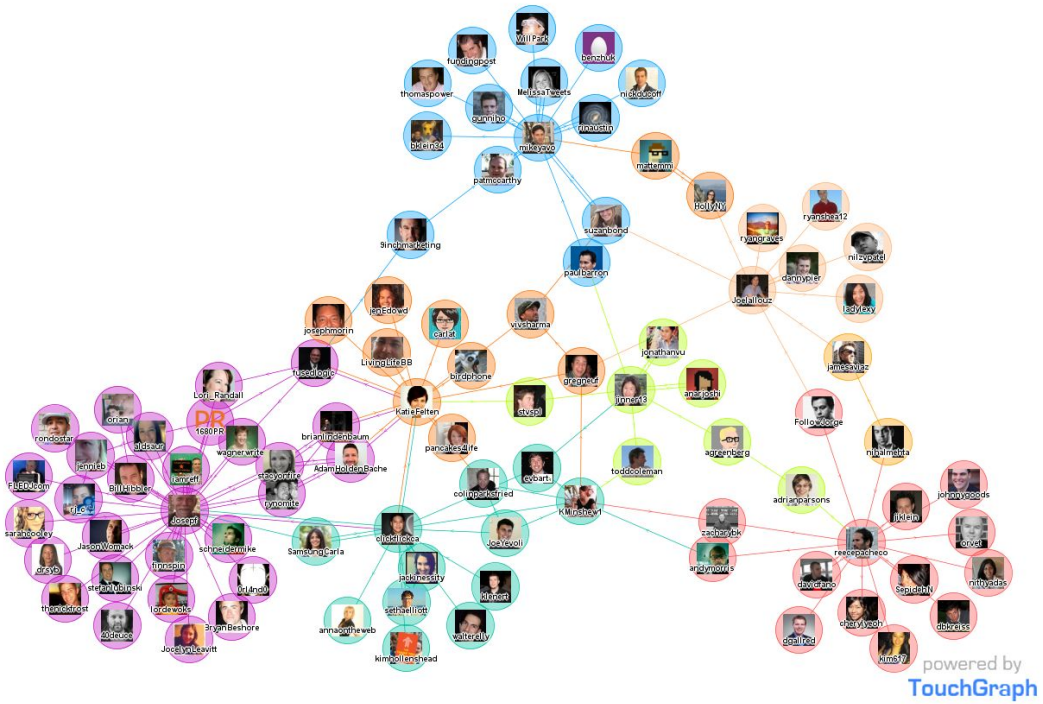
Unsupervised Learning: Clustering

- Given $\mathcal{T} = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$ (without labels) output hidden structure behind the $\mathbf{x}$'s – e.g., clustering



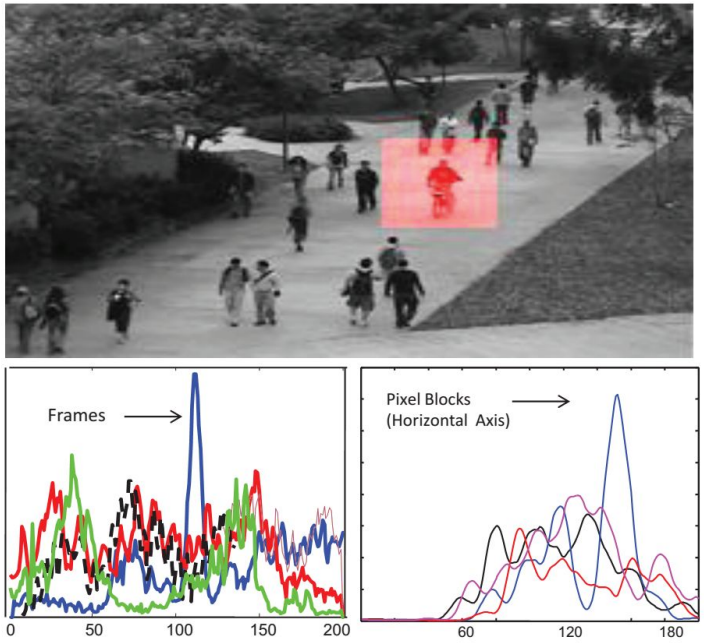
Unsupervised Learning

- Social network analysis



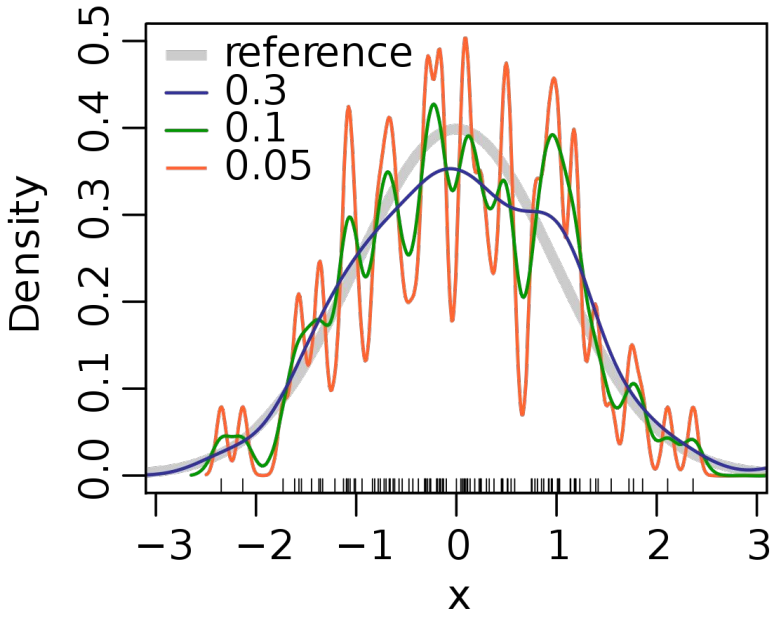
Unsupervised Learning

- **Anomaly detection:** the computer program sifts through a set of events or objects and flags some of them as being unusual or atypical.
- Example: credit card fraud detection, video surveillance.



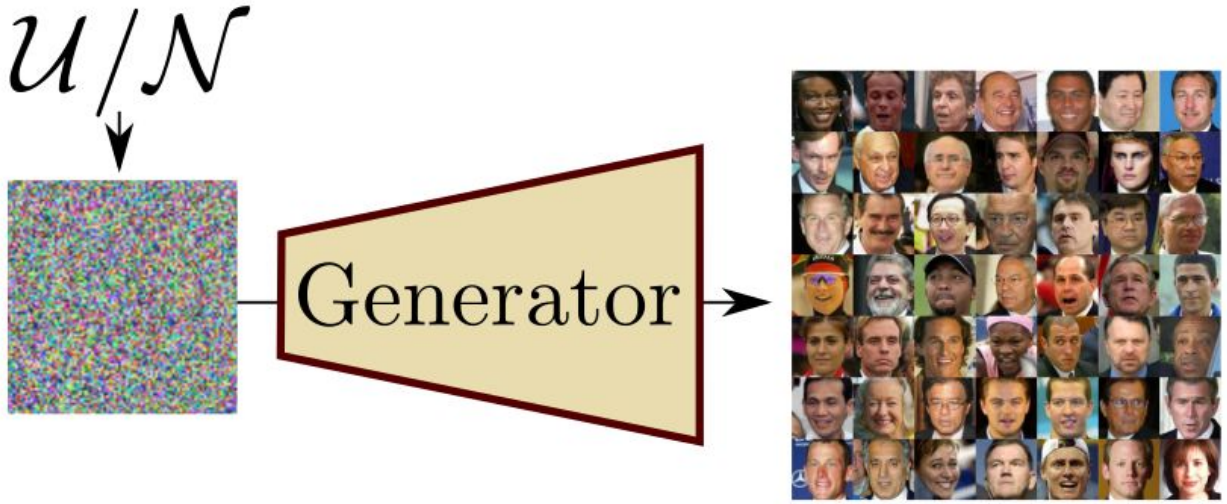
Unsupervised Learning

- Density Estimation: learn a function $p(\boldsymbol{x})$: $R^d \rightarrow R$



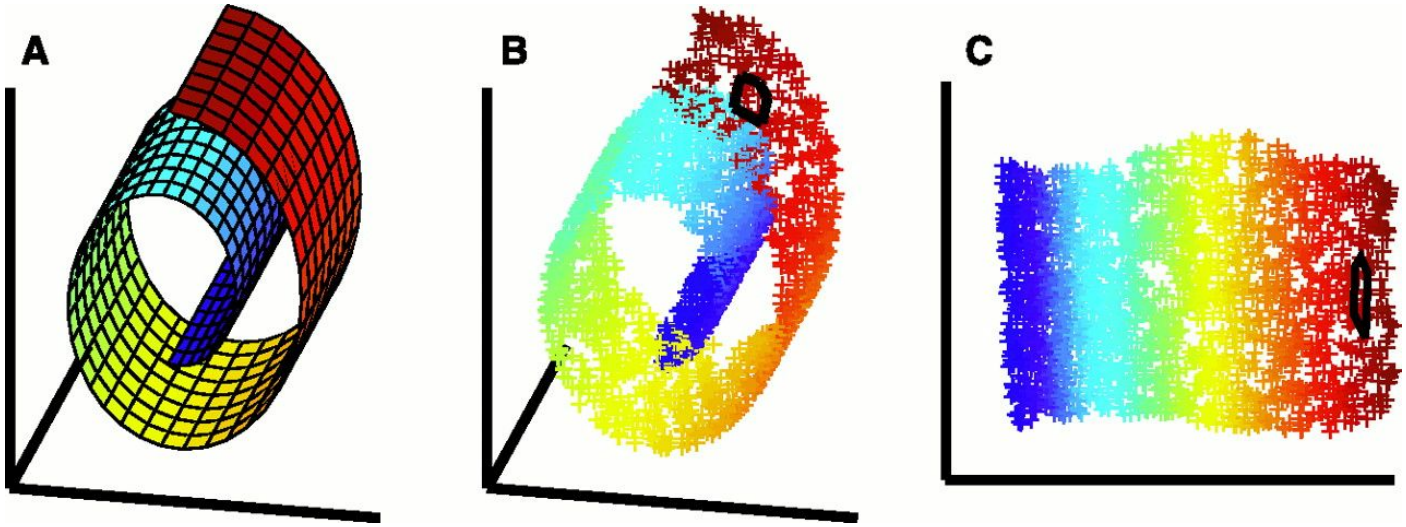
Unsupervised Learning

- Sampling: learn to sample according to $p(\boldsymbol{x})$: $R^d \rightarrow R$



Unsupervised Learning

- Dimensionality Reduction



Supervised vs Unsupervised Learning

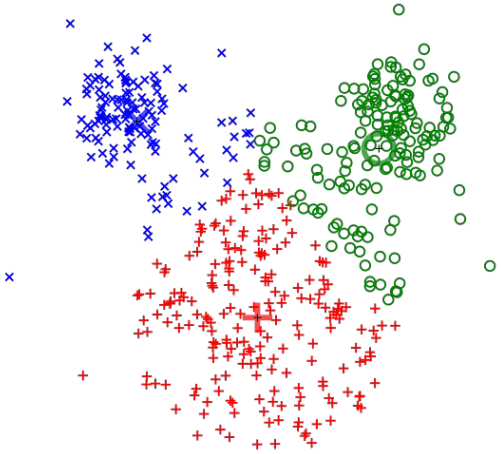
Unsupervised Learning

Data: x

Just data, no labels!

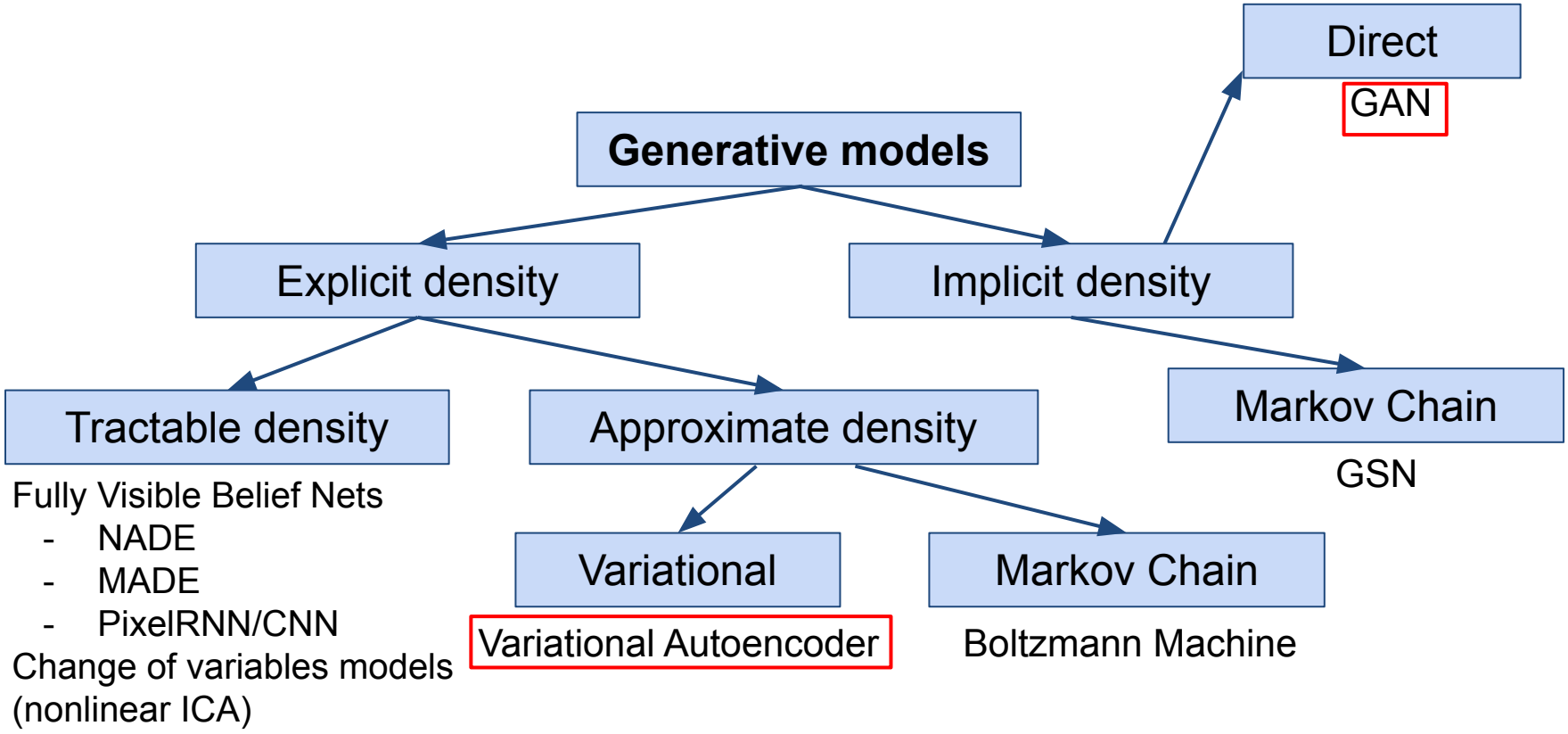
Goal: Learn some underlying hidden *structure* of the data

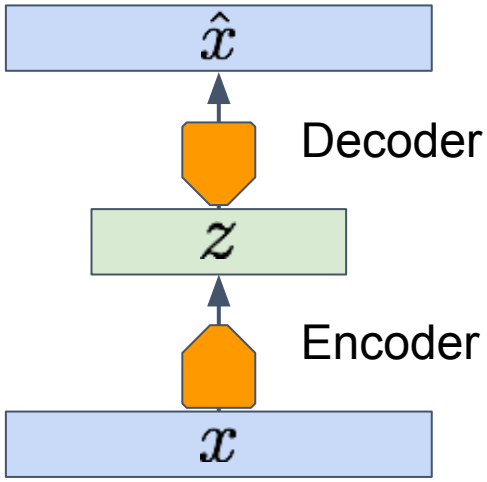
Examples: Clustering, dimensionality reduction, feature learning, density estimation, **learn a data sampler**, etc.



One of the solution for sampling in GAN.

Taxonomy of Generative Models

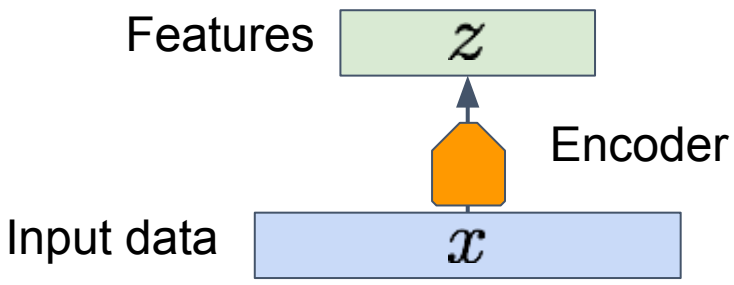


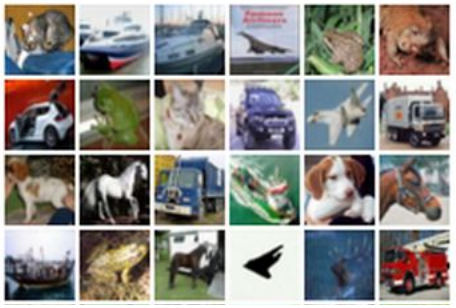


Autoencoders

Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data





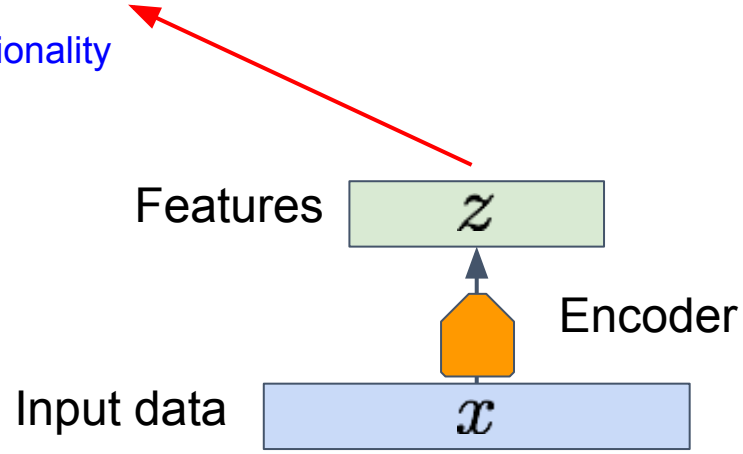
Auto-association by multilayer perceptrons and singular value decomposition, H Bourlard, Y Kamp - Biological cybernetics, 1988

Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

z usually smaller than **x**
(dimensionality reduction)

Q: Why dimensionality reduction?



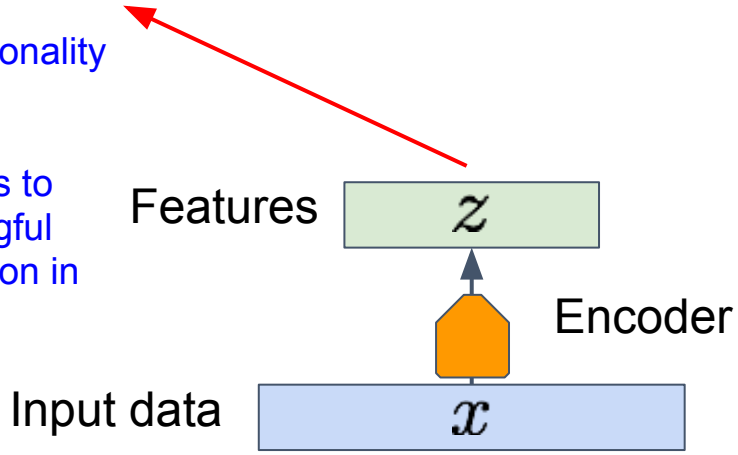
Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

z usually smaller than **x**
(dimensionality reduction)

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data



Autoencoders

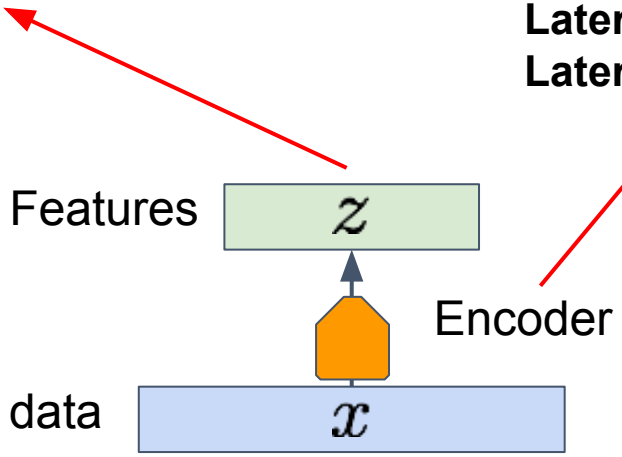
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

z usually smaller than **x** (dimensionality reduction)

Q: Why dimensionality reduction?

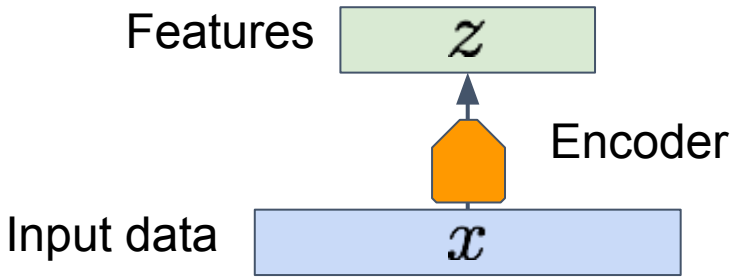
A: Want features to capture meaningful factors of variation in data

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN



Autoencoders

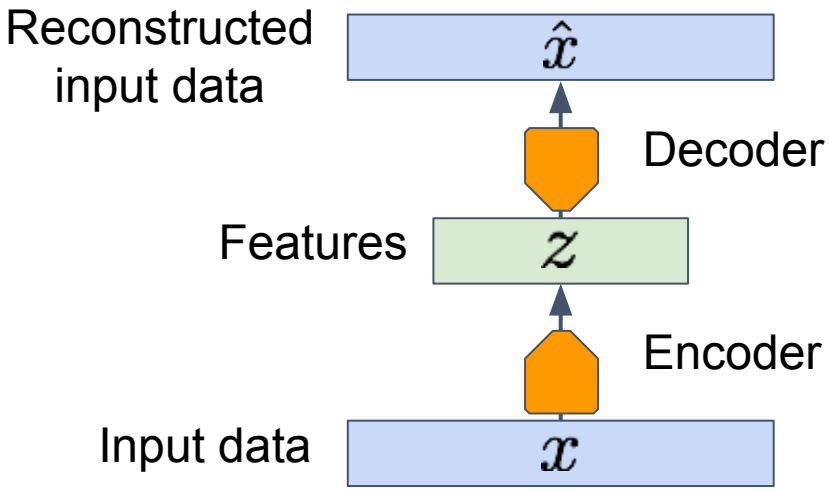
How to learn this feature representation?



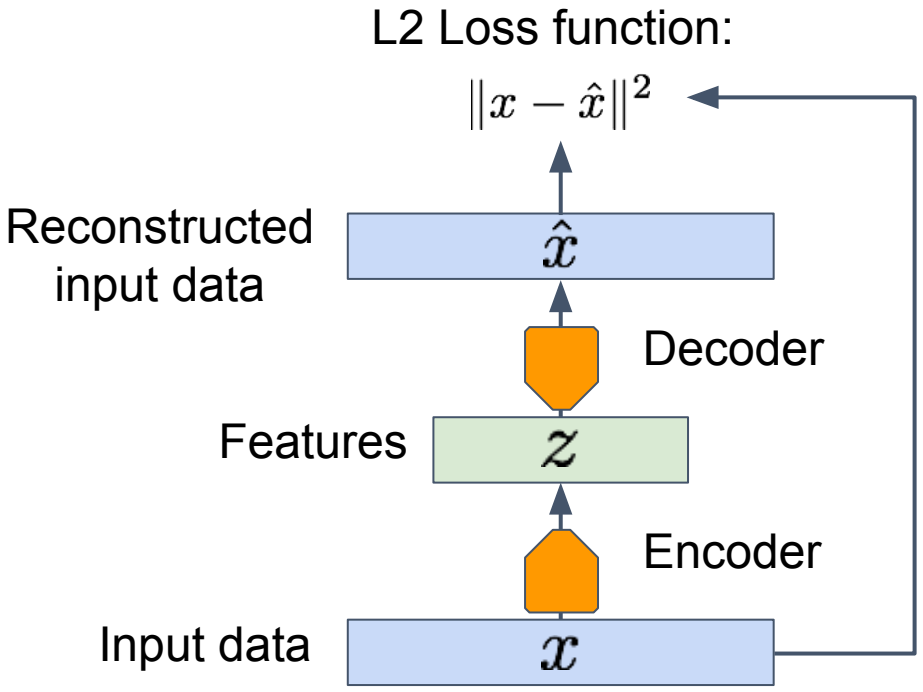
Autoencoders

Train such that features can be used to reconstruct original data

“Autoencoding” - encoding itself



Train such that features can be used to reconstruct original data



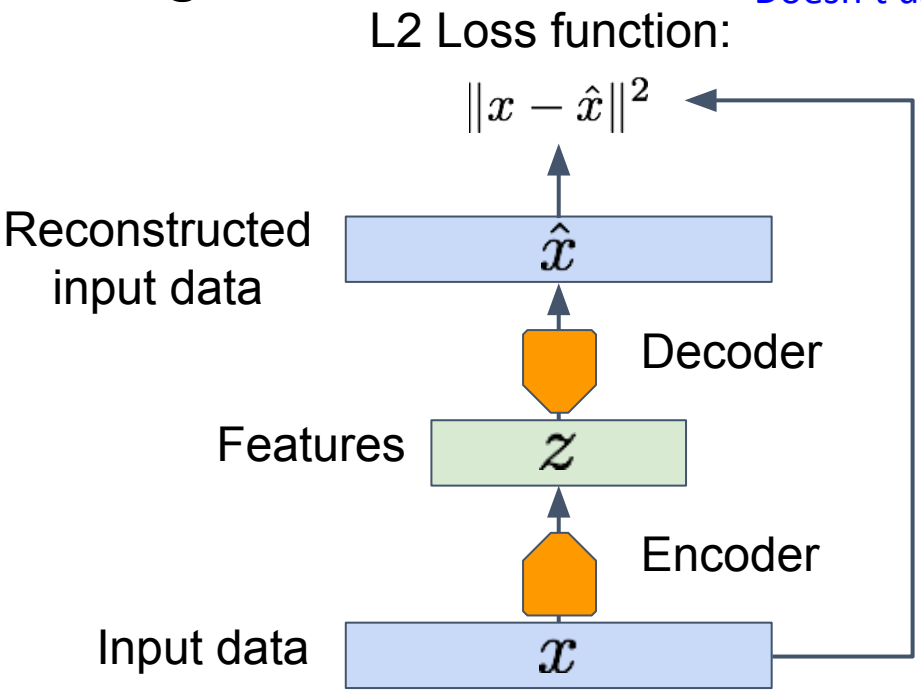
Reconstructed data

Input data

Autoencoders

Train such that features can be used to reconstruct original data

Doesn't use labels!

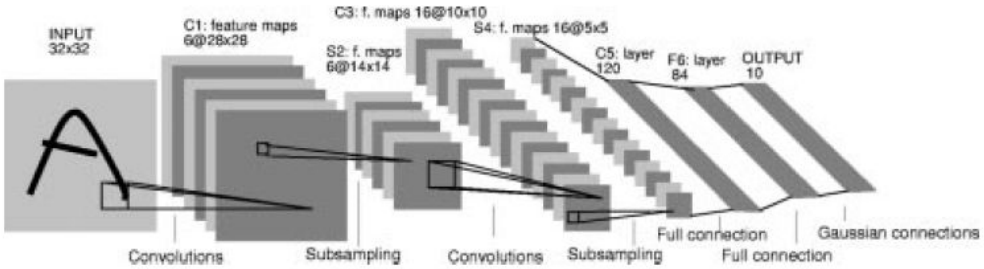


Reconstructed data

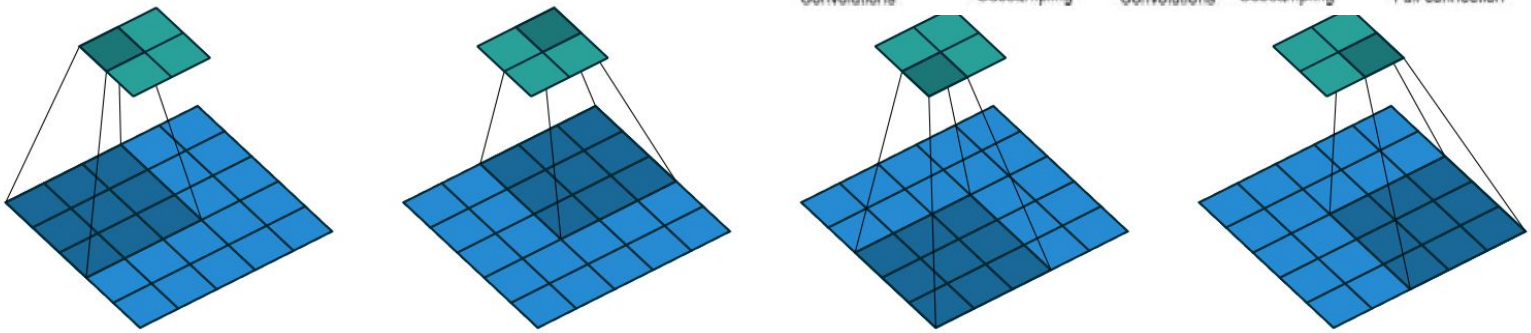
Input data

Autoencoders

Architecture:



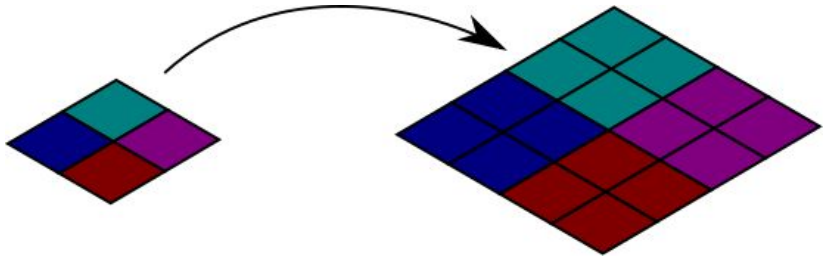
- Encoder: ConvNet architecture



Convolving a 3 × 3 kernel over a 5 × 5 input using 2 × 2 strides

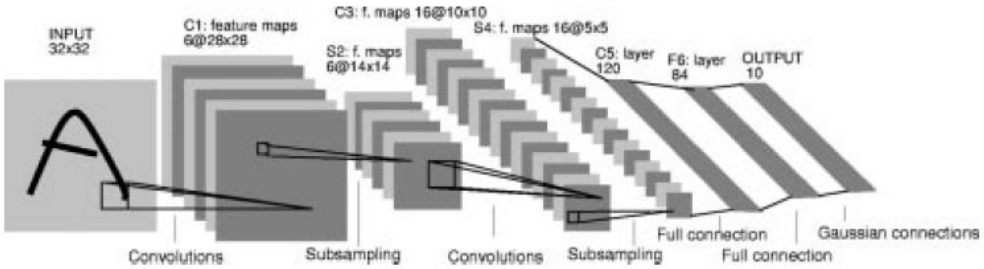
- Decoder: How to go from a vector to an image? How to “deconvolve”?

Up-sampling

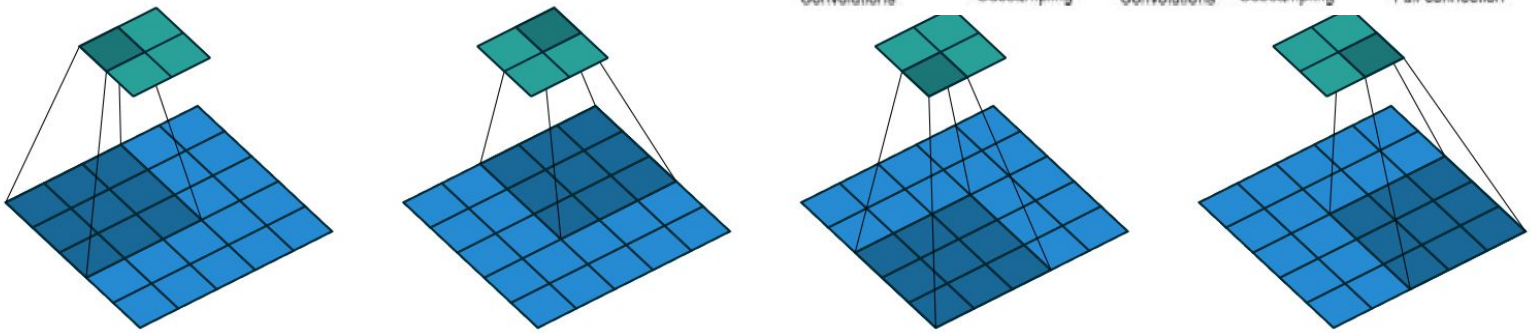


Autoencoders

Architecture:



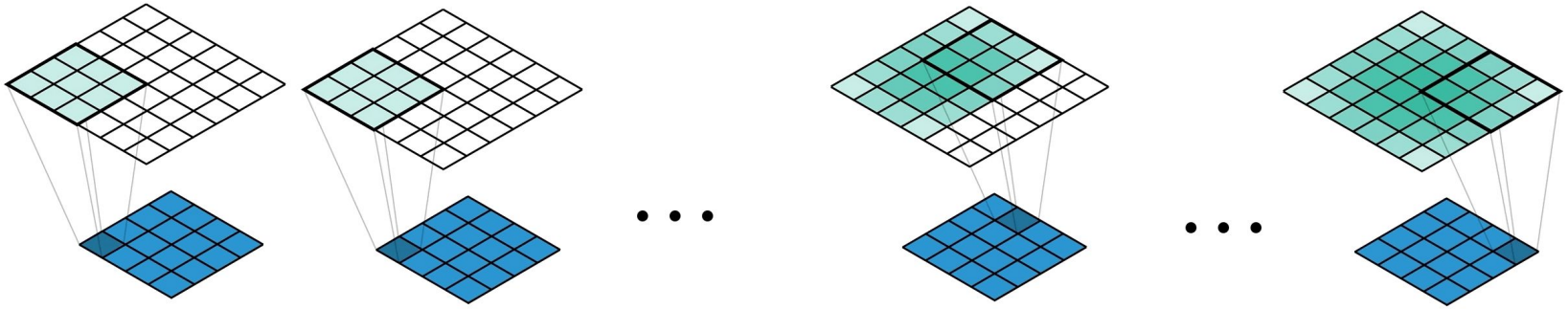
- Encoder: ConvNet architecture



Convolving a 3 × 3 kernel over a 5 × 5 input using 2 × 2 strides

- Decoder: How to go from a vector to an image? How to “deconvolve”?

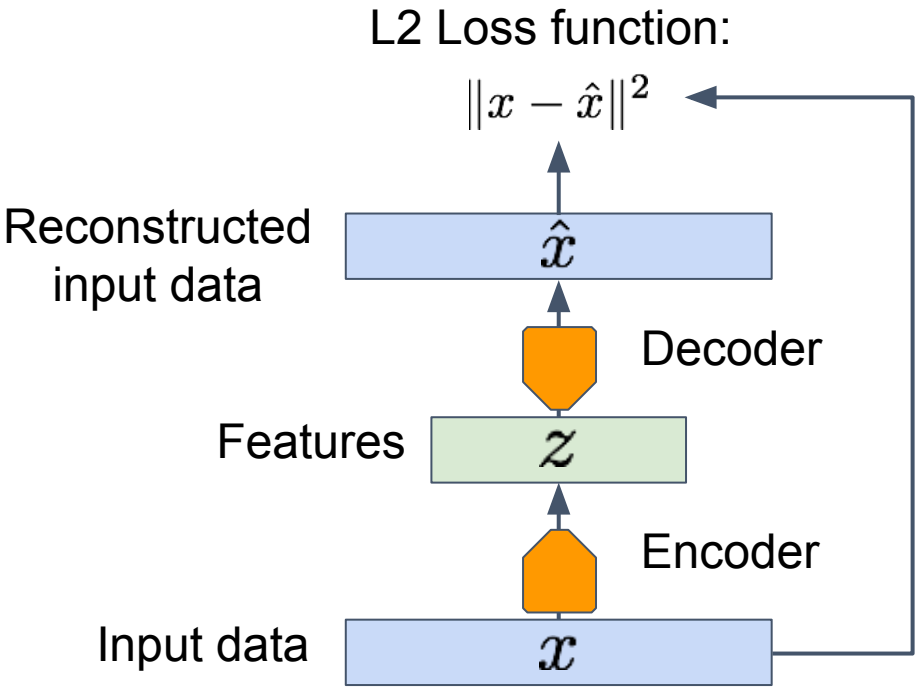
Transpose Convolution:



A guide to convolution arithmetic for deep learning ,Vincent Dumoulin, Francesco Visin

Autoencoders

Train such that features can be used to reconstruct original data



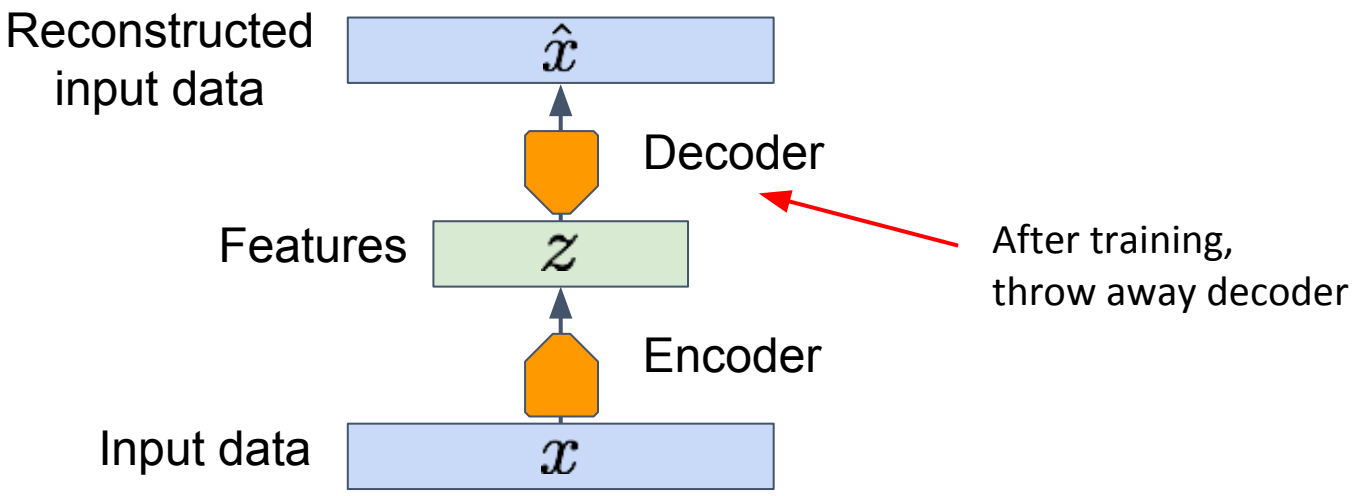
Reconstructed data

Input data

What can we do now?

Autoencoders

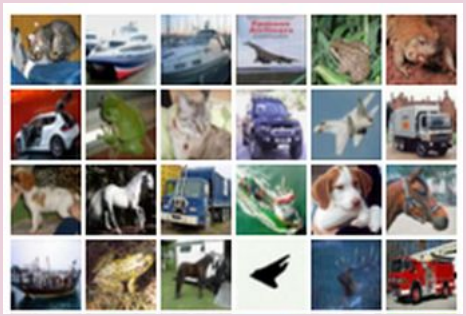
Encoder can be used to initialize a **supervised** model



Autoencoders

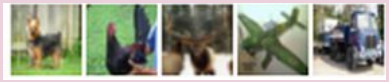
Encoder can be used to initialize a **supervised** model.

Unlabeled dataset



bird plane

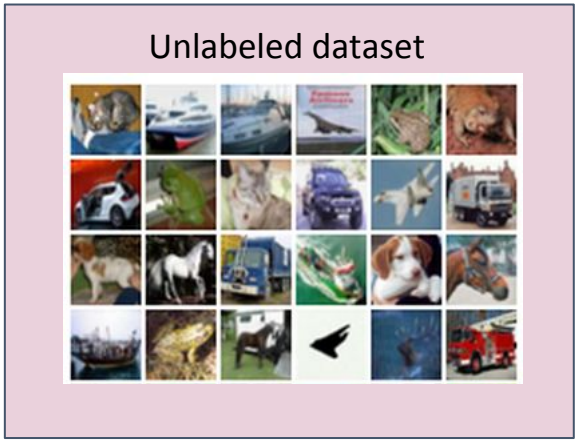
dog deer truck



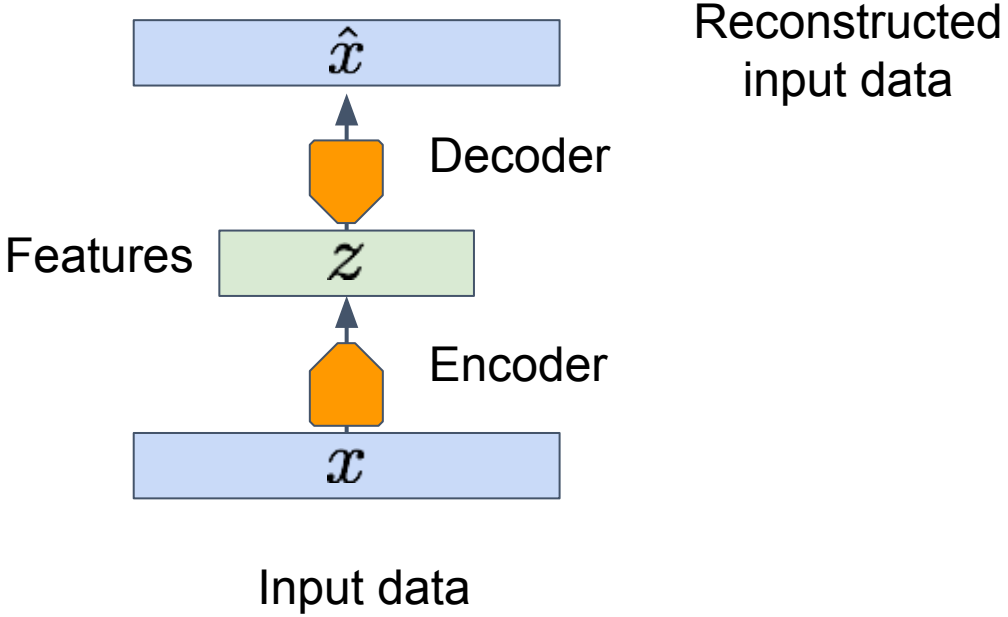
Small annotated dataset

Autoencoders

Encoder can be used to initialize a **supervised** model.

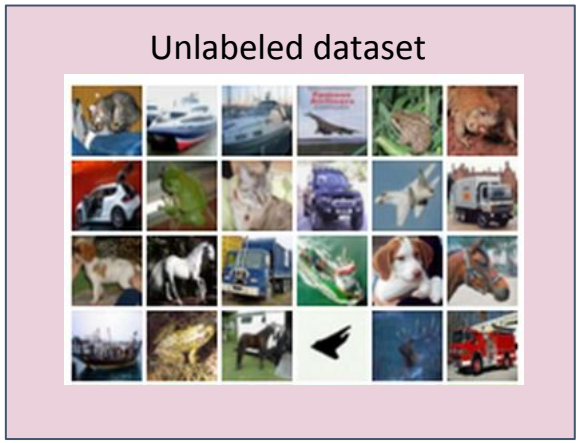


Pretrain an auto-encoder on the unlabeled data.

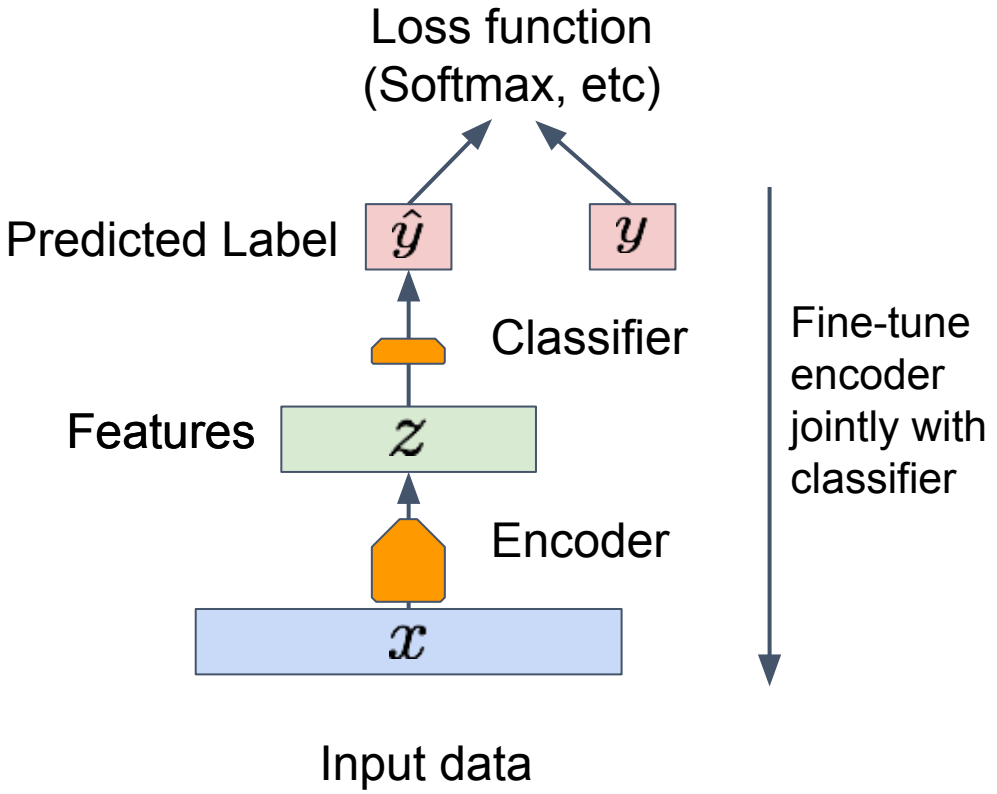


Autoencoders

Encoder can be used to initialize a **supervised** model.

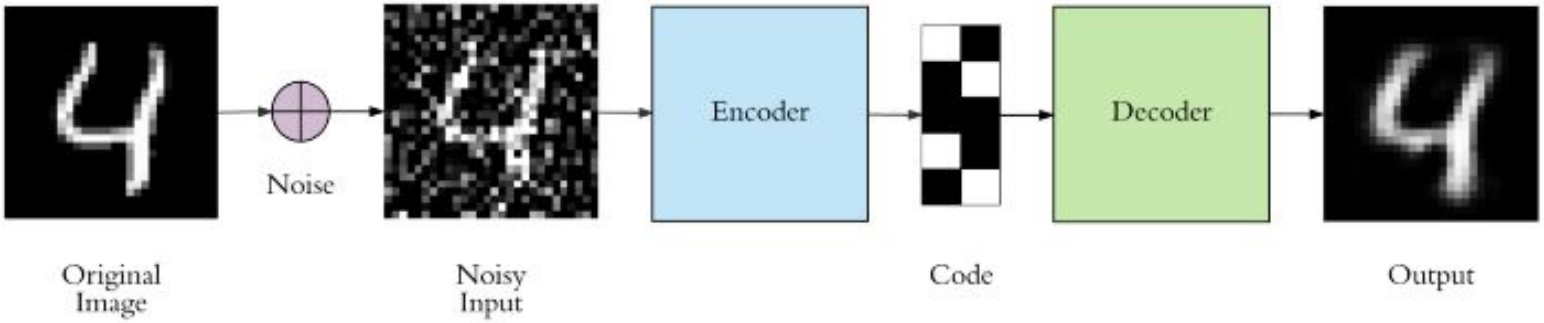


Train a classifier on the labeled data using the learned feature.



Denoising Autoencoders

- Keeping the code layer small (latent representation) forces the autoencoder to learn an intelligent representation of the data.
- Another way to force the autoencoder to learn useful features is adding random noise to its inputs and making it recover the original noise-free data.

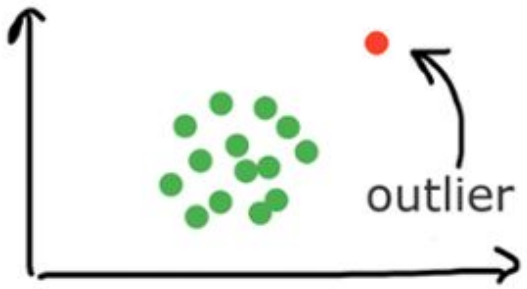


Vincent, H. Larochelle Y. Bengio and P.A. Manzagol, Extracting and Composing Robust Features with Denoising Autoencoders, Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08), pages 1096 - 1103, ACM, 2008.

Autoencoders

Auto-Encoder can:

- Reduce dimension
- Unsupervised pre-training
- Denoising
- Detect outlier $\|x - \hat{x}\|^2$



Autoencoders

Auto-Encoder can?

Autoencoders

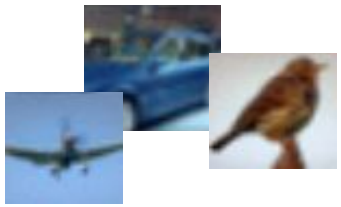
Auto-Encoder can:

- Reduce dimension
- Unsupervised pre-training
- Denoising
- Detect outlier $\|x - \hat{x}\|^2$

Auto-Encoder cannot:

- Estimate the density function p(x)
- Sample according to p(x)

Given training data, generate new samples from same distribution



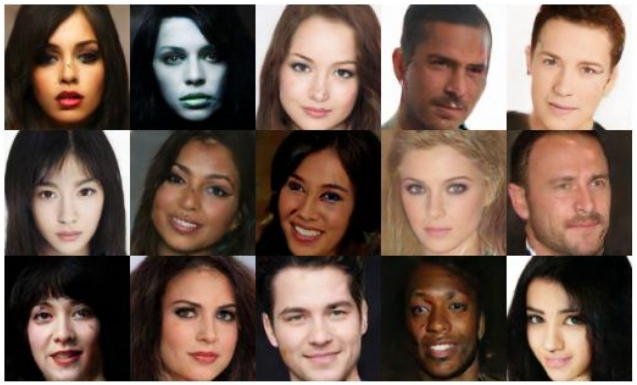
Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

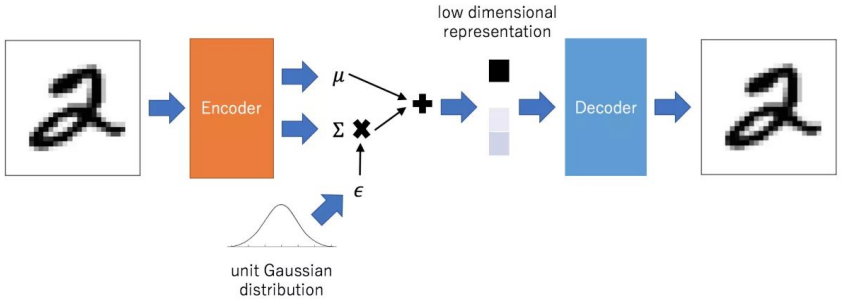
Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Why Generative Models?

- Realistic samples for artwork, super-resolution, colorization, etc.


- Generative models of time-series data can be used for simulation and planning (reinforcement learning applications!)
- Training generative models can also enable inference of **latent representations** that can be useful as general features



Variational Autoencoders (VAE)

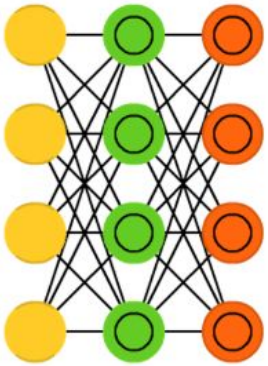
Variational Autoencoders

VAEs are a combination of the following ideas:

1. Autoencoders
2. Variational Approximation & Variational Lower Bound
3. “Reparameterization” Trick

Variational Autoencoders

D. Kingma, M. Welling, *Auto-Encoding Variational Bayes*, ICLR, 2014



Variational autoencoders (VAE) have the same architecture as AEs but are "taught" something else: an approximated probability distribution of the input samples. It's a bit back to the roots as they are bit more closely related to BMs and RBMs. They do however rely on Bayesian mathematics regarding probabilistic inference and independence, as well as a re-parametrisation trick to achieve this different representation. The inference and independence parts make sense intuitively, but they rely on somewhat complex mathematics. The basics come down to this: take influence into account. If one thing happens in one place and something else happens somewhere else, they are not necessarily related. If they are not related, then the error propagation should consider that. This is a useful approach because neural networks are large graphs (in a way), so it helps if you can rule out influence from some nodes to other nodes as you dive into deeper layers.

Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).

Original Paper PDF

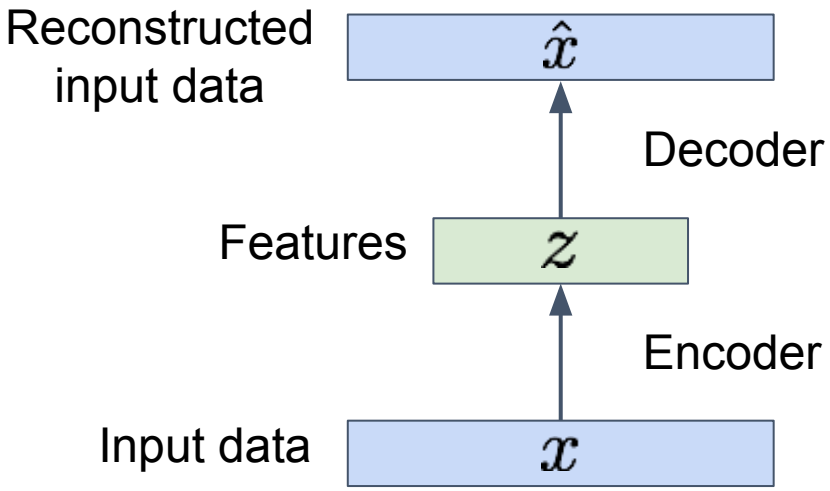
Variational Autoencoders

VAEs are a combination of the following ideas:

1. **Autoencoders**
2. Variational Approximation & Variational Lower Bound
3. “Reparameterization” Trick

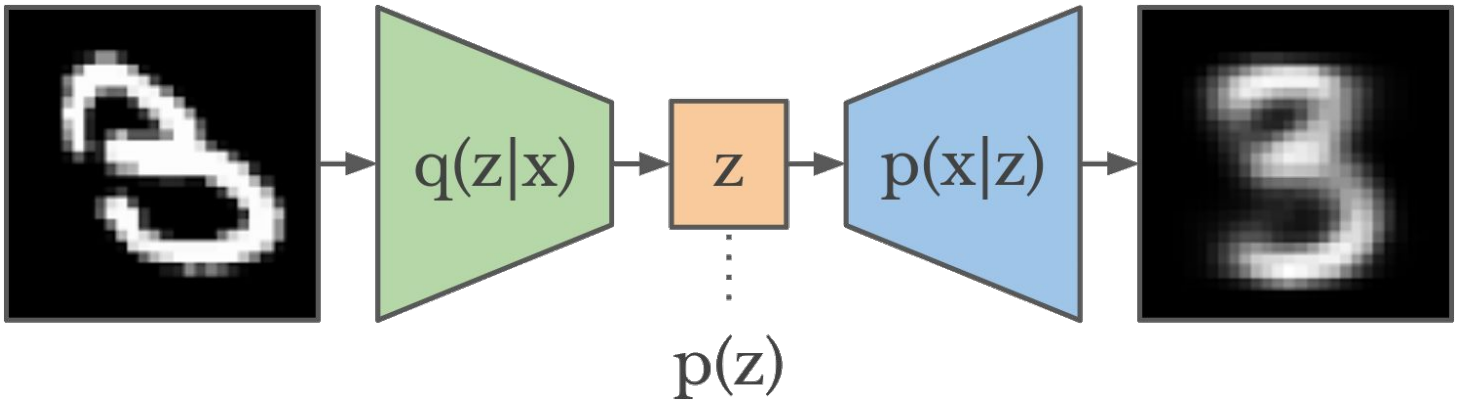
Autoencoders

Autoencoders can reconstruct data, and can learn features to initialize a supervised model.
Features capture factors of variation in training data. **Can we generate new images from an autoencoder?**



Variational Autoencoders

Probabilistic spin on autoencoders - will let us sample from the model to generate data!



Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014
C. Doearsh, Tutorial on Variational Autoencoders, https://arxiv.org/pdf/1606.05908.pdf

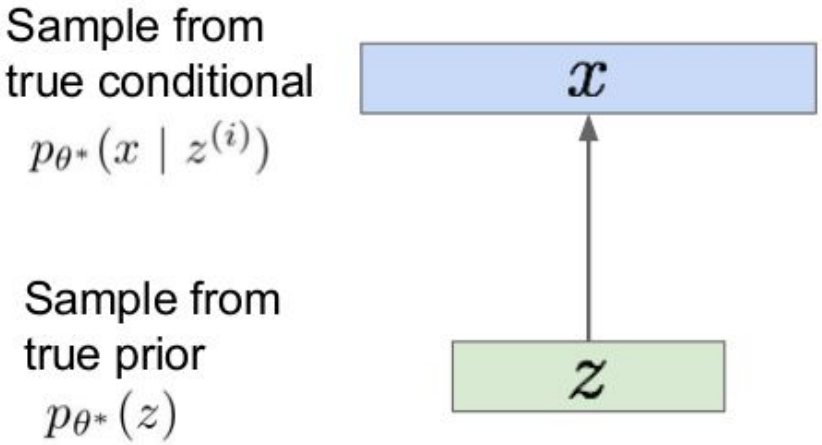
Variational Autoencoders

VAEs are a combination of the following ideas:

1. Autoencoders
2. **Variational Approximation & Variational Lower Bound**
3. “Reparameterization” Trick

Variational Autoencoders

Assume training data representation x is generated from underlying unobserved (latent) z

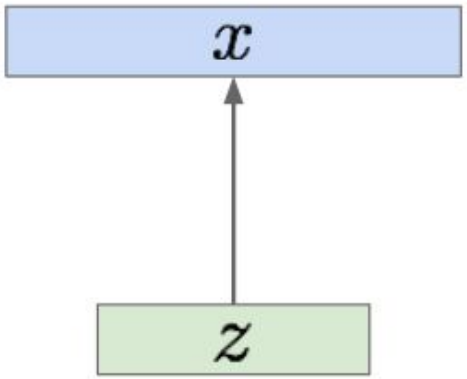


Variational Autoencoders

Assume training data representation x is generated from underlying unobserved (latent) z

Sample from true conditional $p_{\theta^*}(x \mid z^{(i)})$

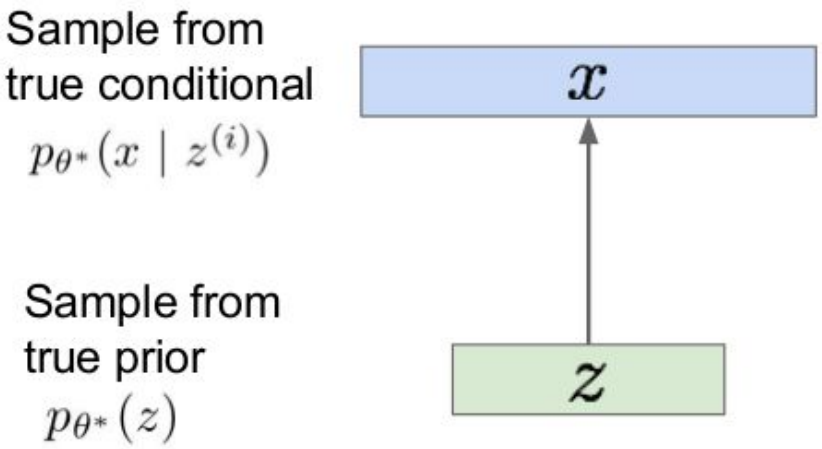
Sample from true prior $p_{\theta^*}(z)$



Intuition:
x is an image, z is a latent factor used to generate x: attributes, orientation, etc.

Variational Autoencoders

Assume training data representation x is generated from underlying unobserved (latent) z



Intuition:
x is an image, z is a latent factor used to generate x: attributes, orientation, etc.

We want to estimate the optimal parameters θ^* of this generative model, according to maximum likelihood

But what is a loss? But maximum likelihood

- **Examples with supervised training**
  - Given training samples

$$T = \{(\mathbf{x}_1, \mathbf{t}_1), (\mathbf{x}_2, \mathbf{t}_2), \ldots, (\mathbf{x}_N, \mathbf{t}_N)\}$$

  - Choose an objective function (loss function)
  - Adjust all the weights of the network θ such that a cost function is minimized

$$\min_w = L_{tot}(w)$$
$$= \sum_i L(\mathbf{t}_i, y(\mathbf{x}_i; w))$$

Loss: Regression

The choice of the loss function depends on the problem.

MSE loss:

$$\frac{1}{2}\sum_{n=1}^{N}\|\mathbf{y}(\mathbf{x}_n,\mathbf{w})-\mathbf{t}_n\|^2$$

But why?

Let's assume a gaussian distribution of the error:

$$p(t|\mathbf{x},\mathbf{w})=\mathcal{N}\left(t|y(\mathbf{x},\mathbf{w}),\beta^{-1}\right)$$

Dataset likelihood:

$$p(\mathbf{t}|\mathbf{X},\mathbf{w},\beta)=\prod_{n=1}^{N}p(t_n|\mathbf{x}_n,\mathbf{w},\beta)$$

Loss: Regression but why MSE?

Let's assume a gaussian distribution of the error:

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$

$$\frac{1}{n}\frac{N}{}, \mathbf{w}) - \mathbf{t}_n\|^2$$

Dataset likelihood:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \mathbf{w}, \beta)$$

We take -log:

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi)$$

Loss: Regression but why MSE?

Maximize the data log-likelihood assuming gaussian errors

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$

$$\Leftrightarrow$$

Minimize the MSE:

$$\frac{1}{2}\sum_{n=1}^{N} \|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2$$

Loss: classification

- Classification problem with `C` classes: negative loglikelihood

$$\text{loss}(x) = -\log\left(prob(c = c_{\text{True}}|x)\right)$$

- How to get probabilities:

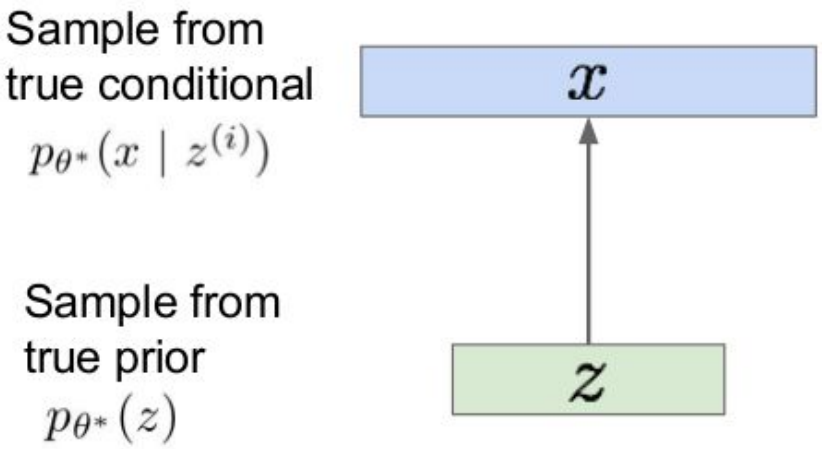
$$Softmax(x) = \frac{\exp(x)}{\sum_c \exp(x[c])} \qquad \sum_{c=0}^{C} Softmax(x)[c] = 1$$

- **Cross-entropy loss**

$$\text{loss}(x) = -\log\left(Softmax(x)[c_{\text{True}}]\right)$$

Variational Autoencoders

Assume training data representation x is generated from underlying unobserved (latent) z

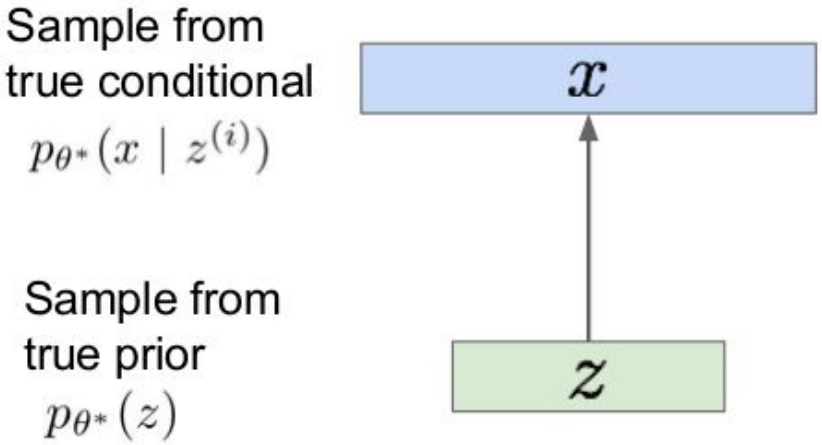


Intuition:
x is an image, z is a latent factor used to generate x: attributes, orientation, etc.

We want to estimate the optimal parameters θ^* of this generative model, according to maximum likelihood

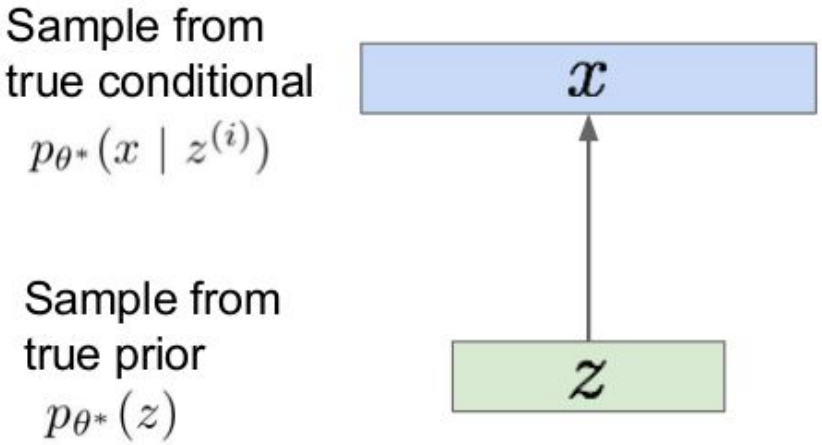
Variational Autoencoders

How should we **represent** this model?
Choose prior $p(z)$ to be simple, e.g. Gaussian.
Conditional $p(x|z)$ is complex (generates image) => represent with neural network



Variational Autoencoders

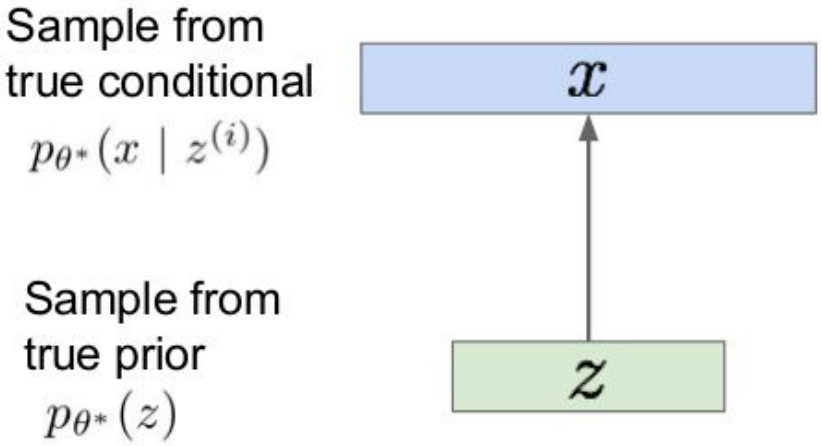
How should we **train** this model?
Learn model parameters to maximize likelihood of training data



$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Variational Autoencoders

How should we **train** this model?
Learn model parameters to maximize likelihood of training data



$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Intractable!

Variational Autoencoders

Data likelihood: $p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$

Intractable to compute p(x|z) for every z!

Simple Gaussian prior

Decoder neural network

Variational Autoencoders

Data likelihood: $p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$

Intractable to compute p(x|z) for every z!

Simple Gaussian prior

Decoder neural network

Posterior density also intractable: $p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$

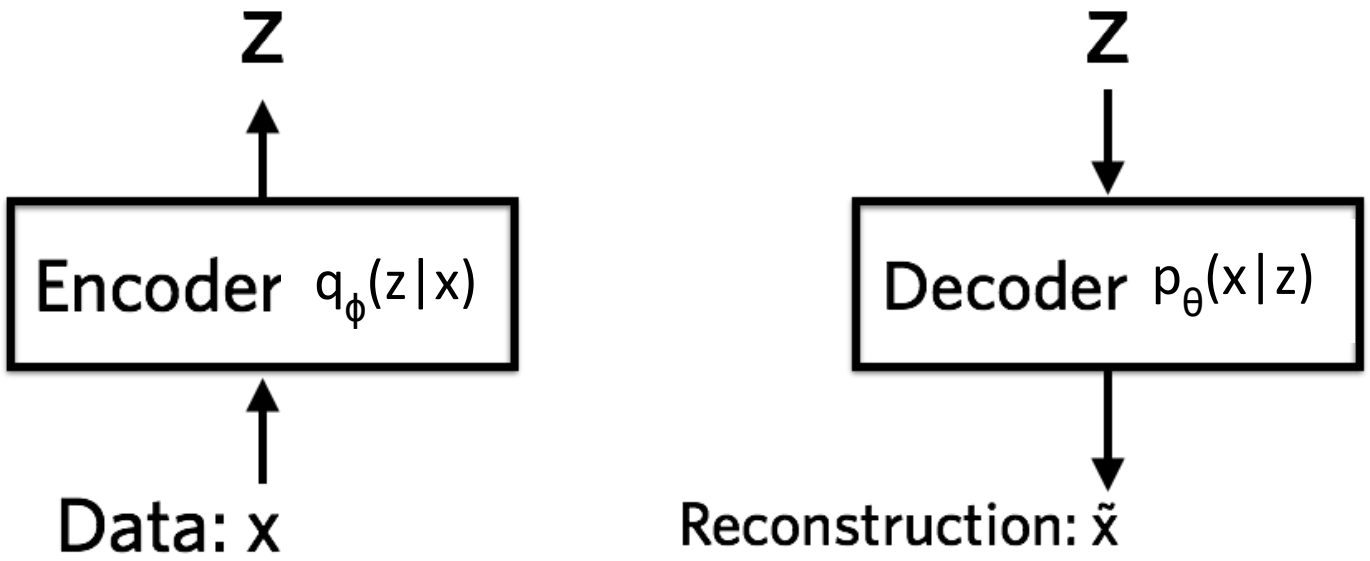
Decoder neural network

Simple gaussian prior

Intractable data likelihood

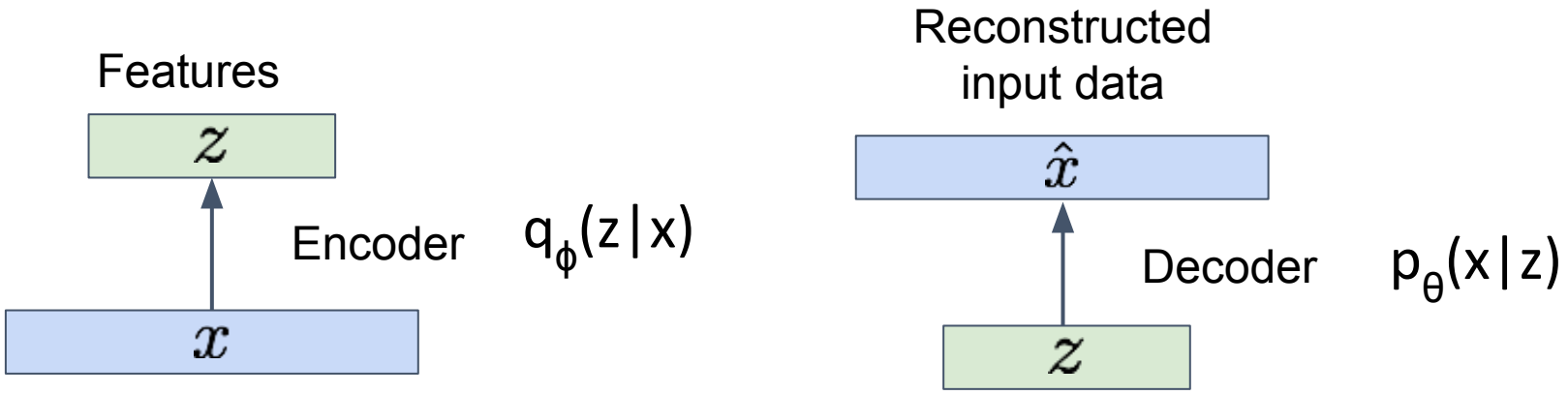
Variational Approximation

- Solution: In addition to decoder network modeling $p_\theta(x|z)$, define **additional encoder network** $q_\phi(z|x)$ that approximates $p_\theta(z|x)$
- This allows us to derive a **lower bound** on the data likelihood that is tractable, which we can optimize



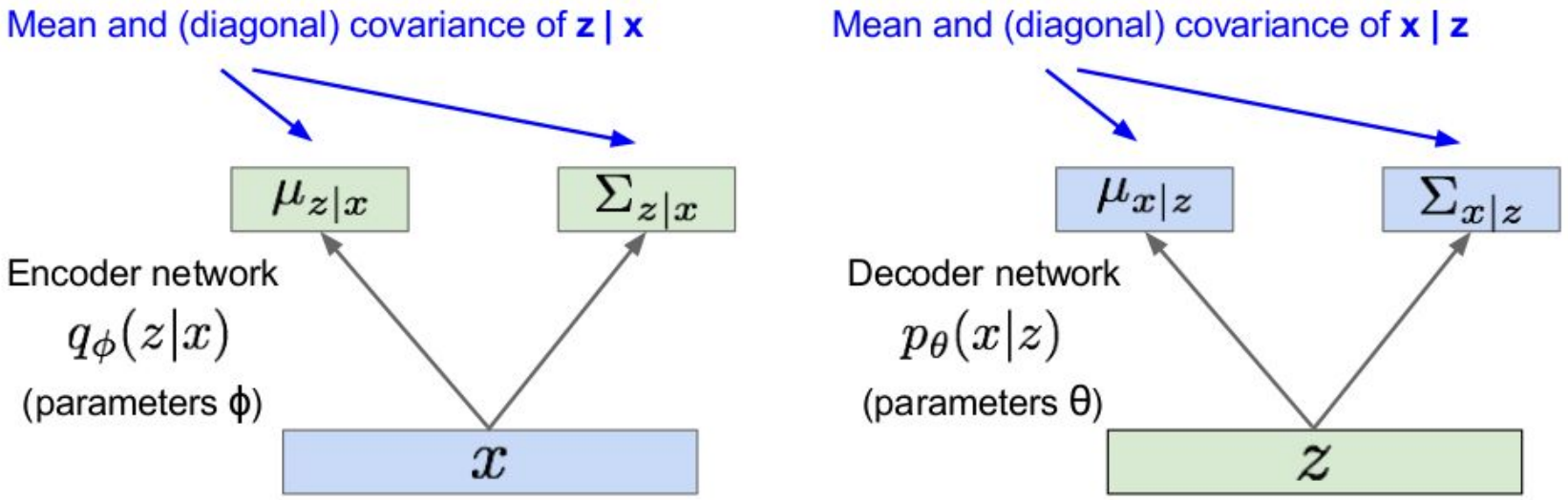
Variational Approximation

- Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic



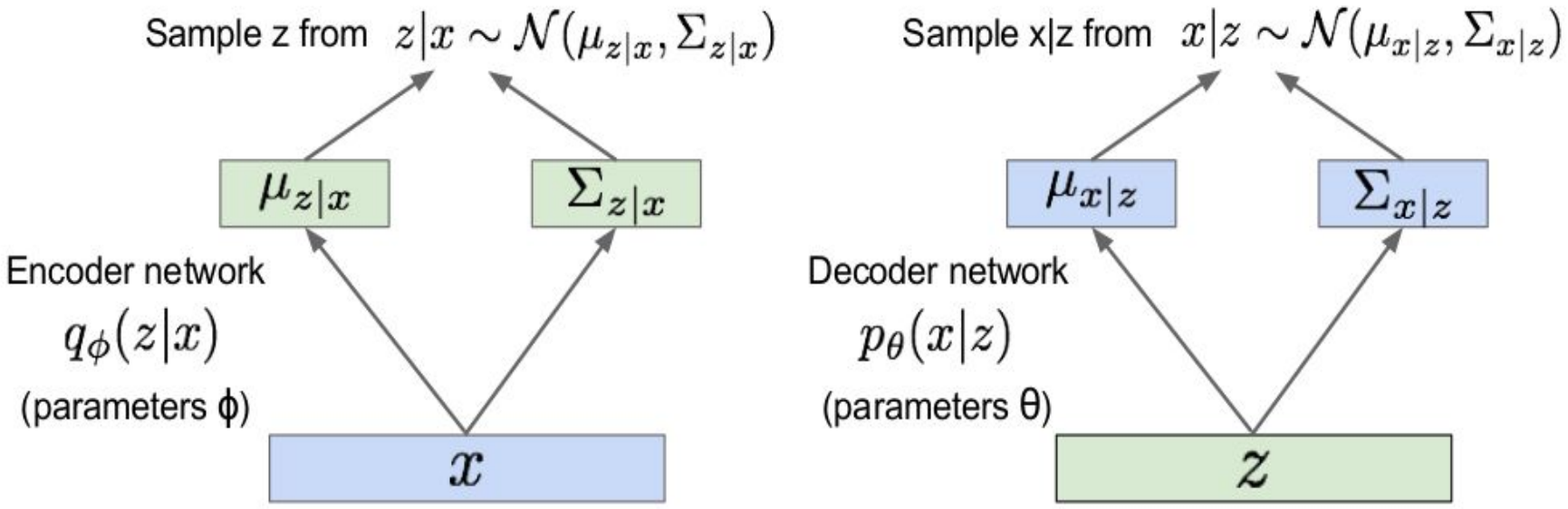
Variational Approximation

- Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic



Variational Approximation

- Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic



- Encoder and decoder networks also called “recognition”/“inference” and “generation” networks

Variational Approximation

- Let us consider the log likelihood of the data

Variational Approximation

- Let us consider the log likelihood of the data

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

The expectation wrt. z (using encoder network) let us write nice KL terms

Variational Approximation

- Let us consider the log likelihood of the data

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))$$

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling. (Sampling differentiable through reparam. trick, see paper.)

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution!

$p_\theta(z|x)$ intractable (saw earlier), can't compute this KL term :(But we know KL divergence always >= 0.

Variational Approximation

- Let us consider the log likelihood of the data

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

We want to maximize the data likelihood

Tractable lower bound which we can take gradient of and optimize! ($p_\theta(x|z)$ differentiable, KL term differentiable)

Variational Approximation

- Let us consider the log likelihood of the data

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{>0}$$

We want to maximize the data likelihood

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)$$

Variational lower bound ("ELBO")

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi)$$

Training: Maximize lower bound

Variational Approximation

- Let us consider the log likelihood of the data

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{>0}$$

Reconstruct the input data

Make approximate posterior distribution close to prior

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)$$

Variational lower bound ("ELBO")

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi)$$

Training: Maximize lower bound

Variational Autoencoders: recap

- Let us consider the log likelihood of the data

Variational Autoencoders

Maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Variational Autoencoders

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

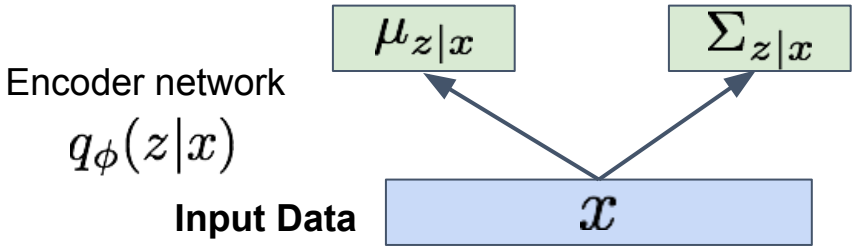
Let's look at computing the bound (forward pass) for a given minibatch of input data

Input Data x

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

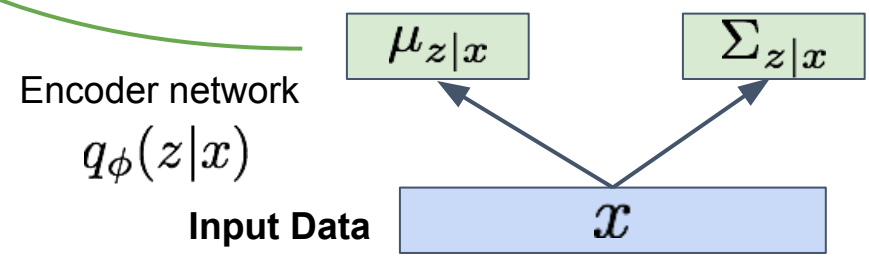


Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Make approximate posterior distribution close to prior

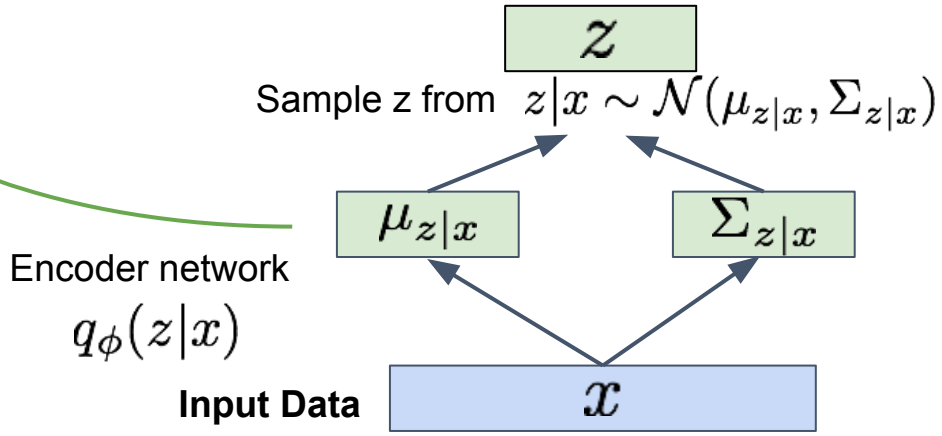


Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

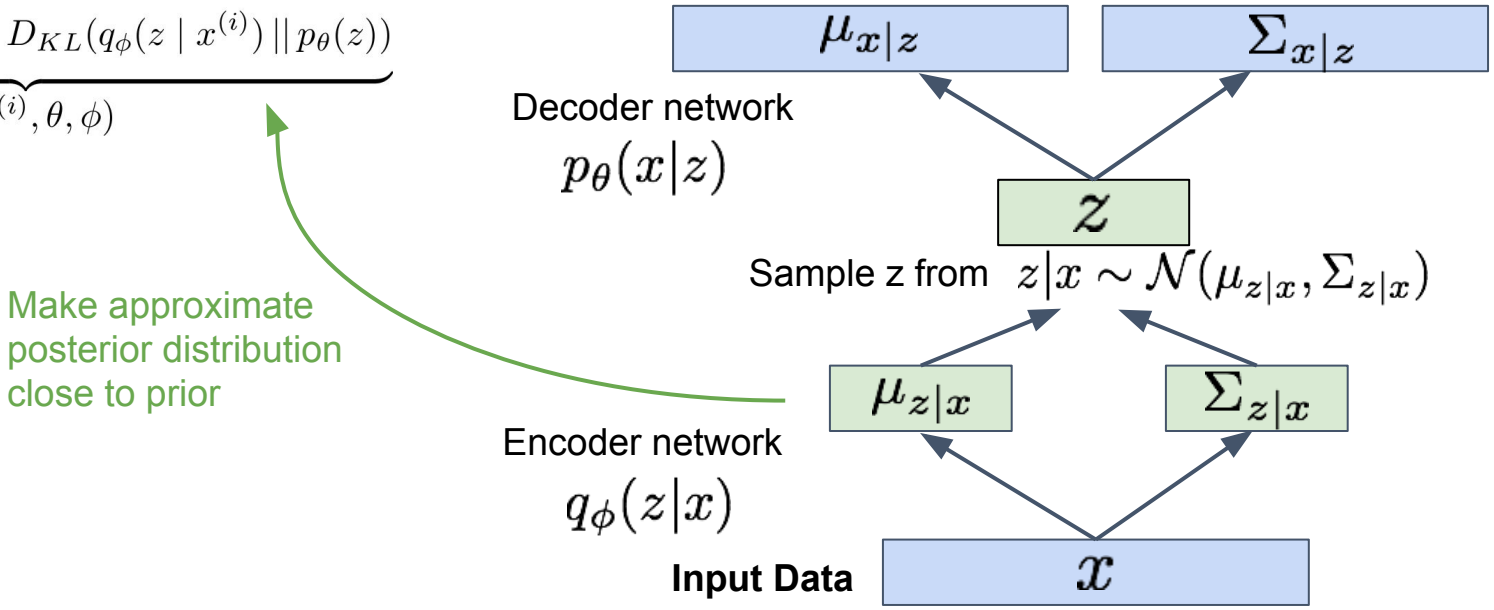
Make approximate posterior distribution close to prior



Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

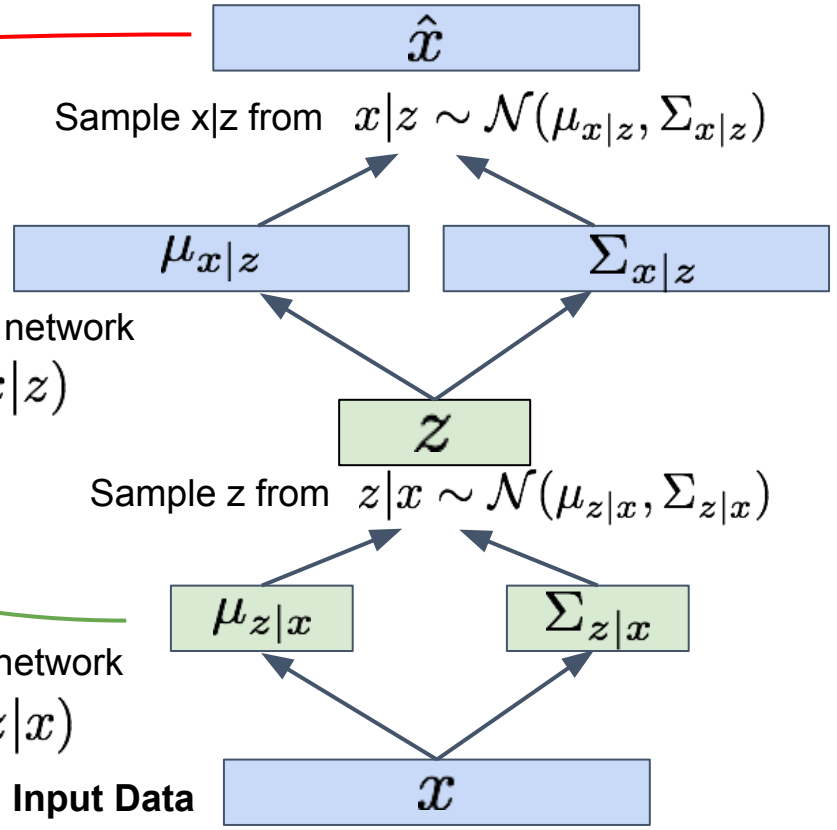


Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Maximize likelihood of original input being reconstructed



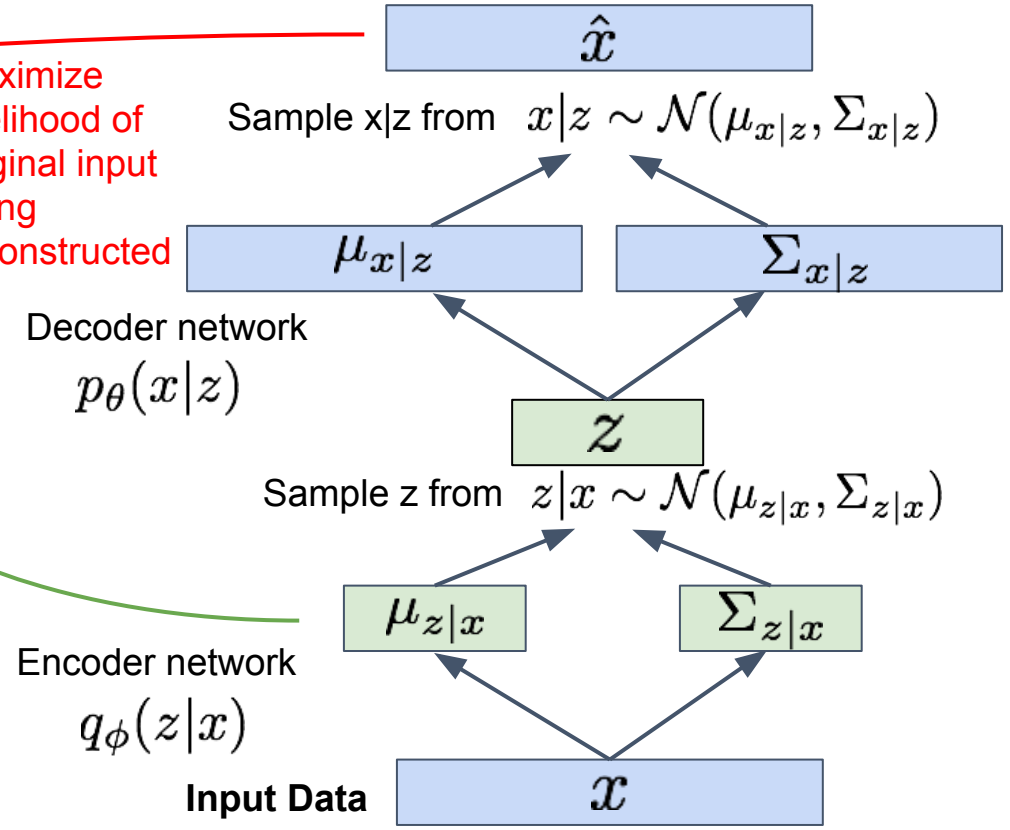
Make approximate posterior distribution close to prior

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Maximize likelihood of original input being reconstructed

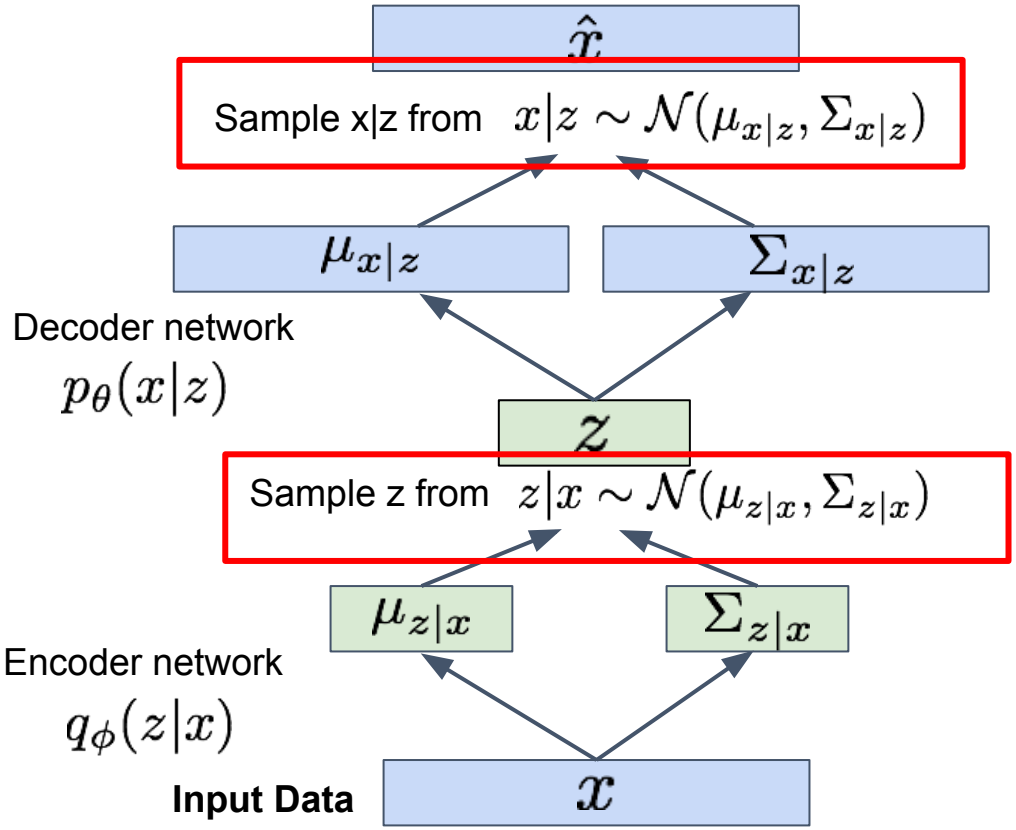


Make approximate posterior distribution close to prior

For every minibatch of input data: compute this forward pass, and then backprop!

Employ "Reparameterization" Trick

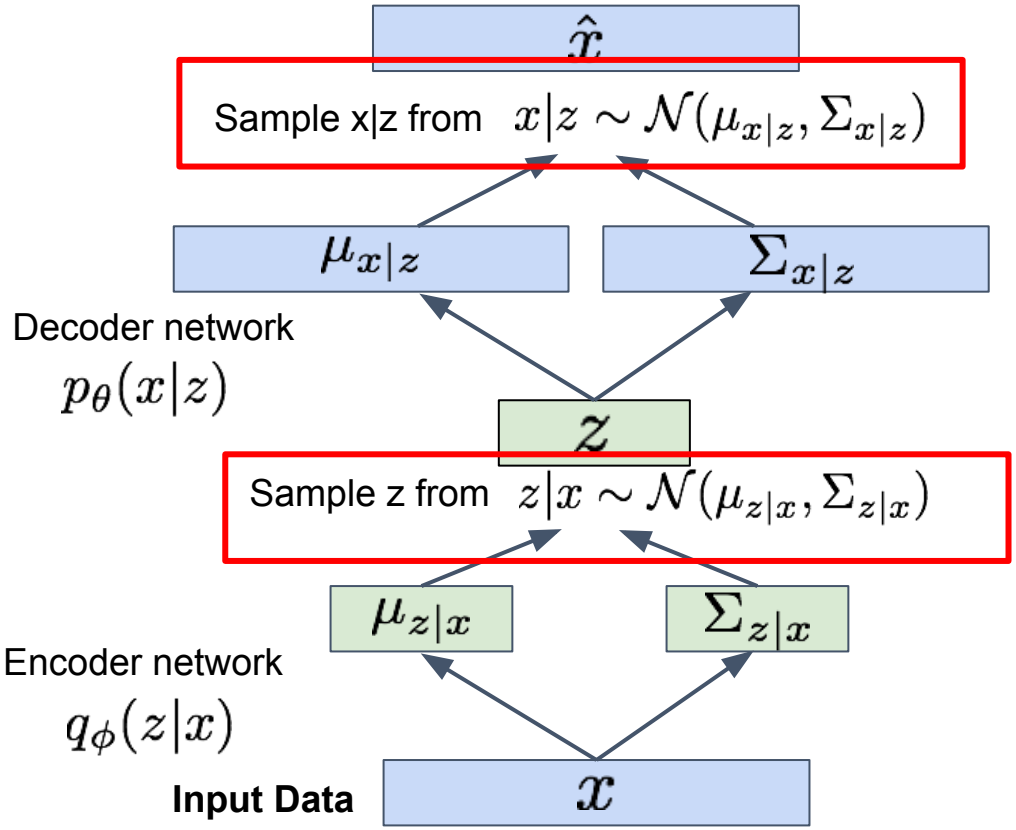
Variational Autoencoders



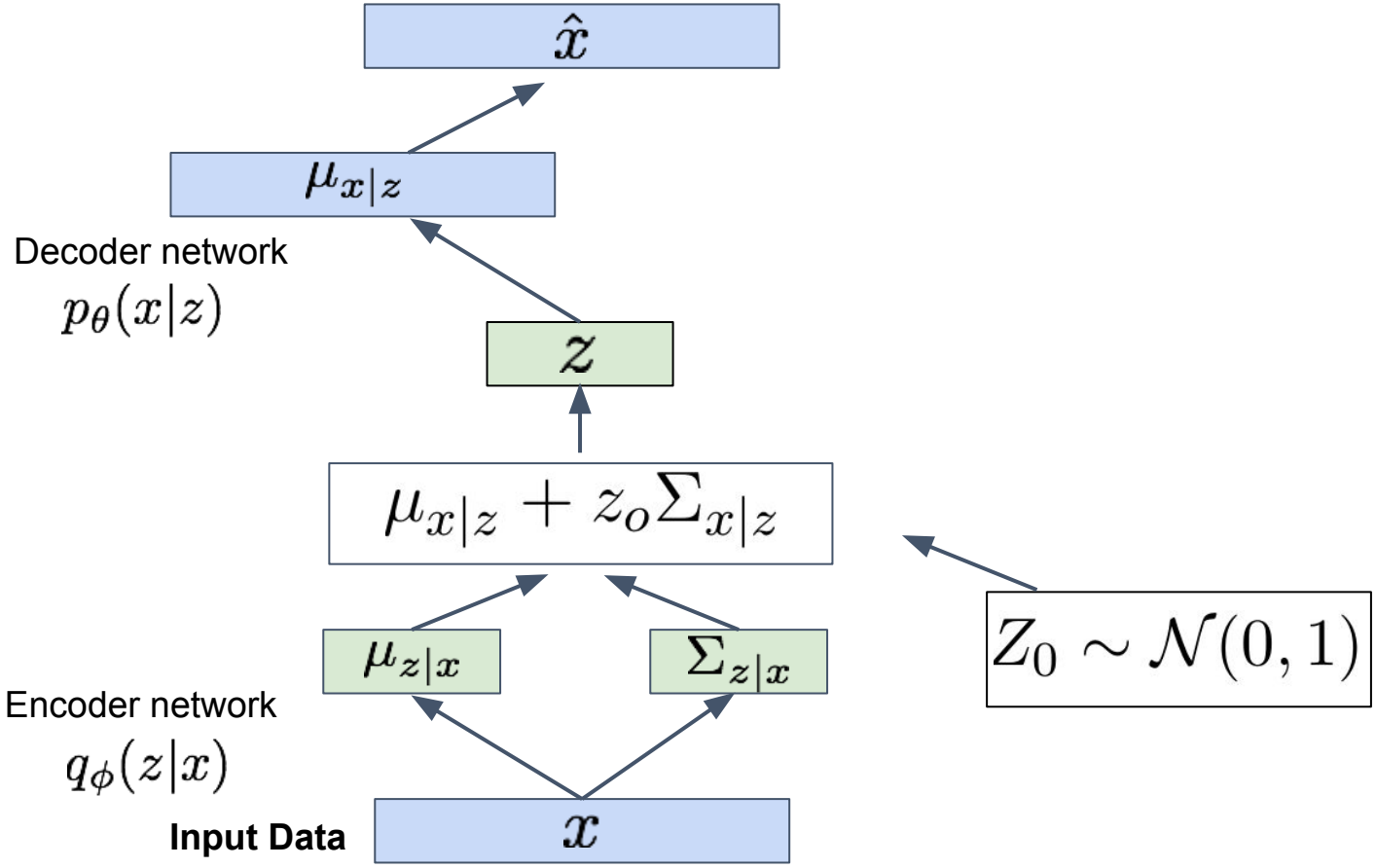
Re-parametrization trick:

If $Z_0 \sim \mathcal{N}(0,1)$, then $z = \mu_{x|z} + z_o \Sigma_{x|z} \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

Variational Autoencoders

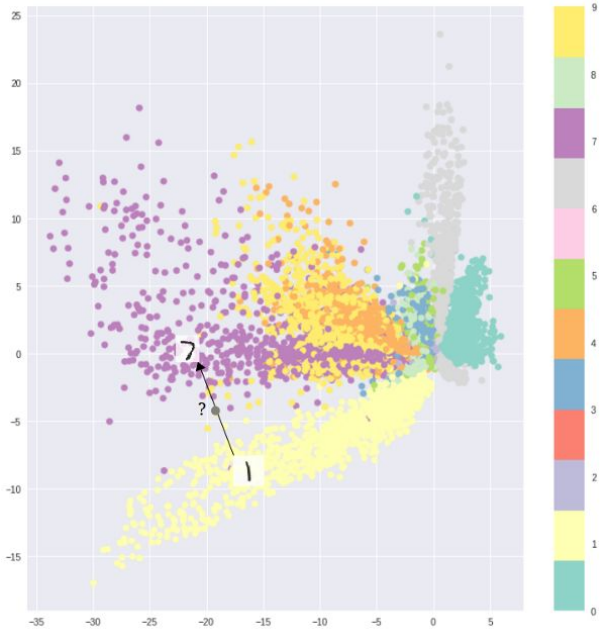


Variational Autoencoders

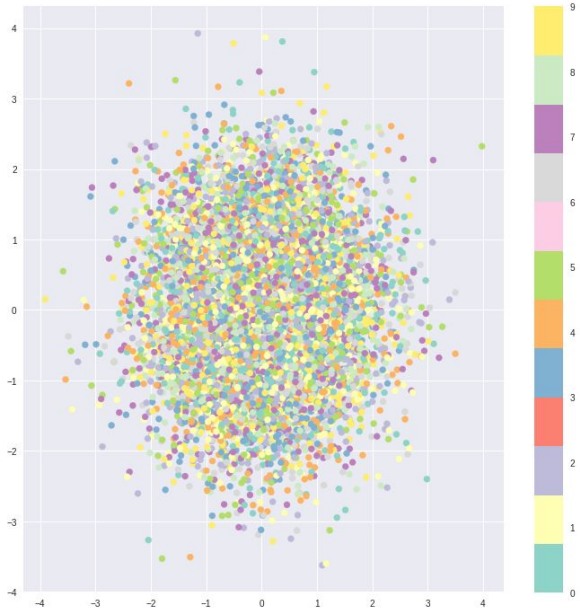


Variational Autoencoders

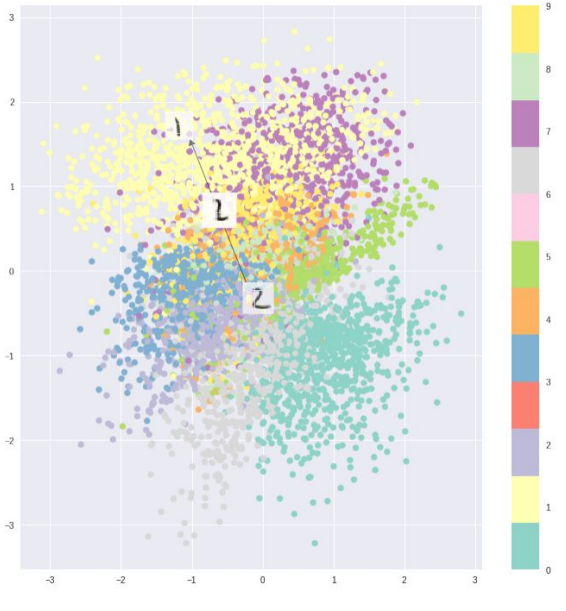
Autoencoders vs Variational Autoencoders



AE (reconstruction)



Only KL term

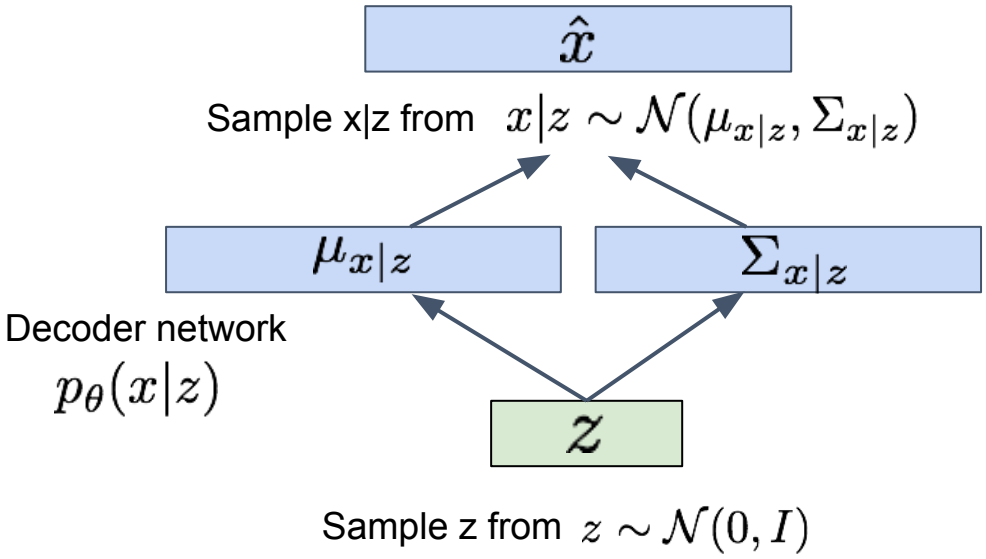


VAE (reconstruction+KL)

https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

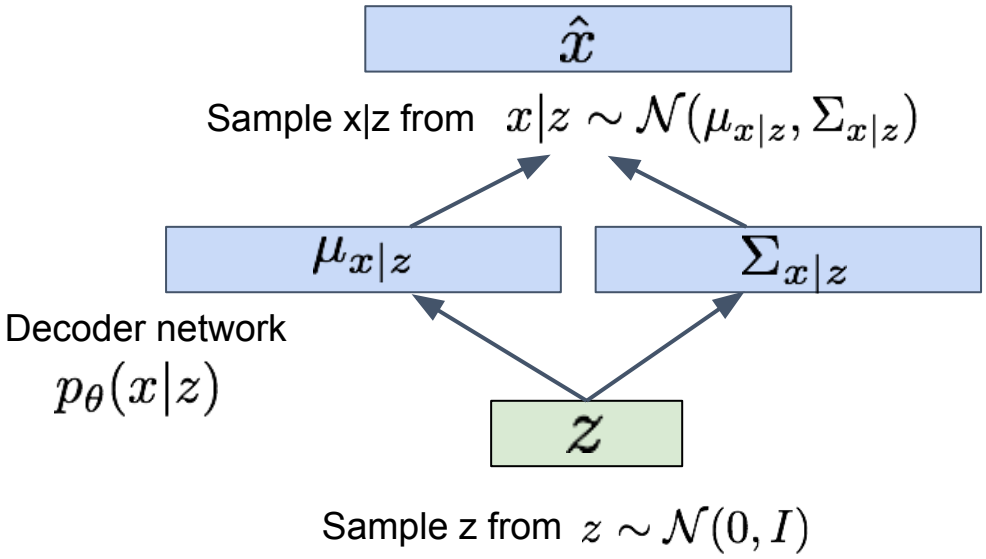
Generating Data

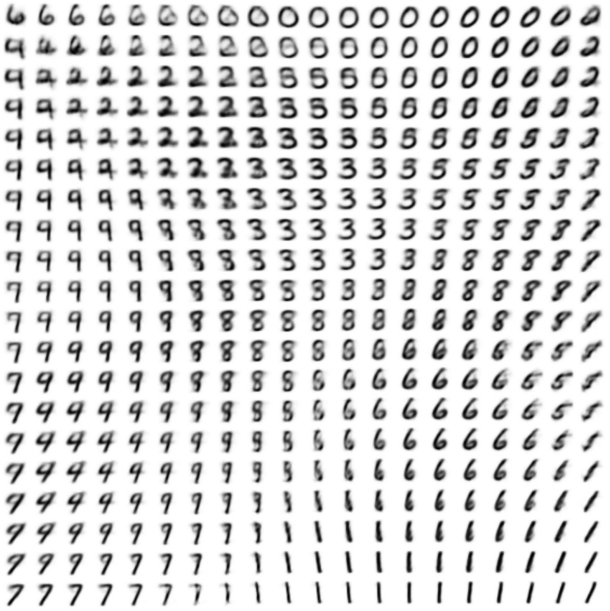
Use decoder network. Now sample z from prior!



Generating Data

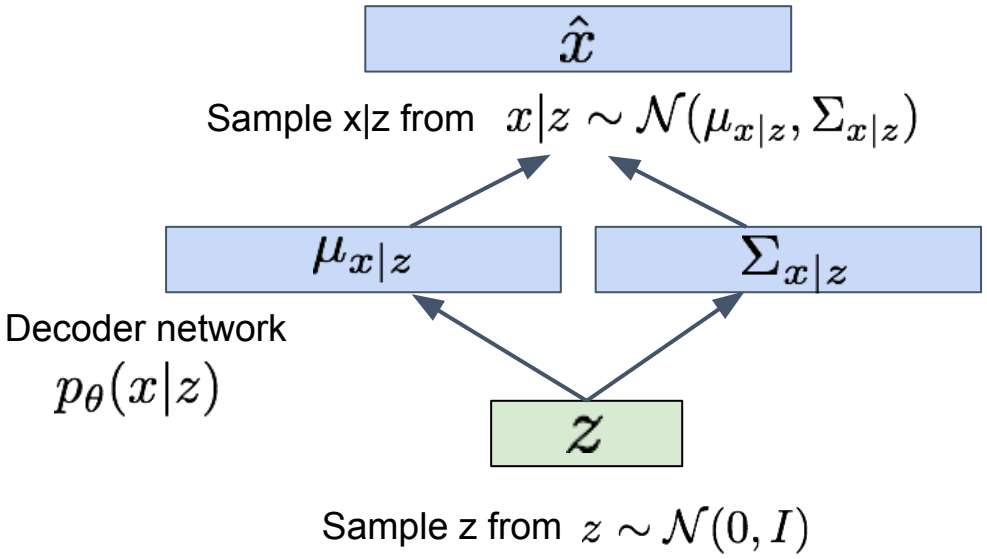
Use decoder network. Now sample z from prior!



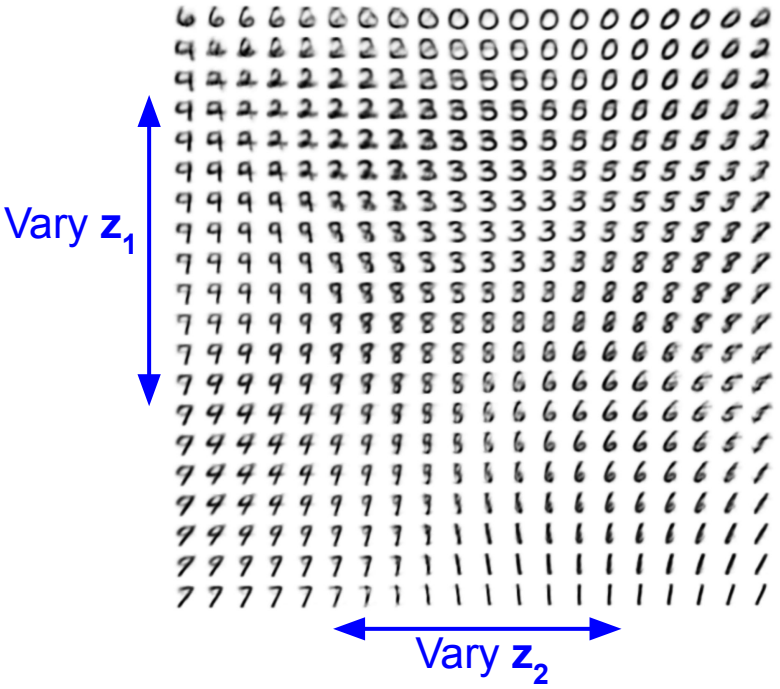


Generating Data

Use decoder network. Now sample z from prior!



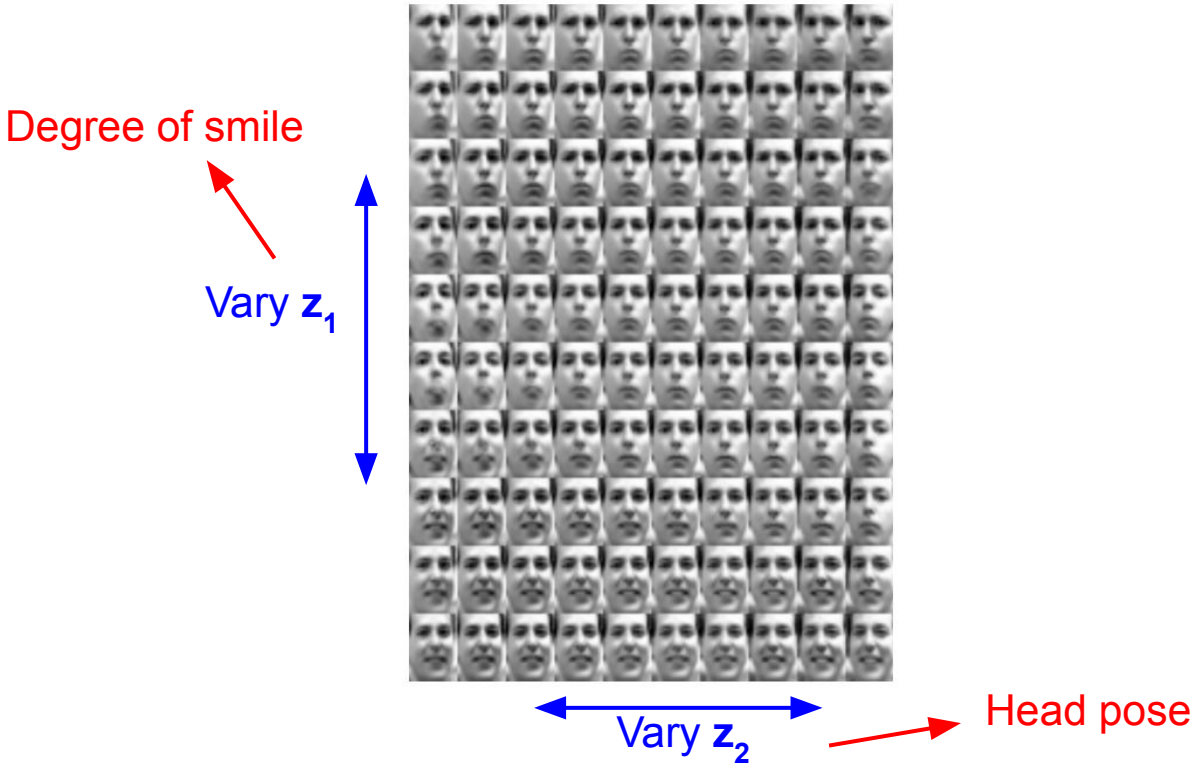
Data manifold for 2-d **z**



Generating Data

Diagonal prior on **z** => independent latent variables

Different dimensions of **z** encode interpretable factors of variation



Generating Data

Diagonal prior on **z**
=> independent
latent variables

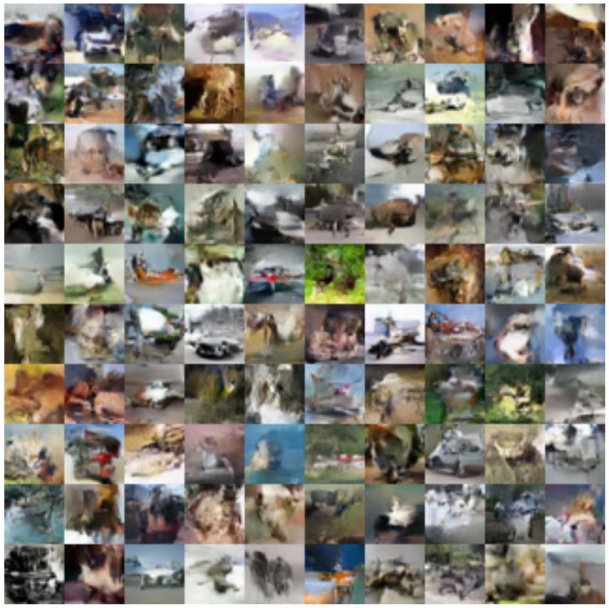
Different
dimensions of **z**
encode
interpretable factors
of variation

Also good feature representation that can be computed using $q_\phi(z|x)$!

http://www.dpkingma.com/sgvb_mnist_demo/demo.html

Generating Data: interpolation

Generating Data

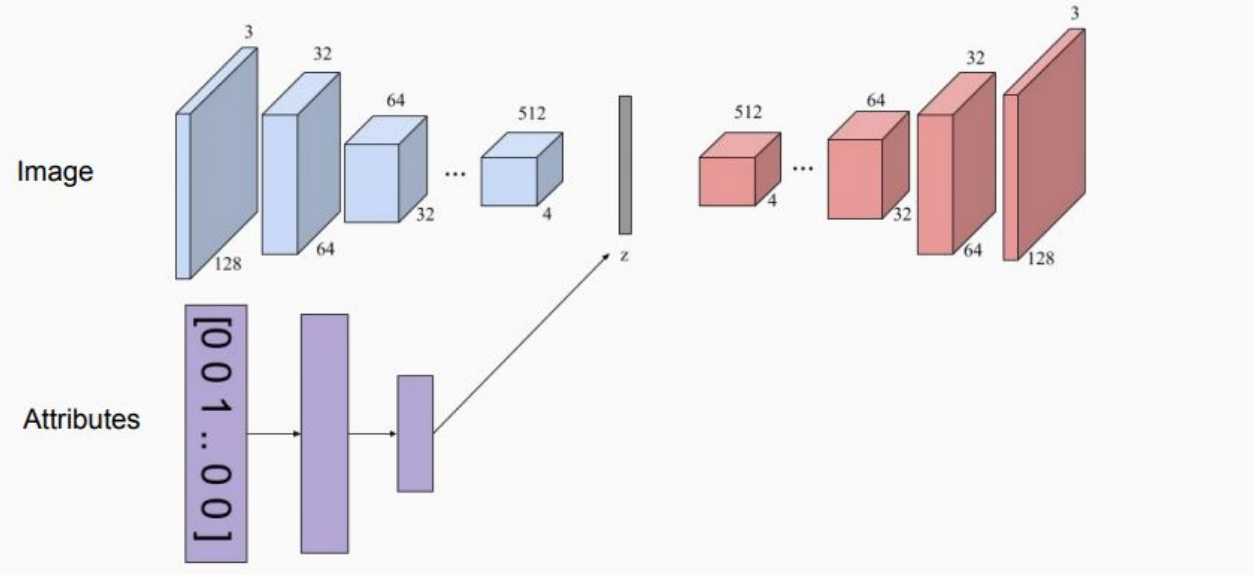


32x32 CIFAR-10

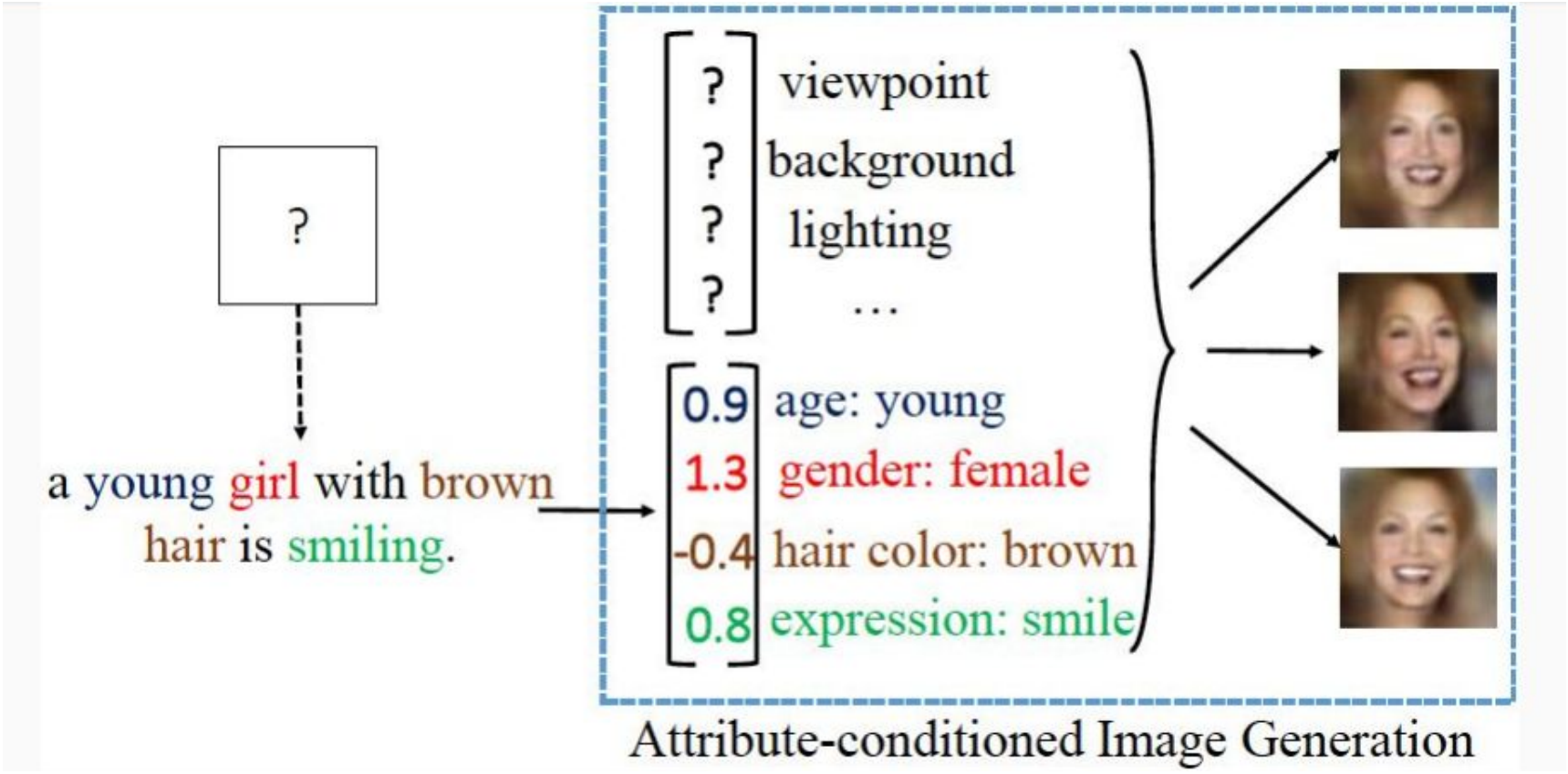
Labeled Faces in the Wild

Conditional VAE

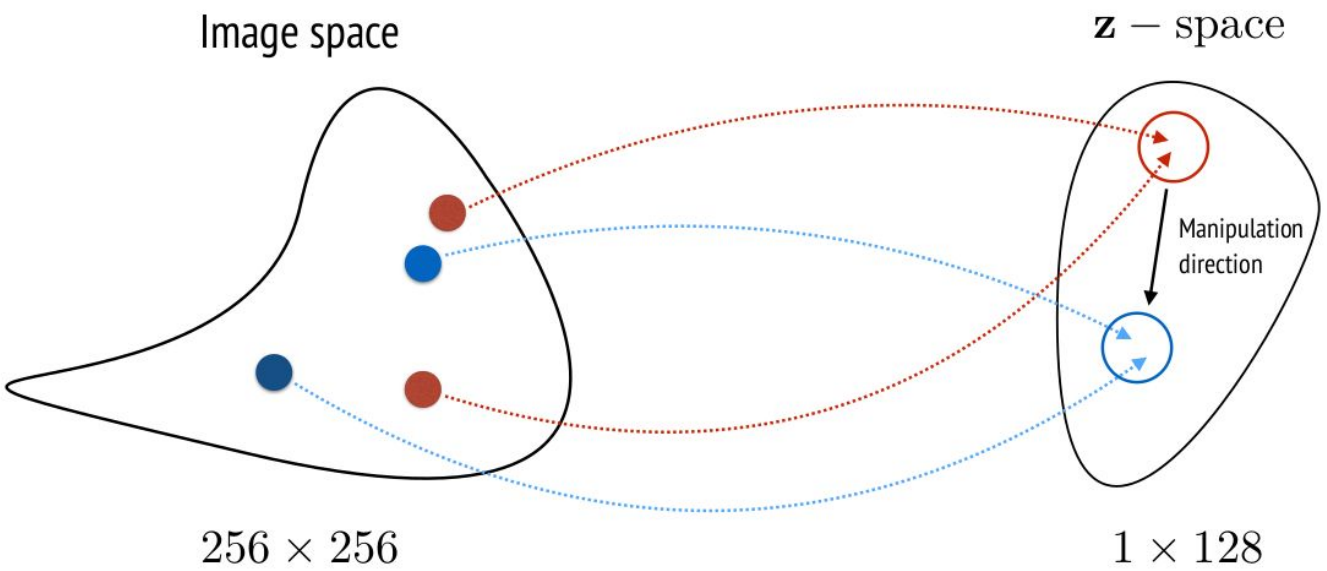
- What if we have labels? (e.g. digit labels or attributes) or other inputs we wish to condition on?
- None of the derivation changes.
- Replace all $p(x|z)$ with $p(x|z,y)$.
- Replace all $q(x|z)$ with $q(x|z,y)$.
- Go through the same KL divergence procedure, to get the same lower bound.

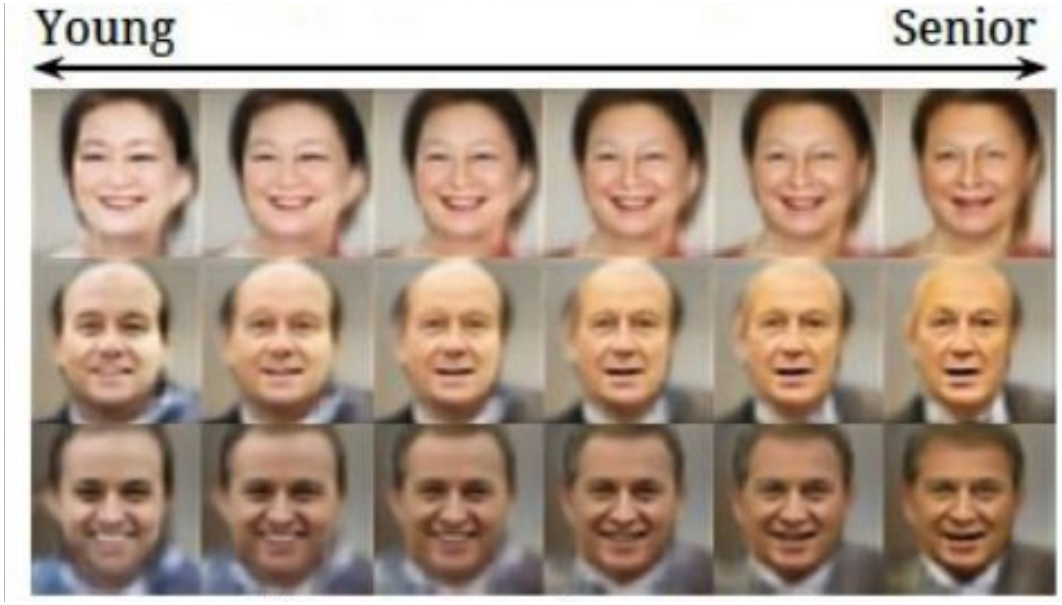


Generating Data: attribute-driven

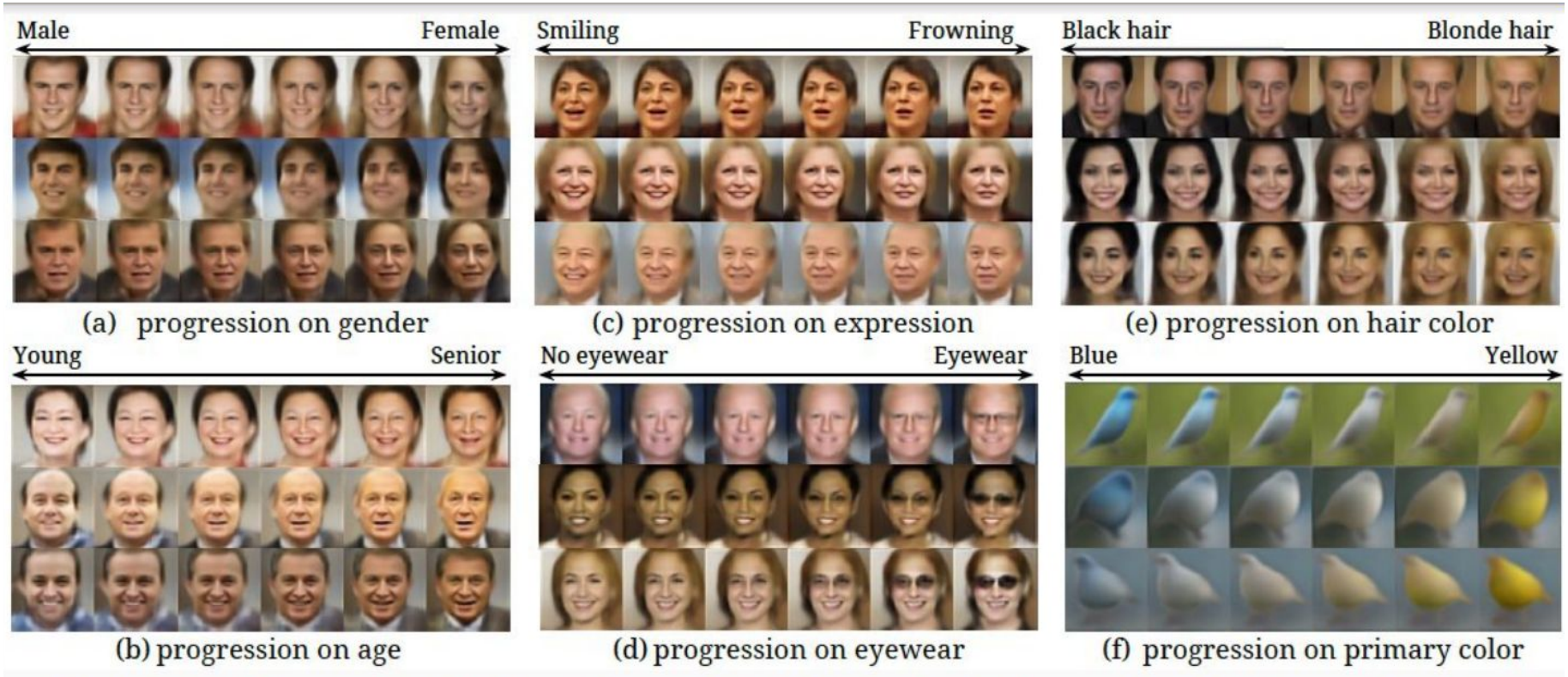


Generating Data: attribute-driven





Generating Data: attribute-driven



Variational Autoencoders

- Probabilistic spin to traditional autoencoders => allows generating data
- Defines an intractable density => derive and optimize a (variational) lower bound

- **Pros:**
  - Principled approach to generative models
  - Allows inference of q(z|x), can be useful feature representation for other tasks

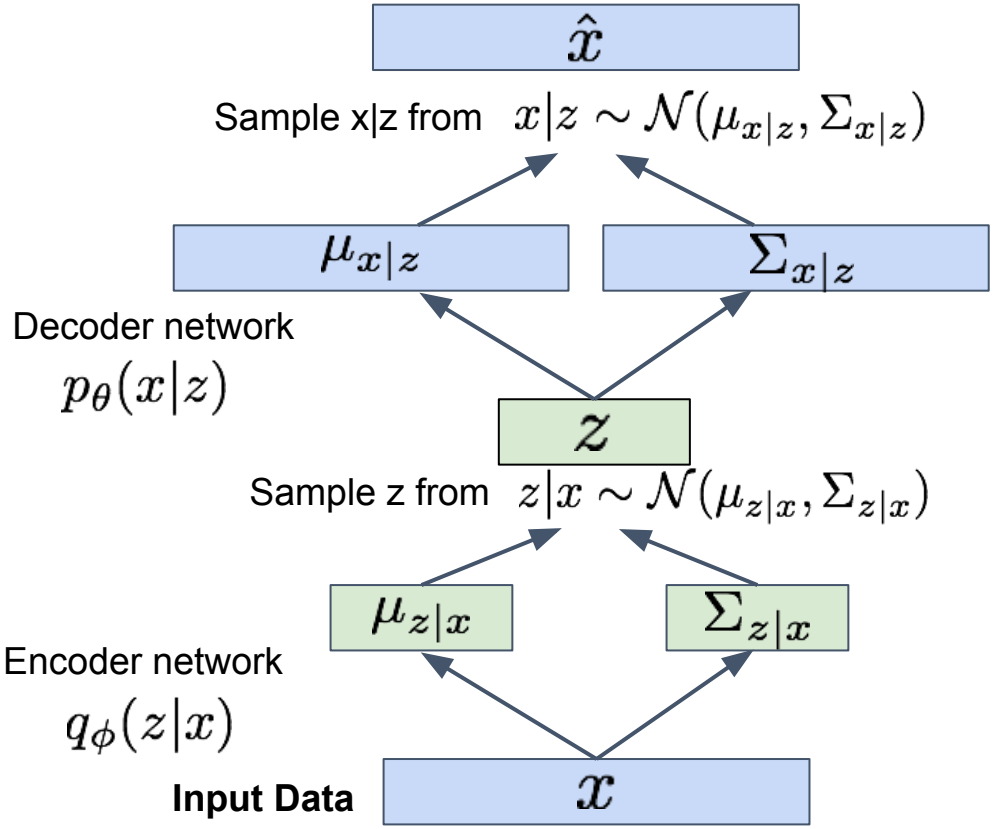
- **Cons:**
  - Maximizes lower bound of likelihood: okay, but not as good evaluation as PixelRNN/PixelCNN
  - Samples blurrier and lower quality compared to state-of-the-art (GANs)

- **Active areas of research:**
  - More flexible approximations, e.g. richer approximate posterior instead of diagonal Gaussian
  - Incorporating structure in latent variables

Variational Autoencoders: recap

Variational Autoencoders: recap

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$



Useful Links

- D. Kingma, M. Welling, *Auto-Encoding Variational Bayes*, ICLR, 2014
- Carl Doersch, *Tutorial on Variational Autoencoders* arXiv, 2016
- Xinchen Yan, Jimei Yang, Kihyuk Sohn, Honglak Lee, *Attribute2Image: Conditional Image Generation from Visual Attributes,* ECCV, 2016
- Jacob Walker, Carl Doersch, Abhinav Gupta, Martial Hebert, *An Uncertain Future: Forecasting from Static Images using Variational Autoencoders*, ECCV, 2016
- Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, Ole Winther, *Autoencoding beyond pixels using a learned similarity metric*, ICML, 2016
- Aditya Deshpande, Jiajun Lu, Mao-Chuang Yeh, David Forsyth, *Learning Diverse Image Colorization*, arXiv, 2016
- Raymond Yeh, Ziwei Liu, Dan B Goldman, Aseem Agarwala, *Semantic Facial Expression Editing using Autoencoded Flow*, arXiv, 2016