Generative Models

An Introduction to Variational Autoencoders and their Extensions

Omar Mahmood - March 2022

Outline

- Introduction to generative models
- Variational autoencoder
- Supervised (conditional) variational autoencoder
- Semi-supervised variational autoencoder

Introduction to generative models

Probabilistic models

Setup

- In ML, we assume datapoints x and labels y are generated from an underlying true probabilistic model
- goal is to build probabilistic model that is close to the true model
- Joint distribution $p(x, y) = p(x)p(y \mid x)$

Discriminative vs generative models

- Two types of probabilistic models
- Discriminative models model $p(y \mid x)$ directly, ignore $p(x)$
  - logistic regression, SVMs, kNN, random forests, some neural nets
- Generative models model $p(x)$ ($p(x, y)$ for supervised learning)
  - Naive Bayes, Hidden Markov Models, Variational Autoencoders

Taxonomy of Generative Models

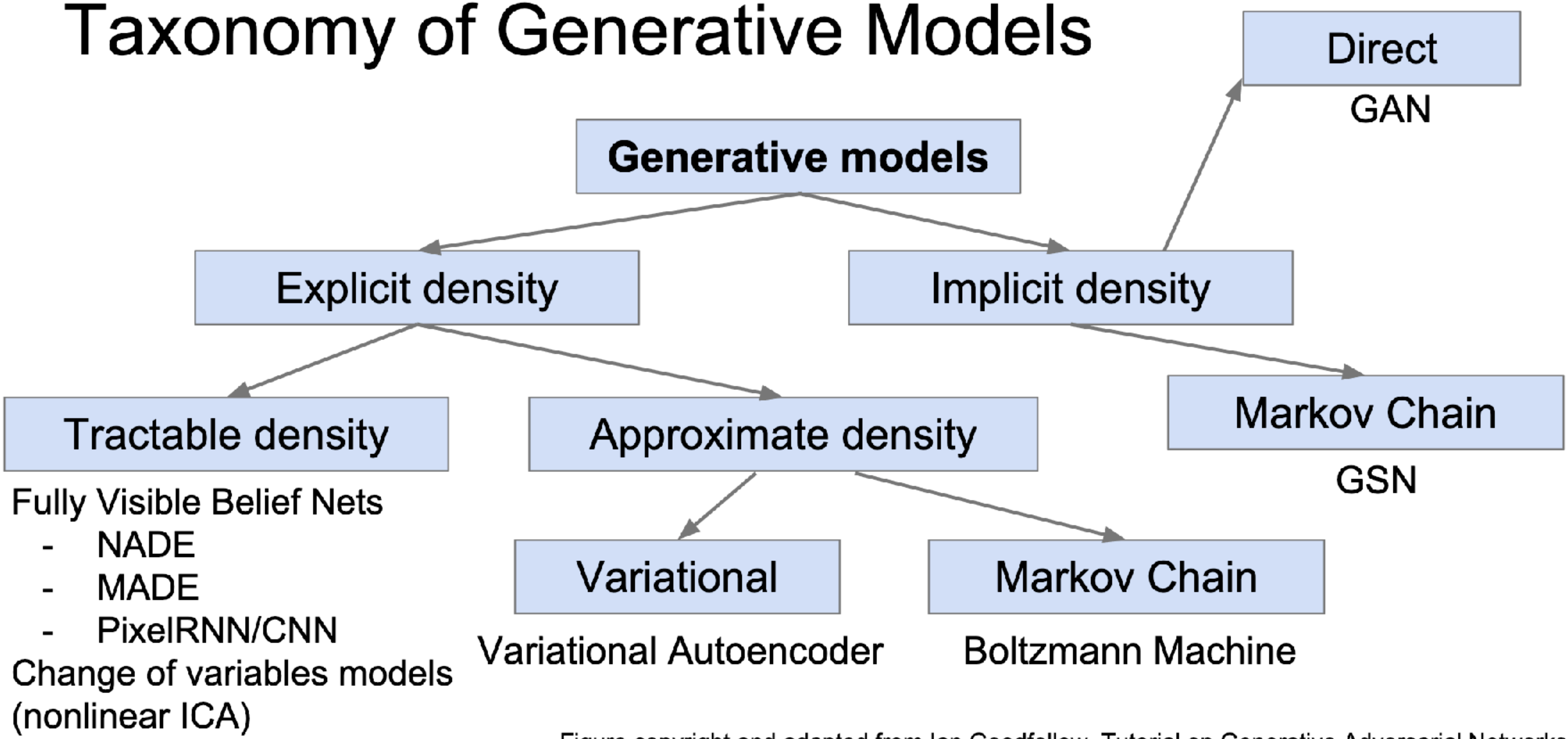


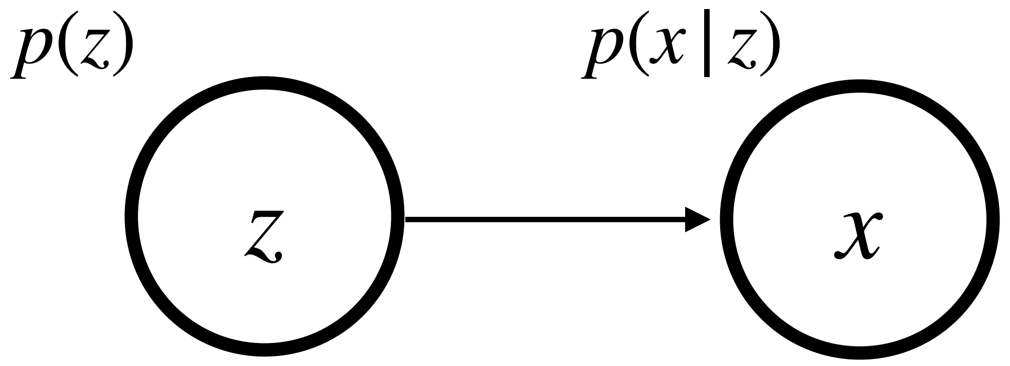
Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Variational Autoencoder (VAE)

Variational Autoencoder (VAE)

Setup

- Unsupervised latent variable model - hidden variable z, datapoint x



- Example: x is an image of a face, z represents facial features
- Choose $p(z)$ to be any simple distribution e.g. $\mathcal{N}(\mathbf{0}, \mathbf{I})$
- Model $p(x|z)$ using neural network with parameters θ -> $p_\theta(x|z)$
- Posterior $p(z|x)$ could also be of interest

Variational Autoencoder (VAE)

What is the problem?

- $p(x) = \int p_\theta(x \mid z) p(z) dz$ -> intractable
- Turns out posterior is also intractable: $p(z \mid x) = \dfrac{p_\theta(x \mid z) p(z)}{p(x)}$
- We can't even compute $p(x)$
  - How can we train a model?
  - How can we compute $p(z \mid x)$?

Variational Autoencoder (VAE)

Solution

- Kill two birds with one stone - train model and get approximation to $p(z \mid x)$
- Can use Markov Chain Monte Carlo (MCMC) or Variational Inference (VI)

Variational Autoencoder (VAE)

MCMC or VI?

- MCMC estimates gradient of $p(x)$ via samples from $p(z|x)$
  - expensive, unfeasible for large datasets
- VI assumes functional form for $p(z|x)$ -> $q(z|x)$
  - Approximate but efficient, scales well to large datasets
- VAEs use VI (hence the name ‘variational’)

Variational Autoencoder (VAE)

Using VI to solve our problem

- Parameterise $q(z|x)$ using a neural network with parameters ϕ -> $q_\phi(z|x)$
- VAE with encoder $q_\phi(z|x)$, decoder $p_\theta(x|z)$
- Still need a way to train the model - derivation follows

Variational Autoencoder (VAE)

Deriving the training objective

Marginalisation

$$p(x) = \int p(x|z)p(z)dz$$

$$= \int p(x|z)\frac{p(z)}{q(z|x)}q(z|x)dz$$

Definition of expectation

$$= \mathbb{E}_{z\sim q(z|x)}\left(p(x|z)\frac{p(z)}{q(z|x)}\right)$$

$$\log p(x) = \log\left[\mathbb{E}_{z\sim q(z|x)}\left(p(x|z)\frac{p(z)}{q(z|x)}\right)\right]$$

Variational Autoencoder (VAE)

Deriving the training objective

$$\log p(x) = \log \left[\mathbb{E}_{z \sim q(z|x)} \left(p(x\,|\,z) \frac{p(z)}{q(z\,|\,x)} \right) \right]$$

Jensen's inequality

$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)} \log \left[p(x\,|\,z) \frac{p(z)}{q(z\,|\,x)} \right]$$

$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)} \left[\log p(x\,|\,z) + \log p(z) - \log q(z\,|\,x) \right]$$

Definition

$$\text{ELBO} = \mathbb{E}_{z \sim q(z|x)} \left[\log p(x\,|\,z) + \log p(z) - \log q(z\,|\,x) \right]$$

Form usually used in VAEs

$$\text{ELBO} = \underbrace{\mathbb{E}_{z \sim q(z|x)} \log p(x\,|\,z)}_{\text{Reconstruction term}} \quad \underbrace{- D_{KL} \left(q(z\,|\,x) \,||\, p(z) \right)}_{\text{KL term}}$$

Variational Autoencoder (VAE)

Training objective

$$\arg\max_{\theta,\phi} \text{ELBO} = \arg\max_{\theta,\phi} \left[\mathbb{E}_{z\sim q_\phi(z|x)} \log p_\theta(x\,|\,z) - D_{KL}\left(\log q_\phi(z\,|\,x)\,||\,p(z) \right) \right]$$

- To train, maximise ELBO instead of $\log p(x)$

Variational Autoencoder (VAE)

Maximising ELBO gives good posterior

- Can show that:
  - $\text{ELBO} = \log p(x) - D_{KL}\left(q_\phi(z|x)||p(z|x)\right)$
  - KL term always nonnegative - missing term in the lower bound equation

$$
\begin{aligned}
\arg\max_\phi \ \text{ELBO} &= \arg\max_\phi \left[\log p(x) - D_{KL}\left(q_\phi(z|x)||p(z|x)\right)\right] \\
&= -\arg\max_\phi \ D_{KL}\left(q_\phi(z|x)||p(z|x)\right) \\
&= \arg\min_\phi \ D_{KL}\left(q_\phi(z|x)||p(z|x)\right)
\end{aligned}
$$

—> Maximising ELBO minimises KL divergence between true posterior and approximate posterior

Variational Autoencoder (VAE)

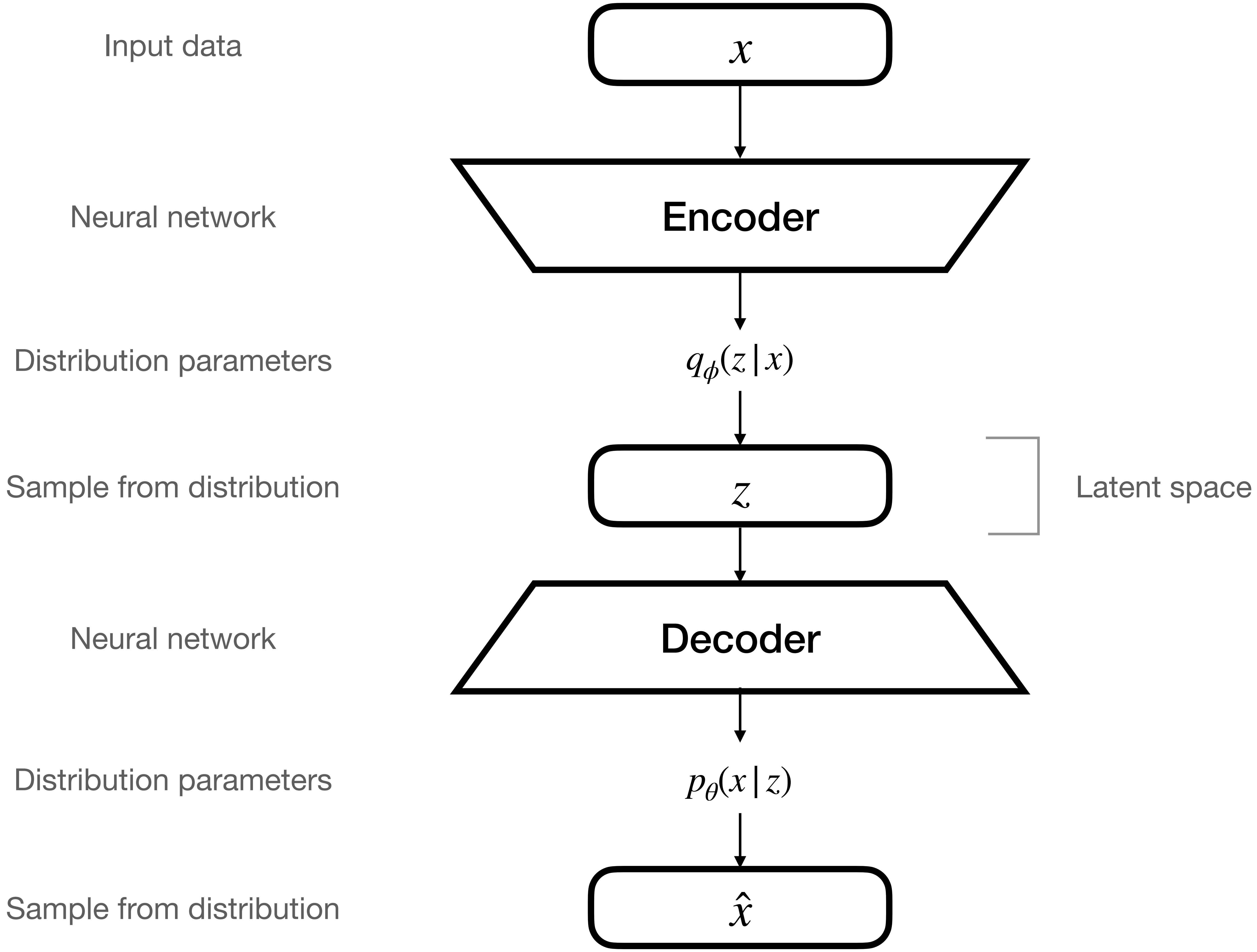
Training procedure

$$\text{ELBO} = \mathbb{E}_{z \sim q_\phi(z|x)} \log p_\theta(x \,|\, z) - D_{KL}\left(q_\phi(z \,|\, x) \,||\, p(z)\right)$$

- Sample z from $q_\phi(z \,|\, x)$ using datapoint x
- Compute $p_\theta(x \,|\, z)$
- Compute KL term analytically or via Simple Monte Carlo

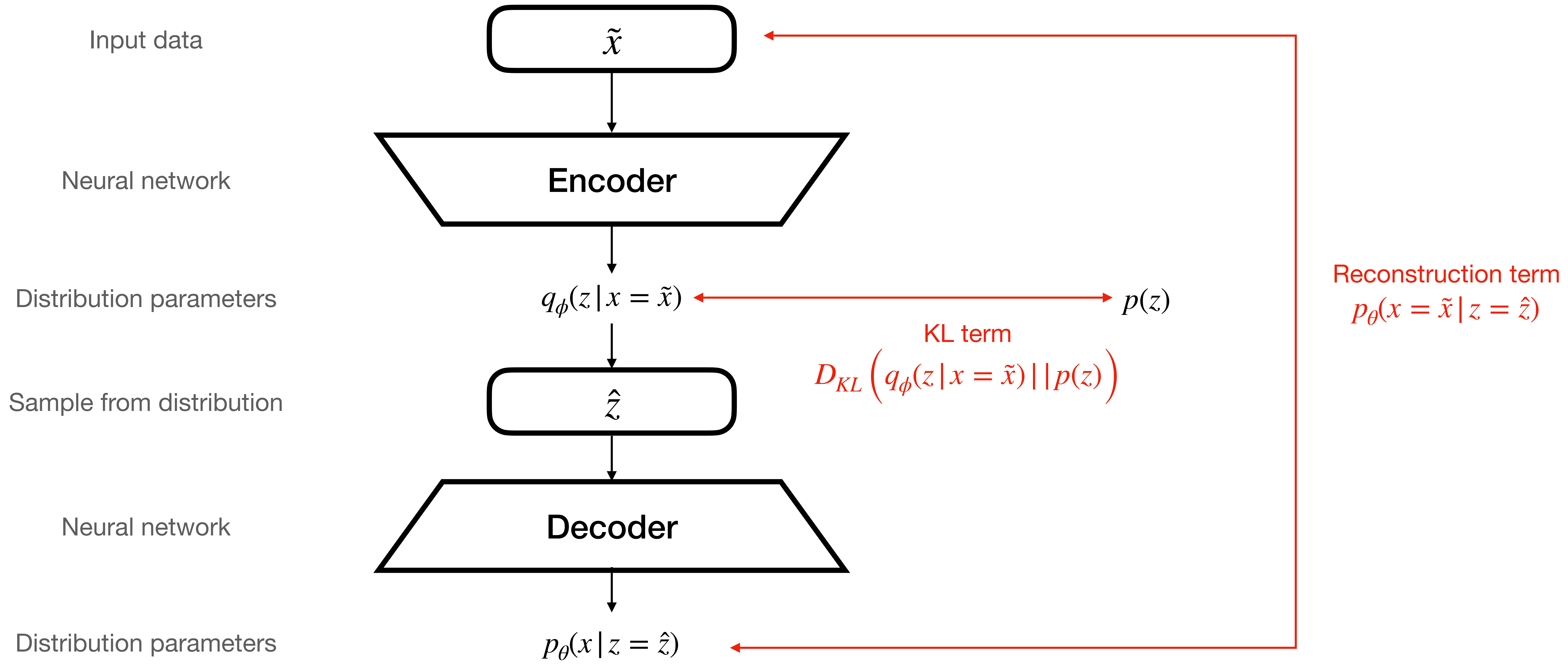
Variational Autoencoder (VAE)

What does this have to do with autoencoders?



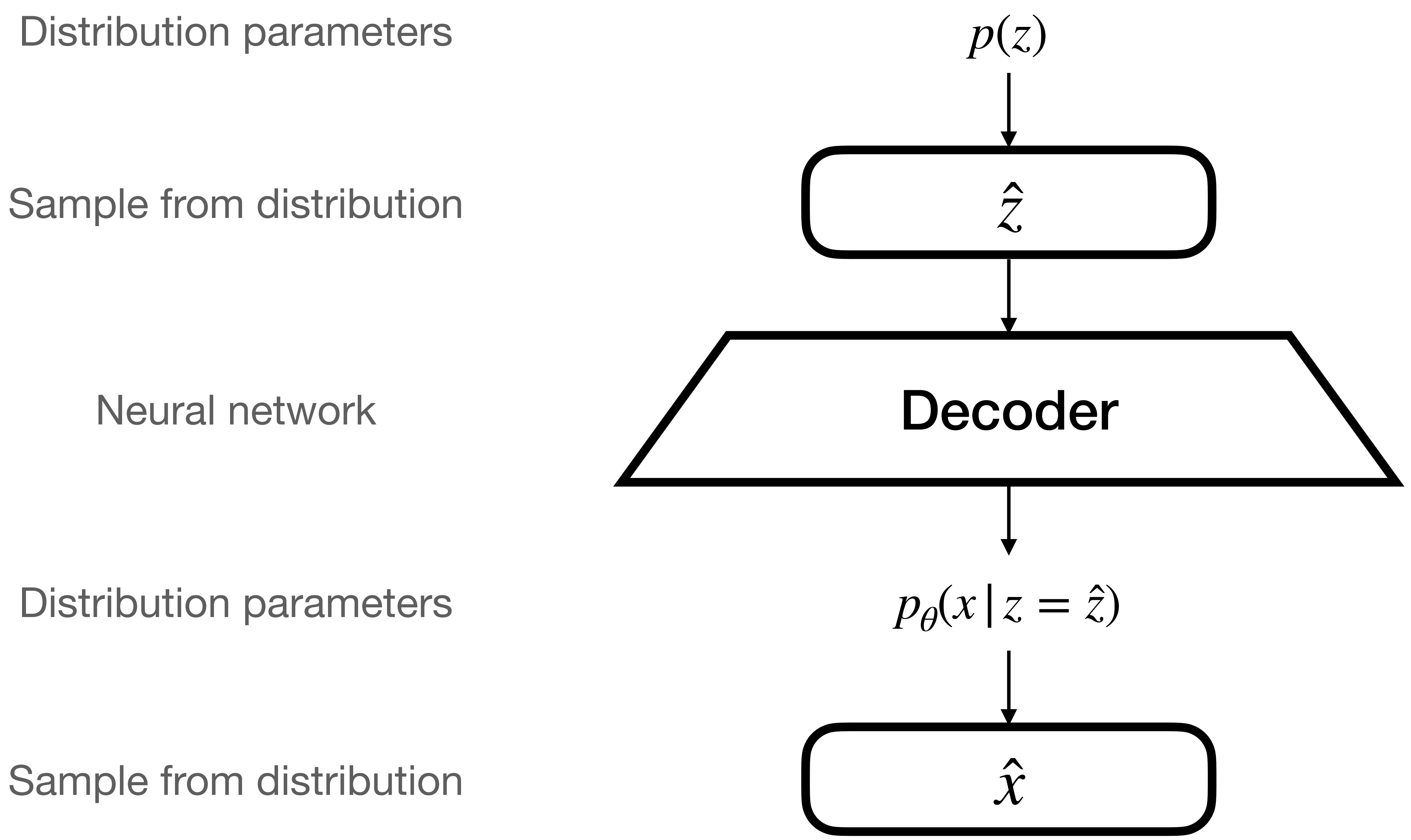
Variational Autoencoder (VAE)

What does training look like?



Variational Autoencoder (VAE)

What does the autoencoder look like at generation time?

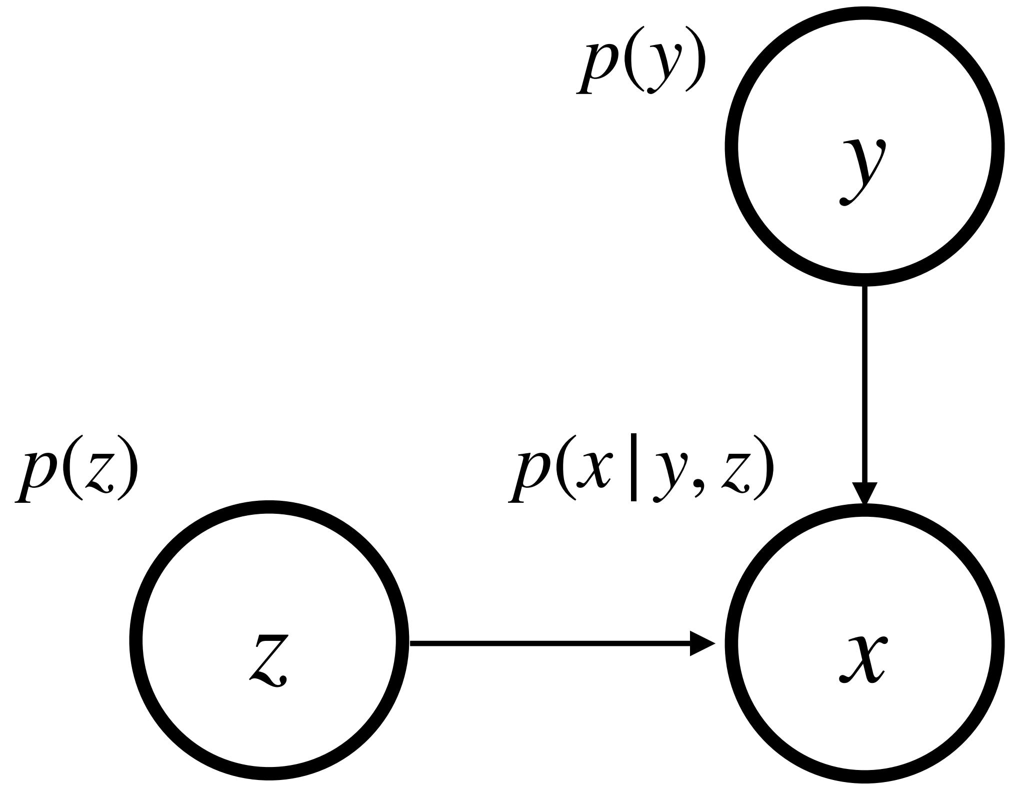


Conditional VAE (Supervised VAE)

Conditional (Supervised) VAE

What is a conditional (supervised) VAE?

- Extension of VAE to supervised setting
- Each datapoint x generated from latent variable z and label y -> $p(x \mid y, z)$
- Goal: build generative model $p(x, y)$



Conditional (Supervised) VAE

What does the training objective look like?

- Condition data generation on y

$$\log p(x) \geq \text{ELBO}(x)$$

$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)} \left[\log p(x \mid z) + \log p(z) - \log q(z \mid x)\right]$$

Condition on y

$$\log p(x, y) \geq \mathbb{E}_{z \sim q(z|x,y)} \left[\log p(x \mid y, z) + \log p(y, z) - \log q(z \mid x, y)\right]$$

Independence of y and z

$$\log p(x, y) \geq \mathbb{E}_{z \sim q(z|x,y)} \left[\log p(x \mid y, z) + \log p(y) + \log p(z) - \log q(z \mid x, y)\right]$$

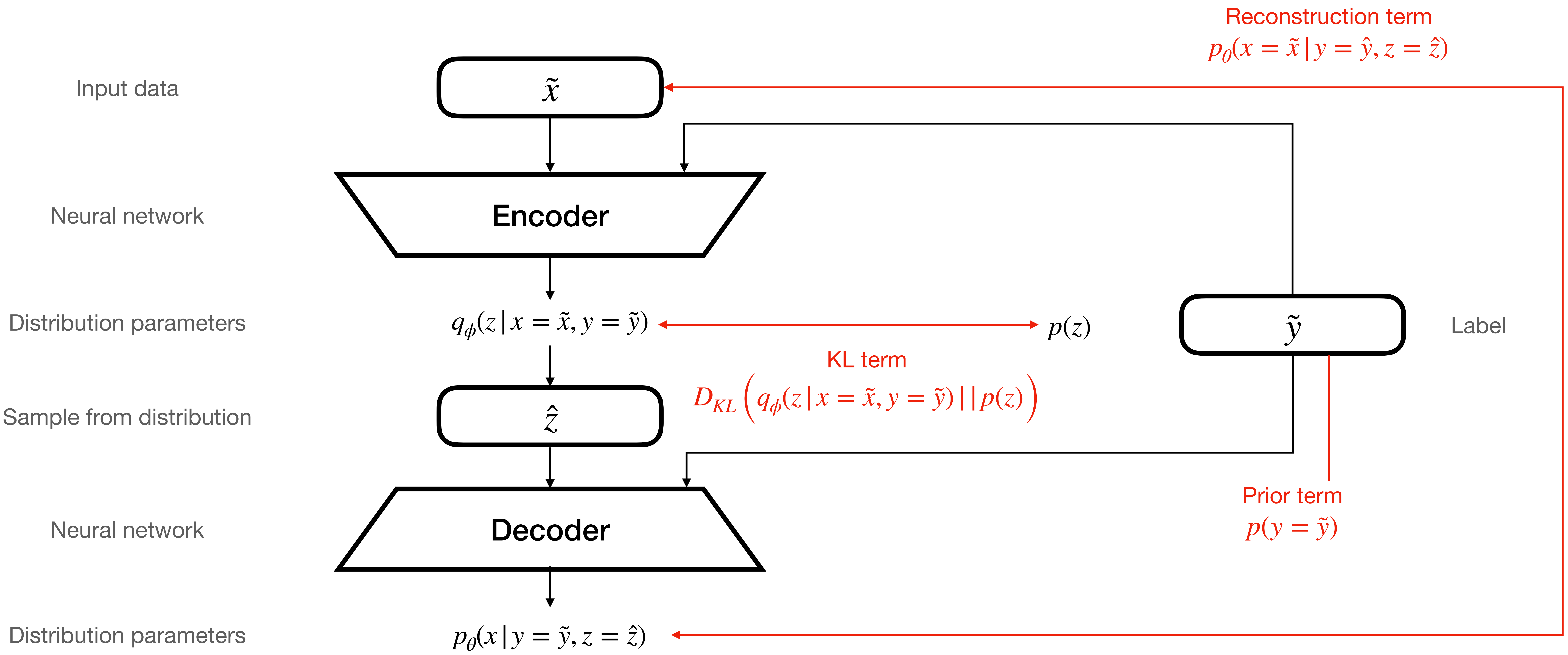
Express last two terms as KLD

$$\log p(x, y) \geq \mathbb{E}_{z \sim q(z|x,y)} \left[\log p(x \mid y, z)\right] + \log p(y) + D_{KL} \left(q(z \mid x, y) \mid\mid p(z)\right)$$

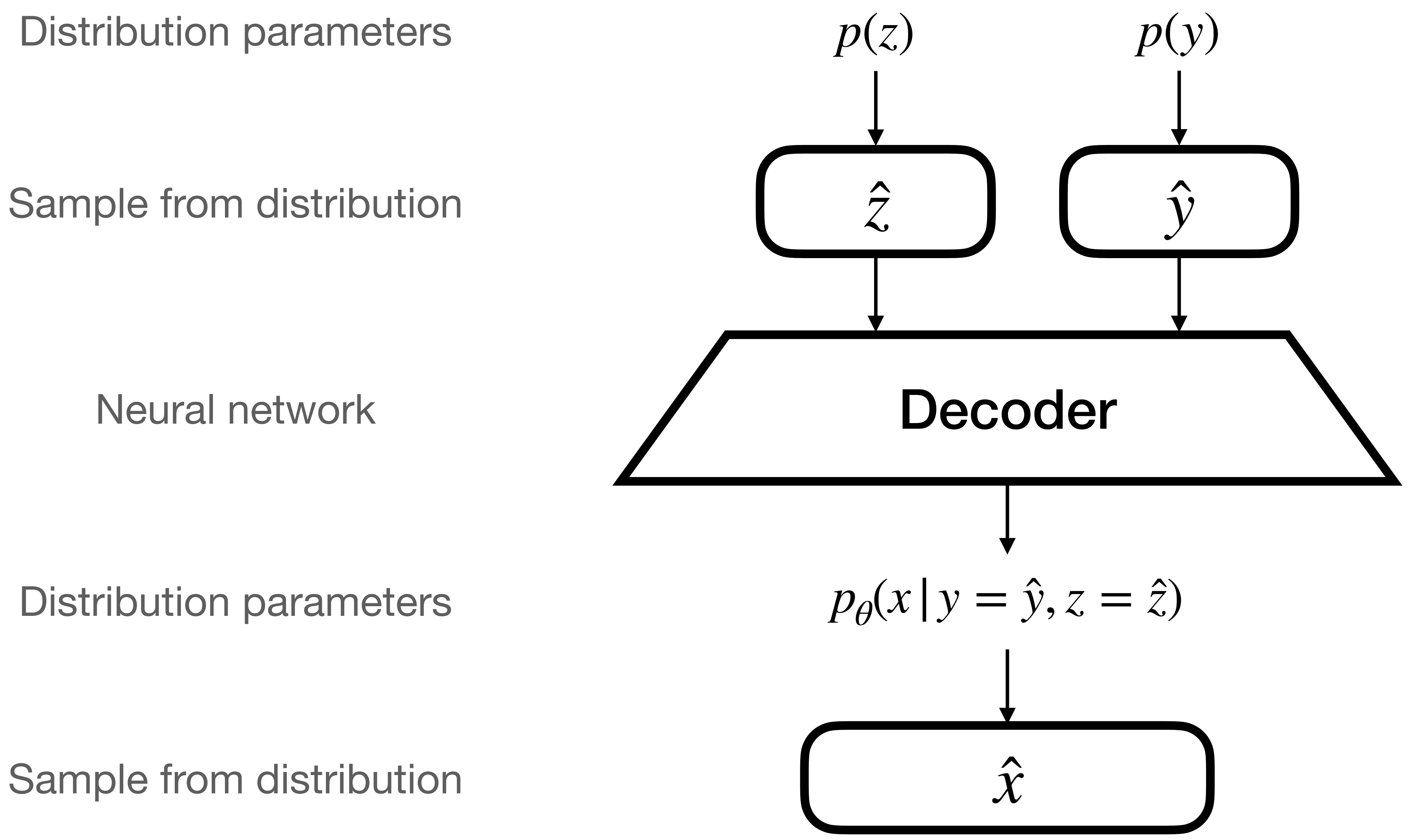
$$\log p(x, y) \geq \text{ELBO}(x, y)$$

Conditional (Supervised) VAE

What does training look like?



Conditional (Supervised) VAE
What does generation look like?



Semi-Supervised VAE

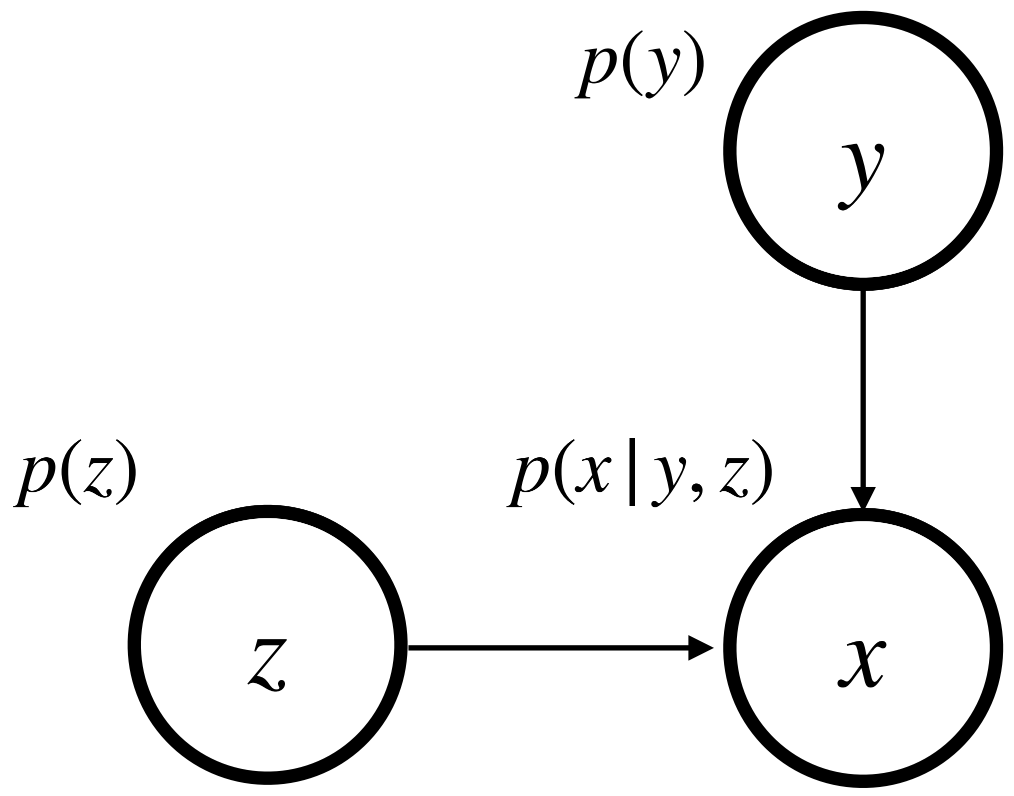
Semi-Supervised VAE

What is semi-supervision?

- Supervised learning - each data point has a label
- Unsupervised learning - no data points have labels
- Semi-supervised learning - small number of data points have labels

Semi-Supervised VAE

What is a semi-supervised VAE?



- Extension of VAE to semi-supervised setting
- For datapoints with labels:
  - Each datapoint x generated from latent variable z and label y -> $p(x \mid y, z)$
- For datapoints without labels:
  - Each datapoint x generated from latent variables z and y -> $p(x \mid y, z)$

Semi-Supervised VAE

What does the training objective look like?

- For data points with labels

$$\log p(x,y) \geq \mathbb{E}_{z\sim q(z|x,y)}\left[\log p(x\,|\,y,z)\right] + \log p(y) + D_{KL}\left(q(z\,|\,x,y)\,||\,p(z)\right) = -\mathscr{L}(x,y)$$

- For data points without labels

$$\log p(x) \geq \mathbb{E}_{y,z\sim q(y,z|x)}\left[\log p(x\,|\,y,z) + \log p(y) + \log p(z) - \log q(y,z\,|\,x)\right]$$

Definitions of joint and conditional probability

$$\log p(x) \geq \mathbb{E}_{y,z\sim q(y,z|x)}\left[\log p(x\,|\,y,z) + \log p(y) + \log p(z) - \log q(y\,|\,x) - \log q(z\,|\,x,y)\right]$$

$$\log p(x) \geq \mathbb{E}_{y\sim q(y|x),\ z\sim q(z|x,y)}\left[\log p(x\,|\,y,z)\right] + \mathbb{E}_{y\sim q(y|x)}\left[\log p(y) - \log q(y\,|\,x)\right] +$$
$$\mathbb{E}_{y\sim q(y|x),\ z\sim q(z|x,y)}\left[\log p(z) - \log q(z\,|\,x,y)\right]$$

$$\log p(x) \geq \mathbb{E}_{y\sim q(y|x),\ z\sim q(z|x,y)}\left[\log p(x\,|\,y,z)\right] - D_{KL}\left(q(y\,|\,x)\,||\,p(y)\right) + \mathbb{E}_{y\sim q(y|x)}\left[D_{KL}\left(q(z\,|\,x,y)\,||\,p(z)\right)\right] = -\mathscr{U}(x)$$

Semi-Supervised VAE

From training objective parts to model

- For data points with labels

$$\log p(x, y) \geq \mathbb{E}_{z \sim q(z|x,y)} \left[\log p(x \mid y, z)\right] + \log p(y) + D_{KL}\left(q(z \mid x, y) \mid\mid p(z)\right) = -\mathscr{L}(x, y)$$

- For data points without labels

$$\log p(x, y) \geq \mathbb{E}_{y \sim q(y|x),\ z \sim q(z|x,y)} \left[\log p(x \mid y, z)\right] - D_{KL}\left(q(y \mid x) \mid\mid p(y)\right) + \mathbb{E}_{y \sim q(y|x)} \left[D_{KL}\left(q(z \mid x, y) \mid\mid p(z)\right)\right] = -\mathscr{U}(x)$$

- $p_\theta(x \mid y, z)$: decoder
- $q_\phi(z \mid x, y)$: encoder
- $q_\phi(y \mid x)$: ‘predictor’

Semi-Supervised VAE

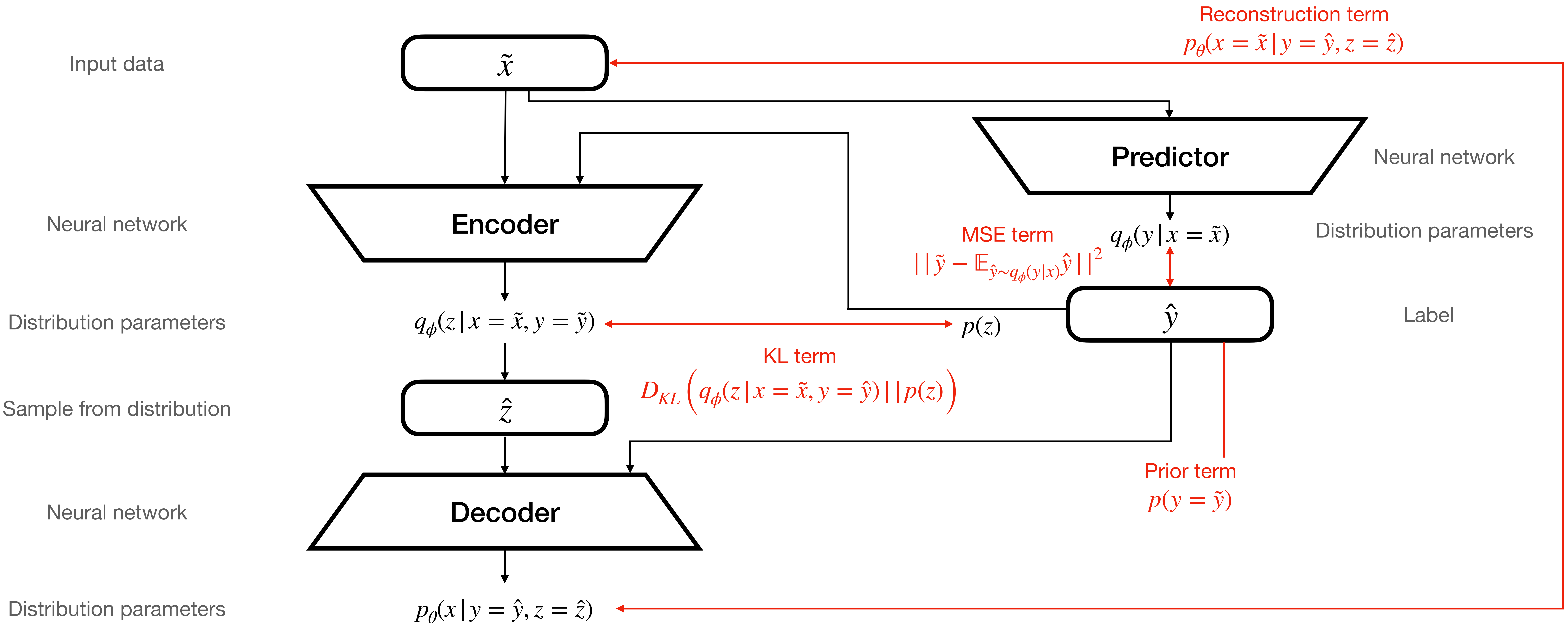
Overall training objective

$$\mathcal{J} = \sum_{x,y \sim p_l} \mathcal{L}(x,y) + \sum_{x \sim p_u} U(x) + \beta \sum_{x,y \sim p_l} ||y - \mathbb{E}_{\hat{y} \sim q_\phi(y|x)} \hat{y}||^2$$

- $\mathcal{J}$ is overall training objective (to be minimised)
- p_l for labelled data, p_u for unlabelled data
- Last term trains predictor to predict observed properties from labelled data
- β is hyperparameter, controls tradeoff between generative and discriminative learning

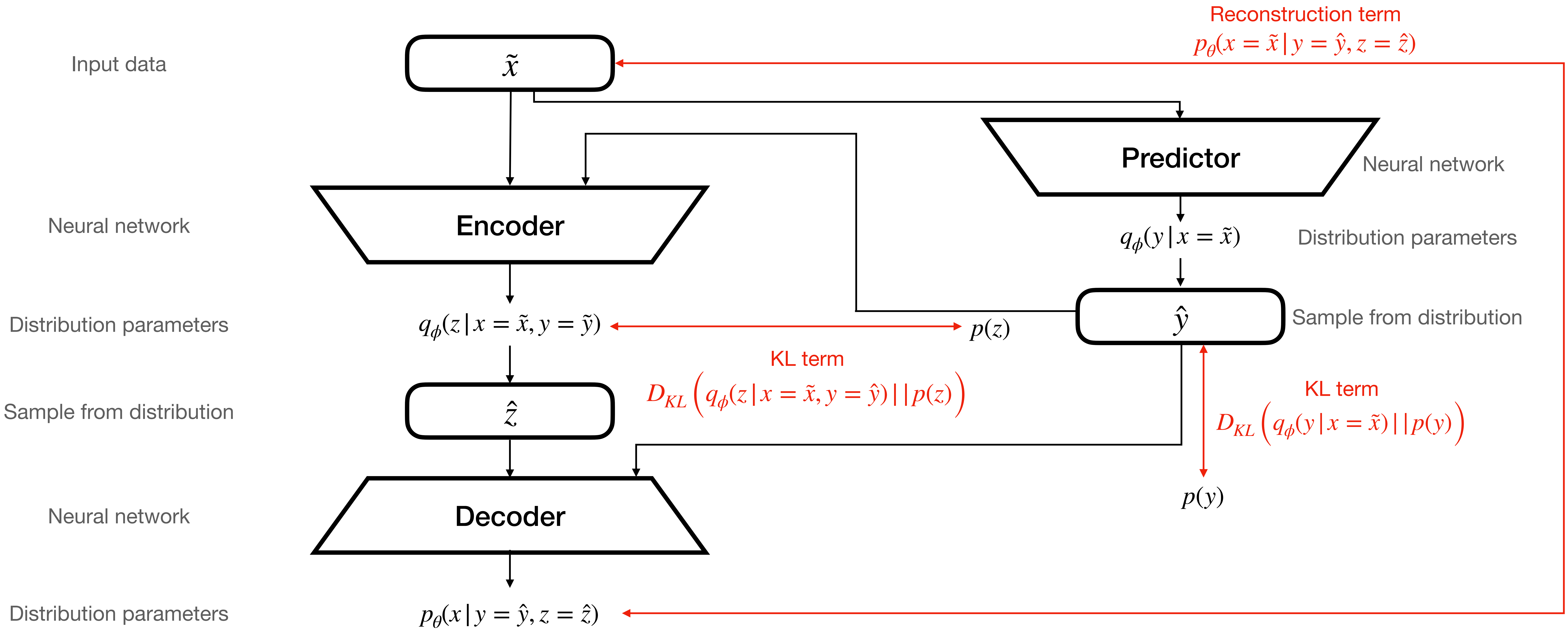
Semi-Supervised VAE

What does training look like for labelled data points?



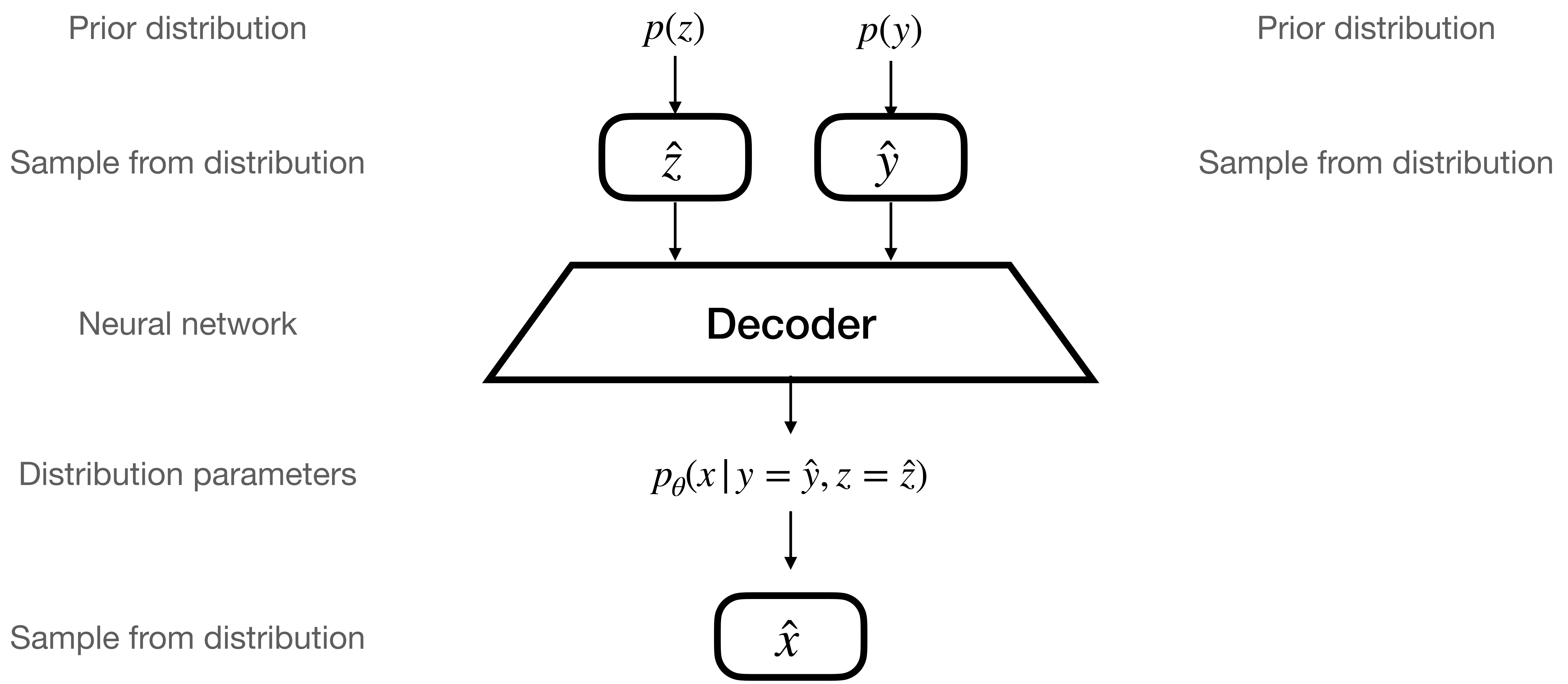
Semi-Supervised VAE

What does training look like for unlabelled data points?



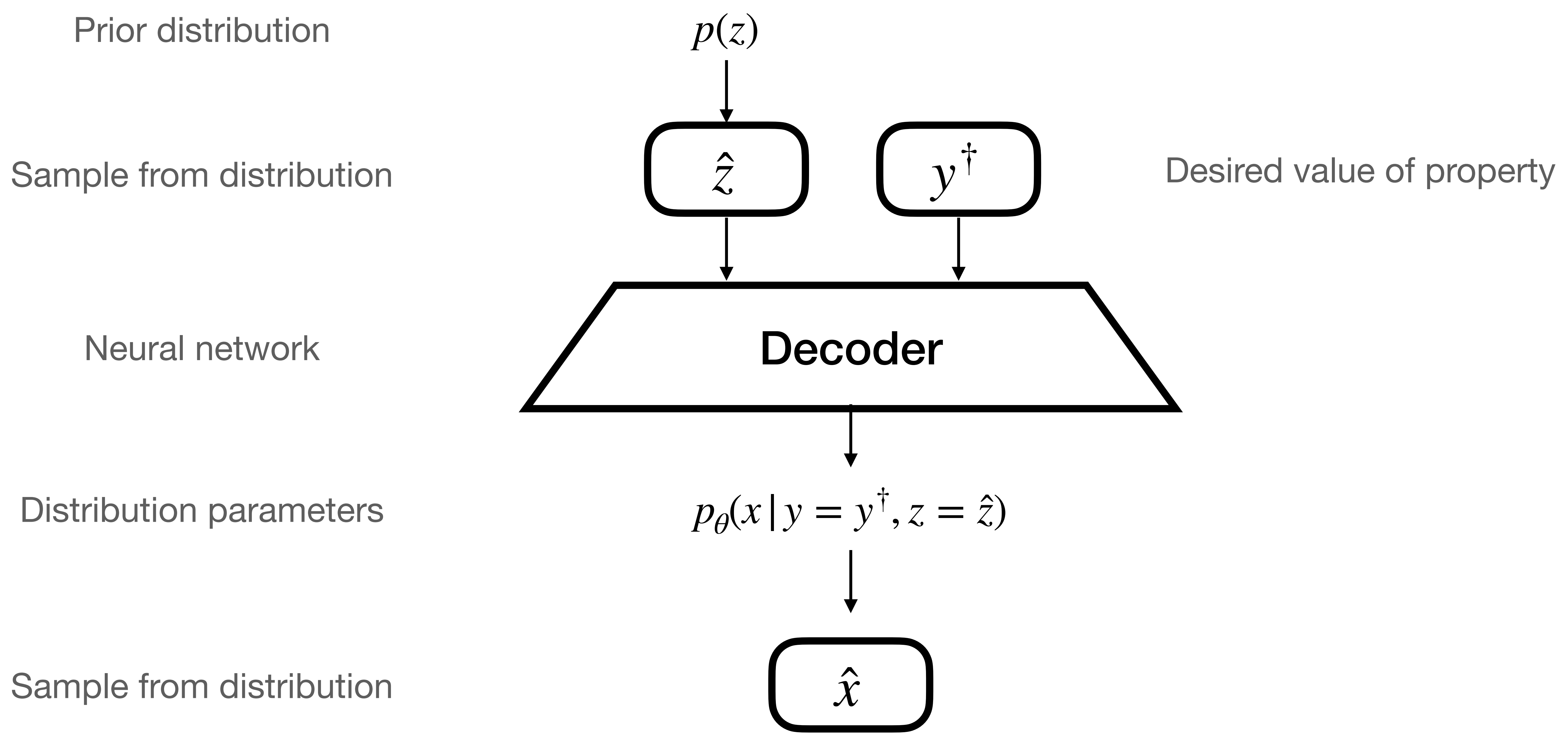
Semi-Supervised VAE

What does unconditional generation look like?



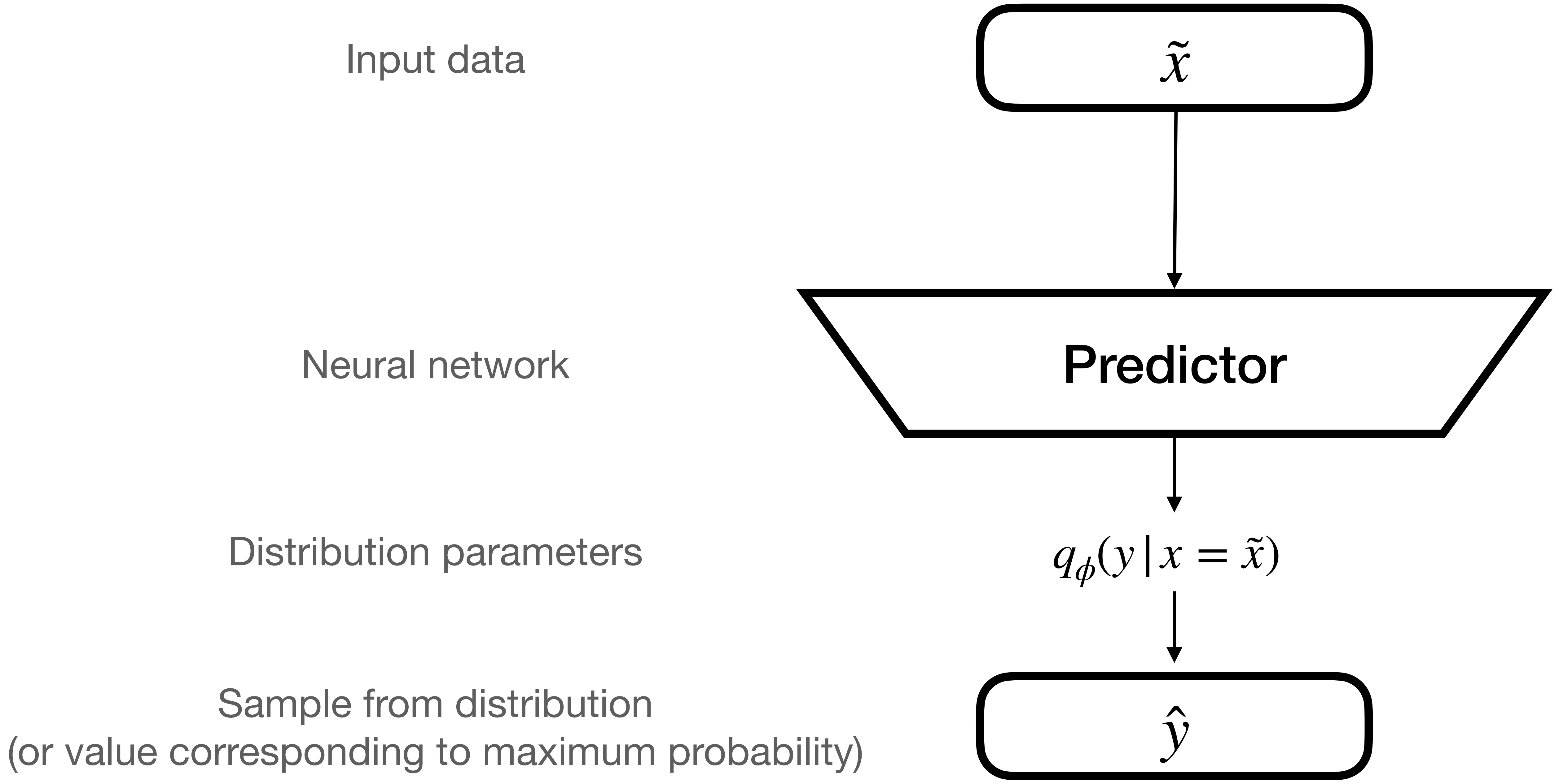
Semi-Supervised VAE

What does conditional generation look like?



Semi-Supervised VAE

What does property prediction look like?



Recap

Recap

- Generative models model data generating distribution $p(x)$
- For (semi-) supervised learning they model $p(x, y)$
- VAEs are a type of generative model
- Conditional VAEs extend VAEs to a supervised setting
- Semi-supervised VAEs extend VAEs and conditional VAEs to a semi-supervised setting

References

Kingma, D.P., & Welling, M. (2014). Auto-Encoding Variational Bayes. CoRR, abs/1312.6114.

Sohn, K., Lee, H., & Yan, X. (2015). Learning Structured Output Representation using Deep Conditional Generative Models. NIPS.

Kingma, D.P., Mohamed, S., Jimenez Rezende, D., & Welling, M. (2014). Semi-supervised Learning with Deep Generative Models. ArXiv, abs/1406.5298.

Kang, S., & Cho, K. (2019). Conditional molecular design with deep generative models. Journal of chemical information and modeling, 59 1, 43-52 .

Slide on "Taxonomy of Deep Generative Models" taken from Stanford course CS231n taught in 2017