TU-Delft Deep Learning course 2018-2019

08.Unsupervised

27 April 2019

Delft University of Technology

Lecturer: Jan van Gemert

Unsupervised learning

Book: Chapters 14, 20

- Q: What is unsupervised learning?

Unsupervised learning

Book: Chapters 14, 20

- Q: What is unsupervised learning?
  A: No ground truth label.

- Q: Why is that good?

Unsupervised learning

Book: Chapters 14, 20

- Q: What is unsupervised learning?
  A: No ground truth label.
- Q: Why is that good?
  A: Labels are often the most expensive to obtain
- Q: What is it good for?

Unsupervised learning

Book: Chapters 14, 20

- Q: What is unsupervised learning?
  A: No ground truth label.

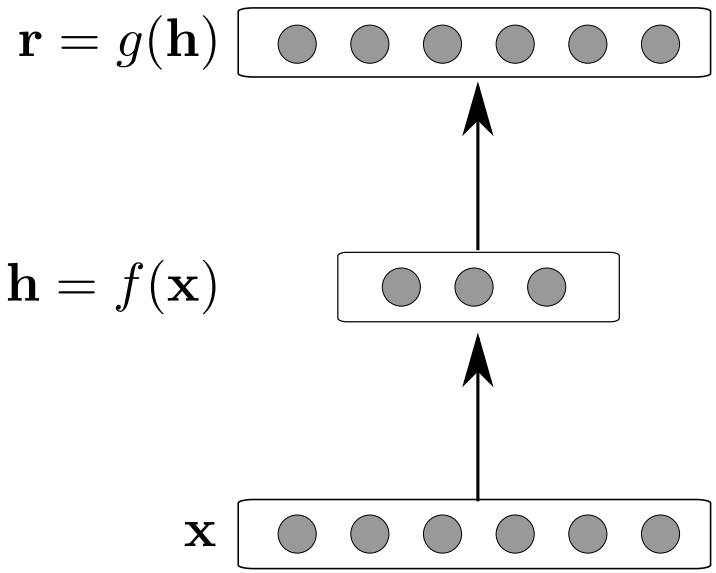
- Q: Why is that good?
  A: Labels are often the most expensive to obtain

- Q: What is it good for?
  A: Learn compression to store large datasets, pre-training for feature learning, density estimation, generating new data samples.

Autoencoder

Book: Chapter 14

Feed forward network to reproduce its input at the output layer; $g(f(x)) = x$



Decoder:

$$\begin{aligned} r &= g(h) \\ &= \sigma(c + W^* h) \end{aligned}$$

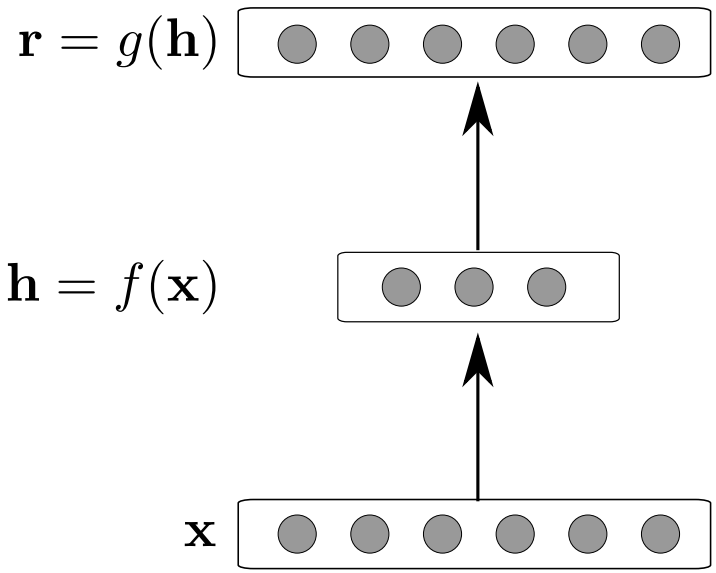
Encoder:

$$\begin{aligned} h &= f(x) \\ &= \sigma(b + Wx) \end{aligned}$$

Autoencoder

Book: Chapter 14

Feed forward network to reproduce its input at the output layer; $g(f(x)) = x$



Decoder:

$$\begin{aligned} r &= g(h) \\ &= \sigma(c + W^* h) \end{aligned}$$

Encoder:

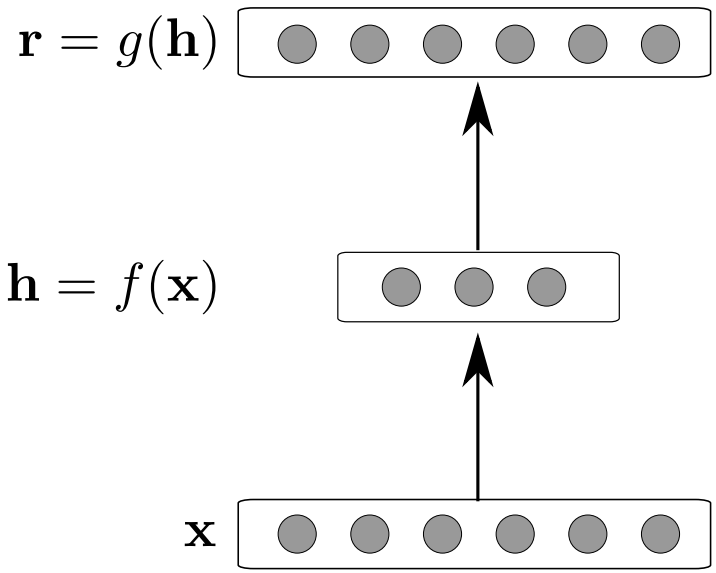
$$\begin{aligned} h &= f(x) \\ &= \sigma(b + Wx) \end{aligned}$$

Q: What loss function would you minimize to train $g(f(x)) = x$?

Autoencoder

Book: Chapter 14

Feed forward network to reproduce its input at the output layer; $g(f(x)) = x$



Decoder:

$$\begin{aligned} r &= g(h) \\ &= \sigma(c + W^* h) \end{aligned}$$

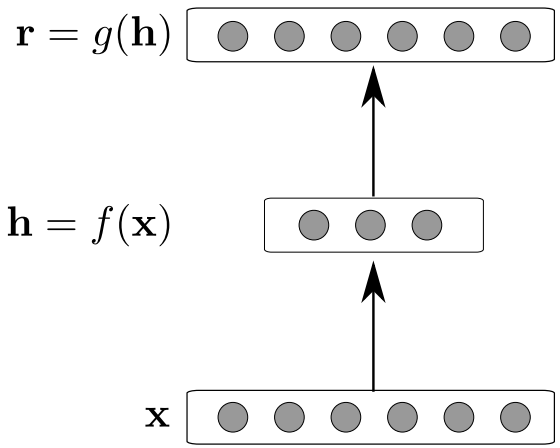
Encoder:

$$\begin{aligned} h &= f(x) \\ &= \sigma(b + Wx) \end{aligned}$$

Q: What loss function would you minimize to train $g(f(x)) = x$?
A: $\sum_i \frac{1}{2}(g(f(x_i)) - x_i)^2$ (if x is continue)

Size of the hidden (bottleneck) layer

Book: Chapter 14

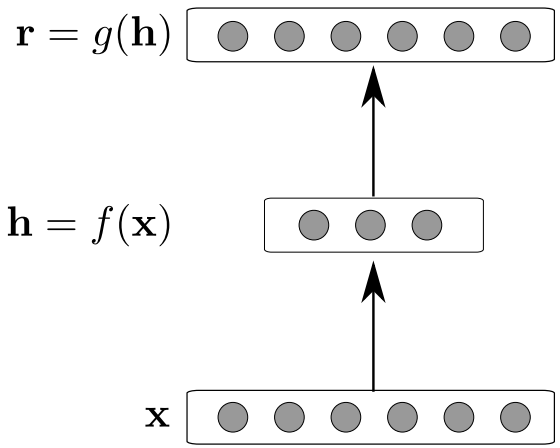


$h < x$, **Undercomplete** hidden layer:

- "Compress" the input
- Compresses well for training samples

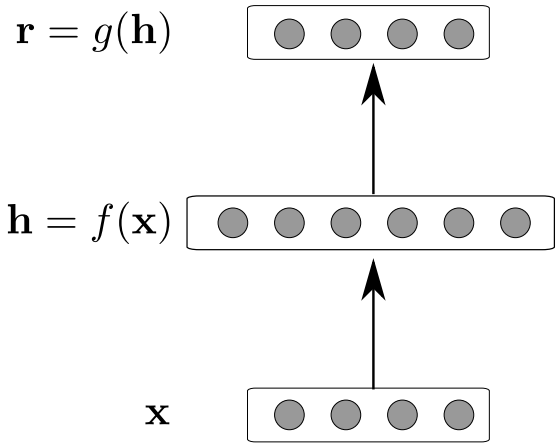
Size of the hidden (bottleneck) layer

Book: Chapter 14



$h < x$, **Undercomplete** hidden layer:

- "Compress" the input
- Compresses well for training samples

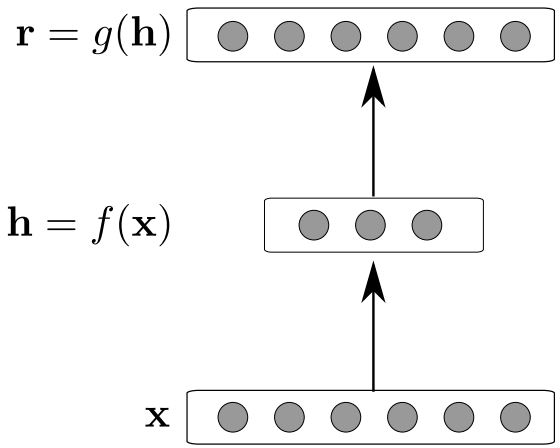


$h > x$, **Overcomplete** hidden layer

- No compression needed (could just copy input)
- Useful for representation learning

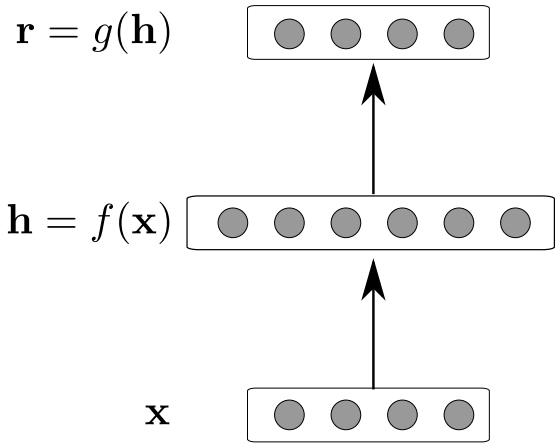
Size of the hidden (bottleneck) layer

Book: Chapter 14



$h < x$, **Undercomplete** hidden layer:

- "Compress" the input
- Compresses well for training samples



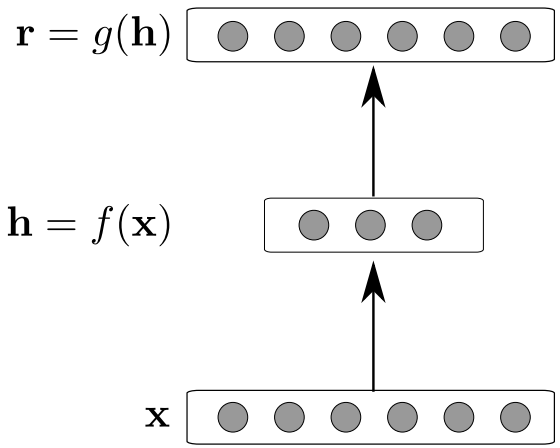
$h > x$, **Overcomplete** hidden layer

- No compression needed (could just copy input)
- Useful for representation learning

  Q: What lecture topic could prevent copying the input?

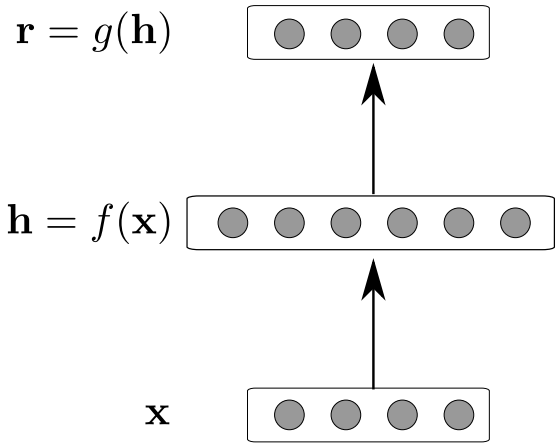
Size of the hidden (bottleneck) layer

Book: Chapter 14



$h < x$, **Undercomplete** hidden layer:

- "Compress" the input
- Compresses well for training samples



$h > x$, **Overcomplete** hidden layer

- No compression needed (could just copy input)
- Useful for representation learning

  Q: What lecture topic could prevent copying the input?
  A: Regularization

Overcomplete: Denoising autoencoder

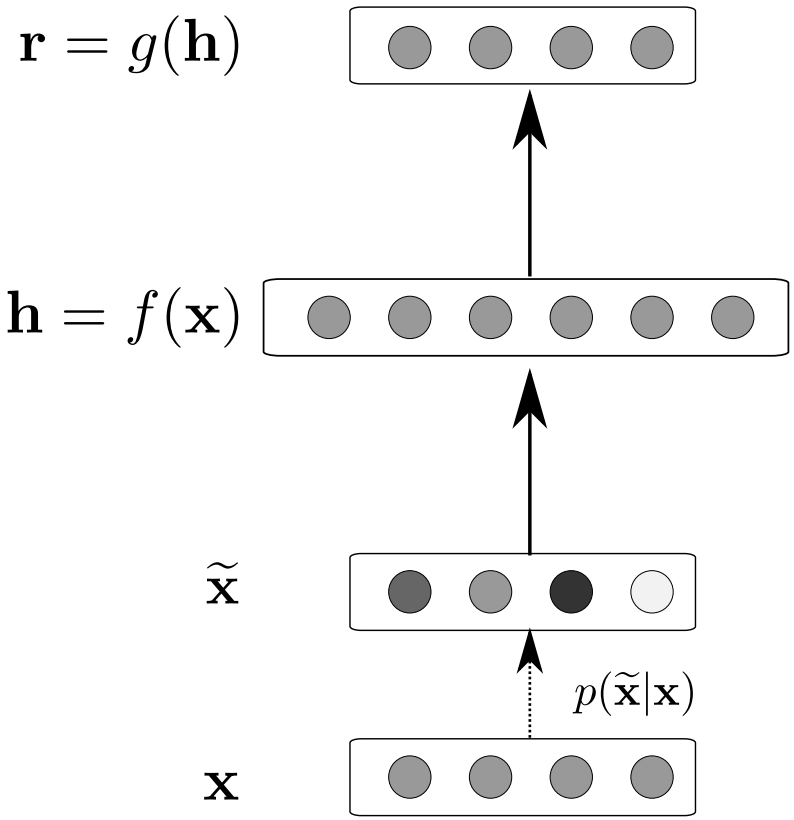
Book: 14.2.2, 14.5

Idea: Regularization by making reconstruction robust to noise.

Overcomplete: Denoising autoencoder

Book: 14.2.2, 14.5

Idea: Regularization by making reconstruction robust to noise.



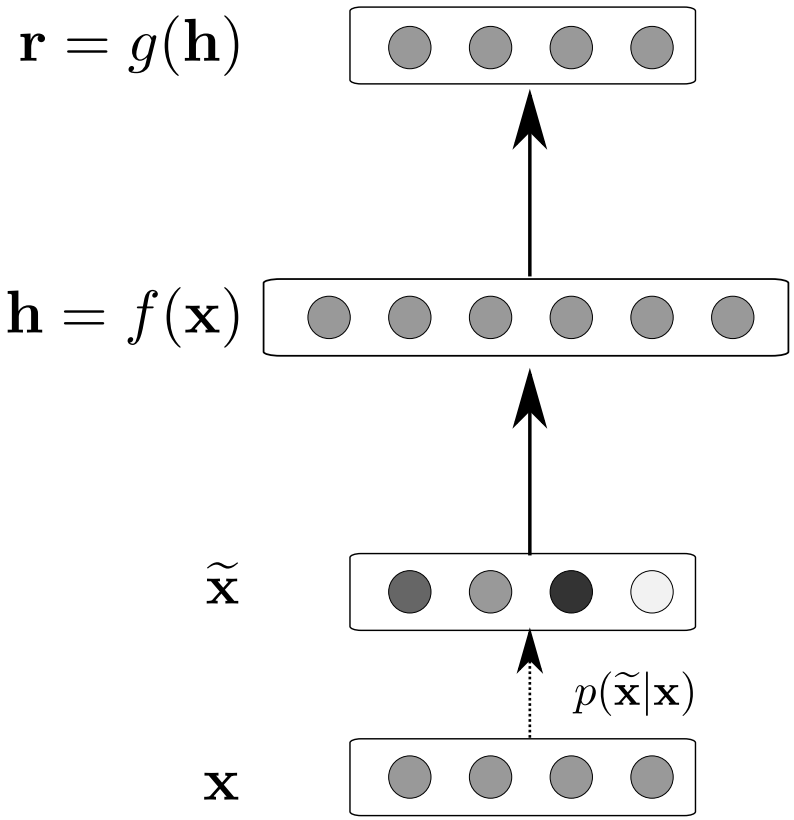
Generate noisy $\widetilde{x}$ from a distribution $p(\widetilde{x}|x)$

Q: How to make robust to noise?

Overcomplete: Denoising autoencoder

Book: 14.2.2, 14.5

Idea: Regularization by making reconstruction robust to noise.



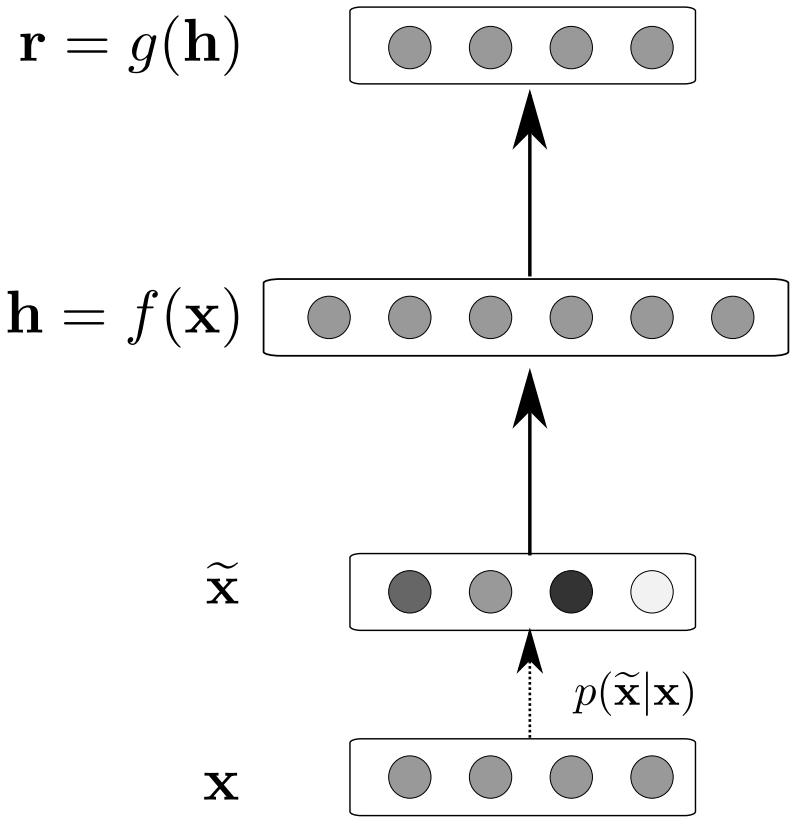
Generate noisy $\widetilde{x}$ from a distribution $p(\widetilde{x}|x)$

Q: How to make robust to noise?
A: Reconstruct the loss on $g(f(\widetilde{x})) = x$

Overcomplete: Denoising autoencoder

Book: 14.2.2, 14.5

Idea: Regularization by making reconstruction robust to noise.



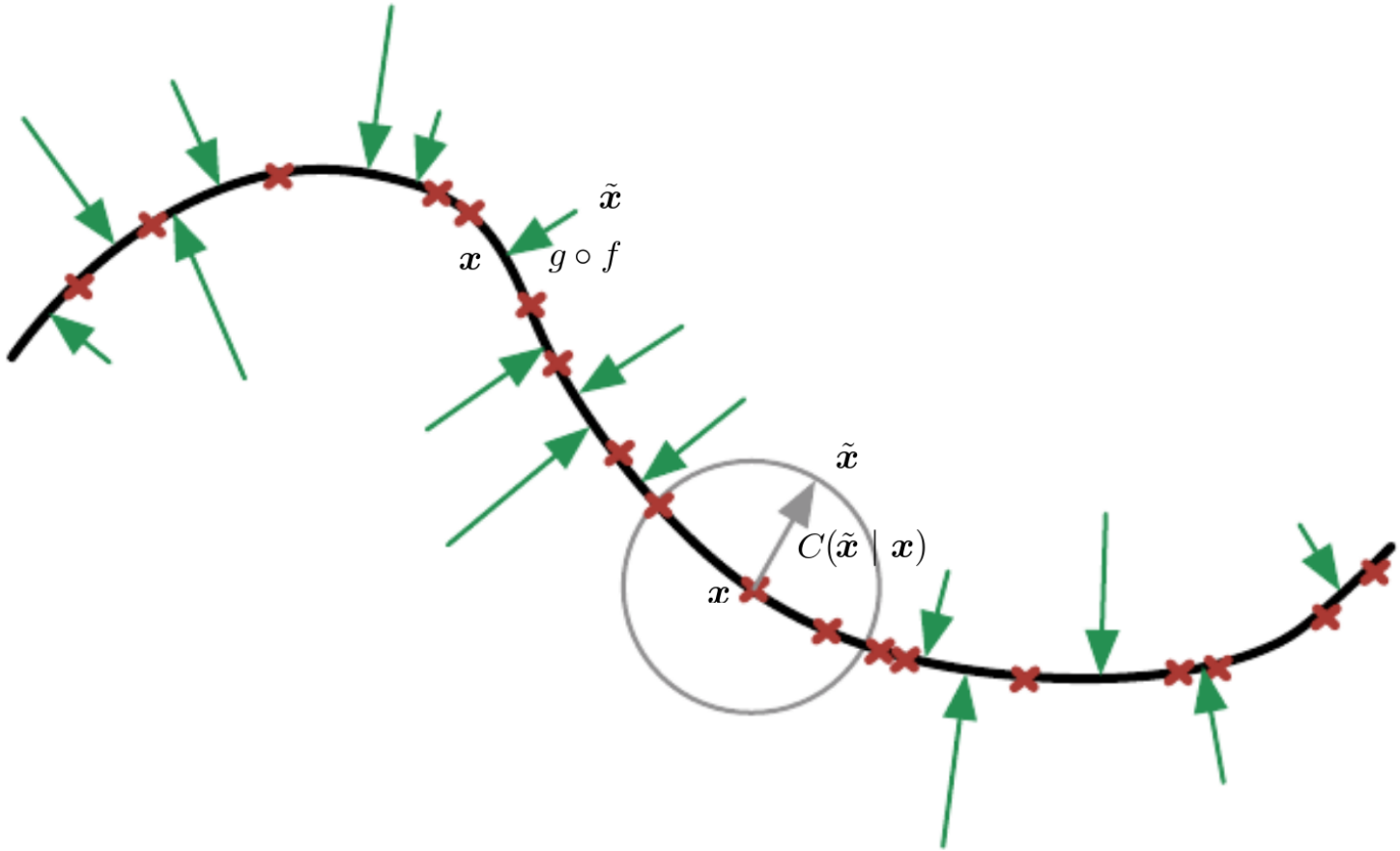
Generate noisy $\widetilde{x}$ from a distribution $p(\widetilde{x}|x)$

Q: How to make robust to noise?
A: Reconstruct the loss on $g(f(\widetilde{x})) = x$

Input a noisy sample $\widetilde{x}$ and optimize to reconstruct the noise-free x.

Denoising autoencoder

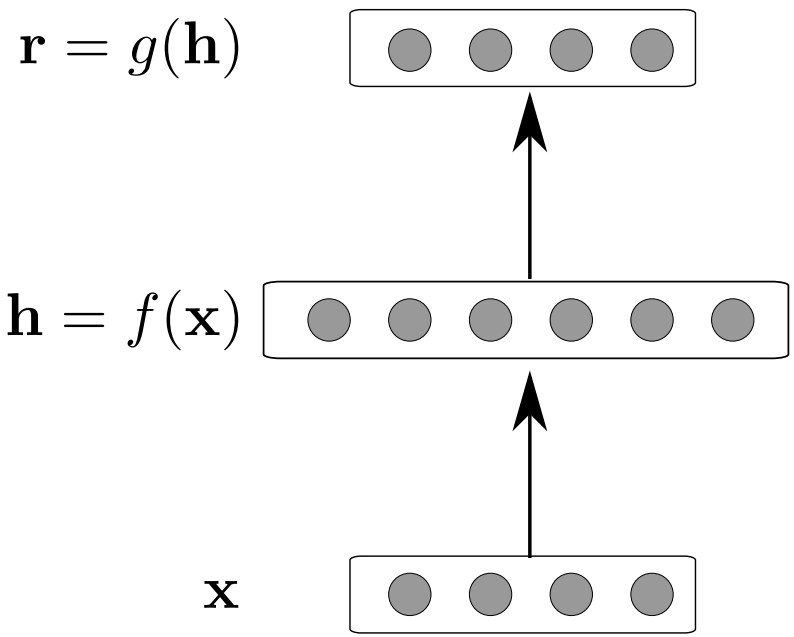
Fig 14.4 in book



True sample x, noisy sample $\widetilde{x}$, Noise process $C(\widetilde{x}|x)$, wrt $p(\widetilde{x}|x)$

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



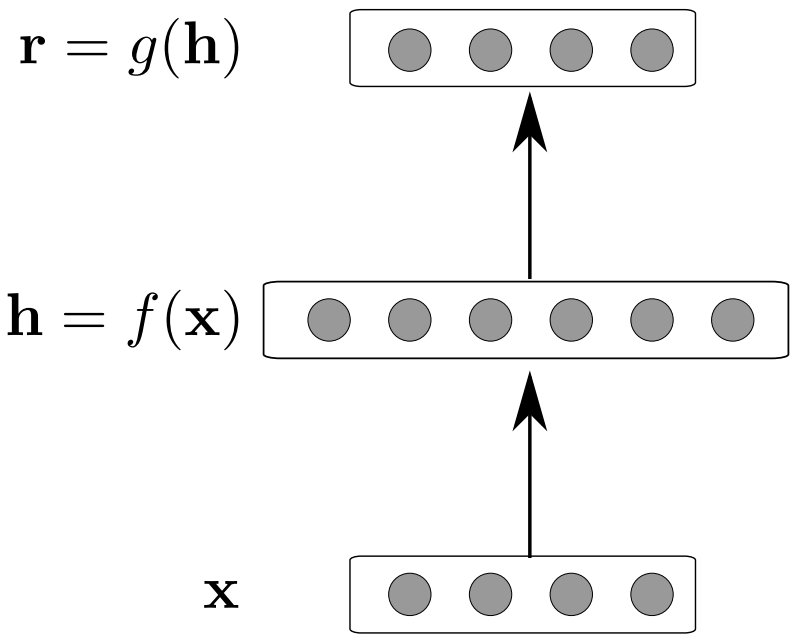
Penalize "unwanted variations".

$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



Penalize "unwanted variations".

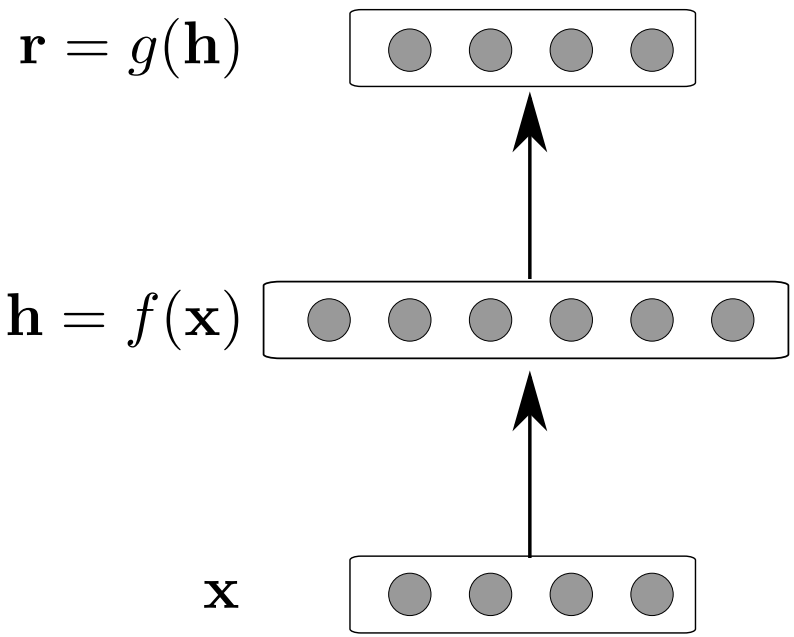
$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

Q: What does $\Omega(h)$ measure?

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



Penalize "unwanted variations".

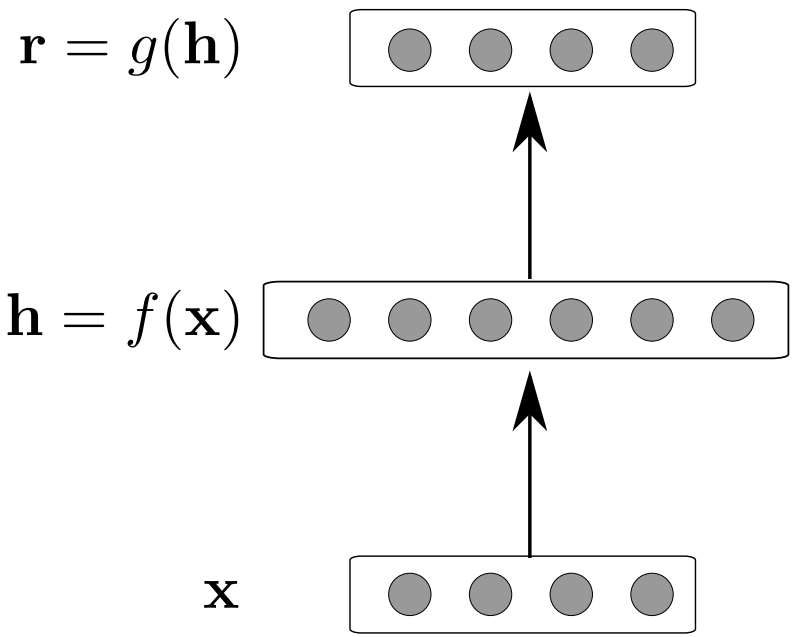
$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

Q: What does $\Omega(h)$ measure?
A: How much h changes when x changes.

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



Penalize "unwanted variations".

$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

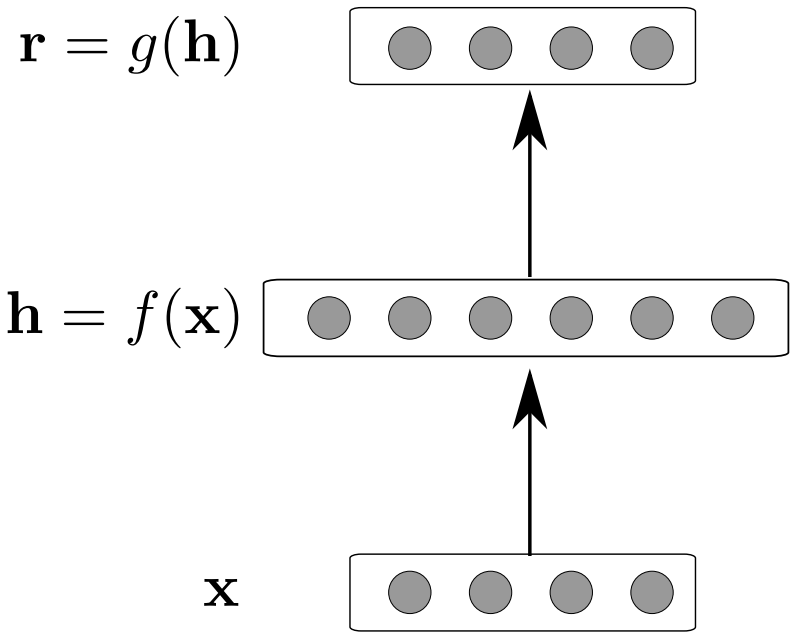
$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

Q: What does $\Omega(h)$ measure?
A: How much h changes when x changes.

Q: What happens when $\Omega(h)$ is small?

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



Penalize "unwanted variations".

$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

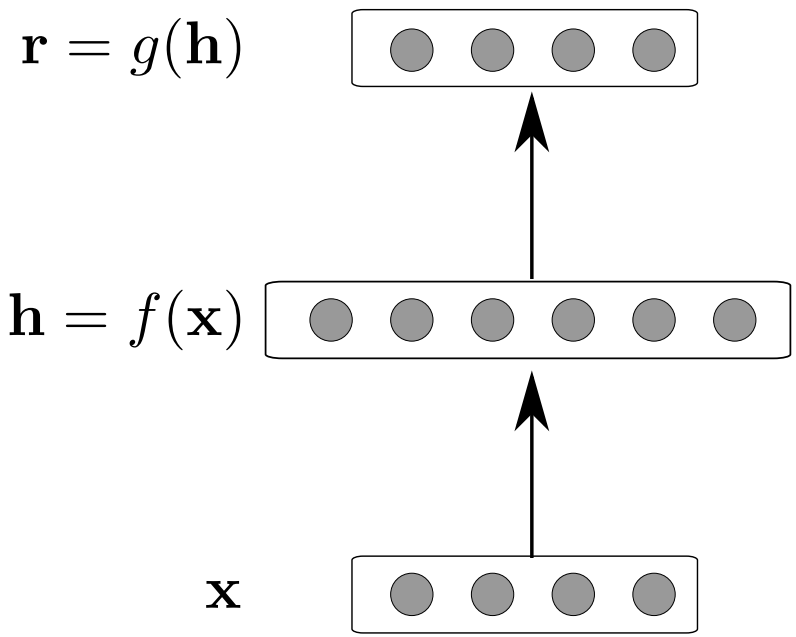
$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

Q: What does $\Omega(h)$ measure?
A: How much h changes when x changes.

Q: What happens when $\Omega(h)$ is small?
A: x changes, h does not change: Robust.

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



Penalize "unwanted variations".

$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

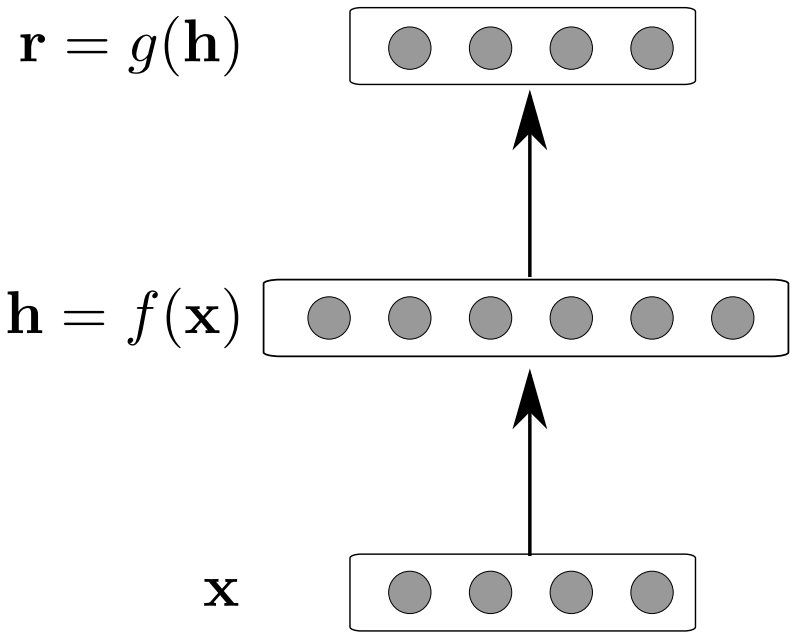
Q: What does $\Omega(h)$ measure?
A: How much h changes when x changes.

Q: What happens when $\Omega(h)$ is small?
A: x changes, h does not change: Robust.

Q: What is the O(compute/memory) used?

Overcomplete: Contractive autoencoder

Book: 14.2.3, 14.7



Penalize "unwanted variations".

$$l(g(f(x)), x) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

$\Omega(h) = \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$ Frobenius norm of Jacobian

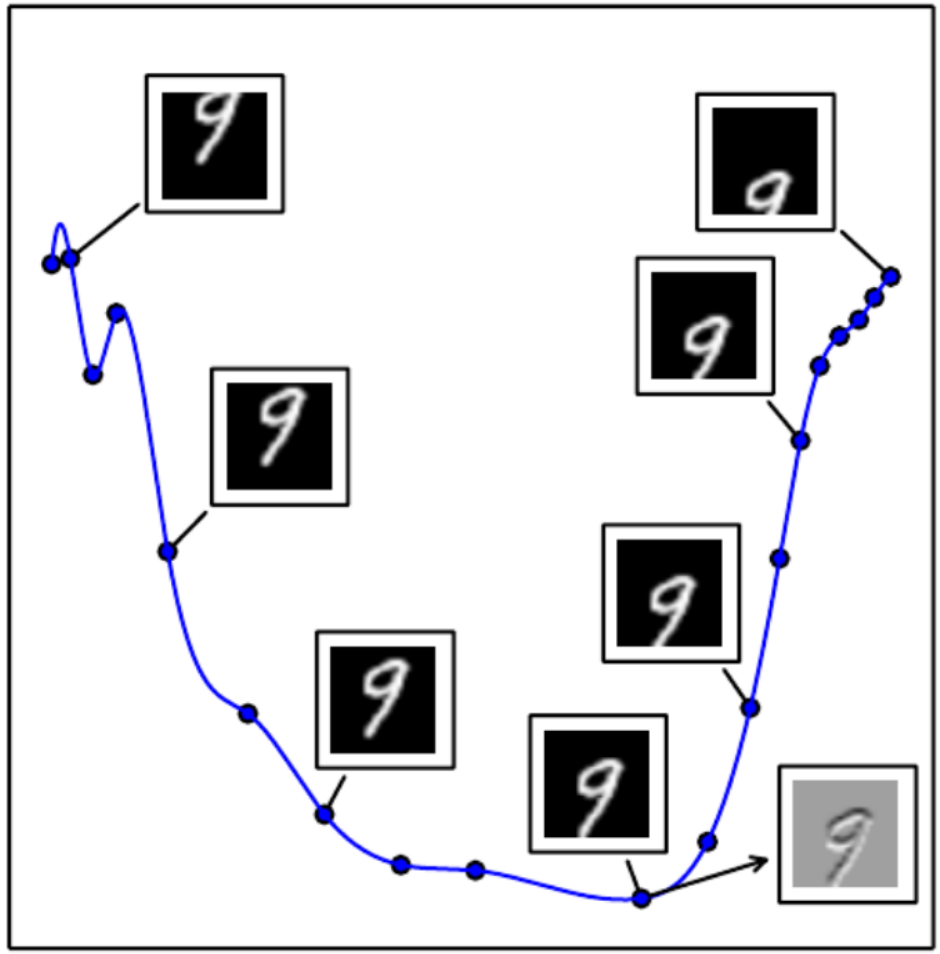
Q: What does $\Omega(h)$ measure?
A: How much h changes when x changes.

Q: What happens when $\Omega(h)$ is small?
A: x changes, h does not change: Robust.

Q: What is the O(compute/memory) used?
A: $O(x \times h)$

Contractive autoencoder

Fig 14.6 in book



Gray pixels: little change. White/Black pixels higher change.

Questions?

Variational Autoencoder

Book: 20.10.3, 20.9, Better: `https://www.jeremyjordan.me/variational-autoencoders/`

Q: How could you sample (generate samples) from an auto encoder?

Variational Autoencoder

Book: 20.10.3, 20.9, Better: `https://www.jeremyjordan.me/variational-autoencoders/`

Q: How could you sample (generate samples) from an auto encoder?
A: Cannot: Undefined range.

Variational Autoencoder

Book: 20.10.3, 20.9, Better: `https://www.jeremyjordan.me/variational-autoencoders/`

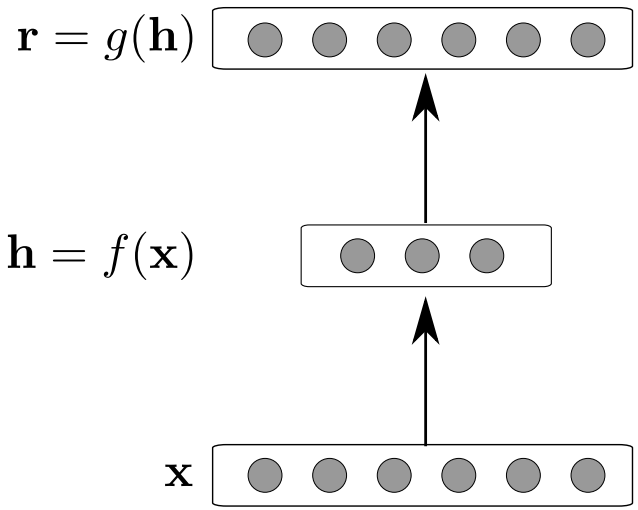
Q: How could you sample (generate samples) from an auto encoder?
A: Cannot: Undefined range.
Idea: Make hidden layer a distribution which is easy to sample from: $\mathcal{N}(0, I)$.

Variational Autoencoder

Book: 20.10.3, 20.9, Better: `https://www.jeremyjordan.me/variational-autoencoders/`

Q: How could you sample (generate samples) from an auto encoder?
A: Cannot: Undefined range.
Idea: Make hidden layer a distribution which is easy to sample from: $\mathcal{N}(0, I)$.

Autoencoder:

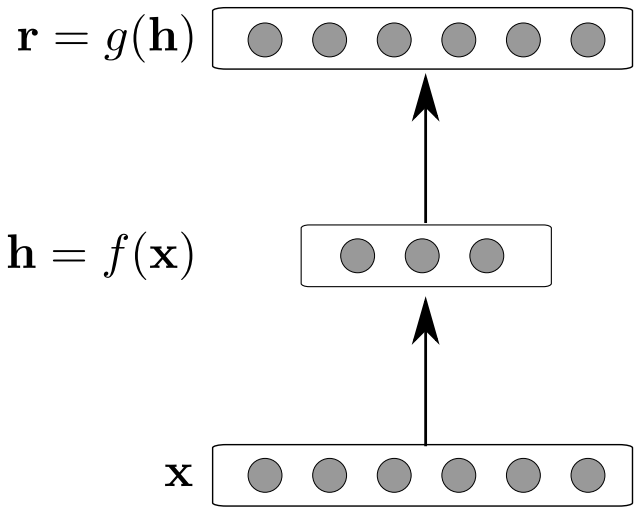


Variational Autoencoder

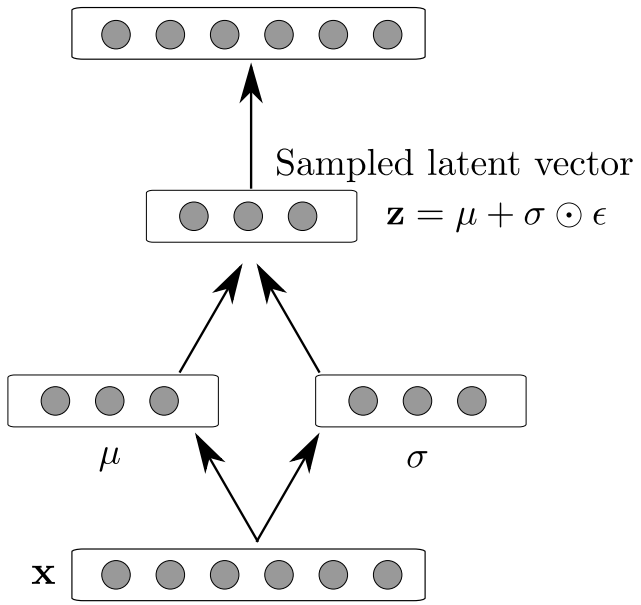
Book: 20.10.3, 20.9, Better: `https://www.jeremyjordan.me/variational-autoencoders/`

Q: How could you sample (generate samples) from an auto encoder?
A: Cannot: Undefined range.
Idea: Make hidden layer a distribution which is easy to sample from: $\mathcal{N}(0, I)$.

Autoencoder:



Variational autoencoder:

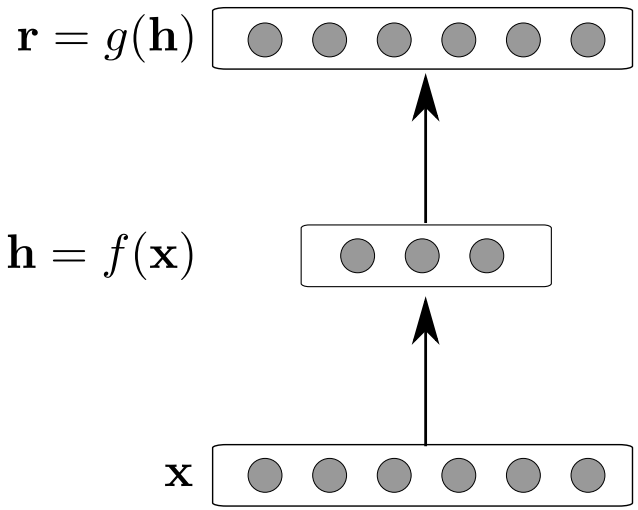


Variational Autoencoder

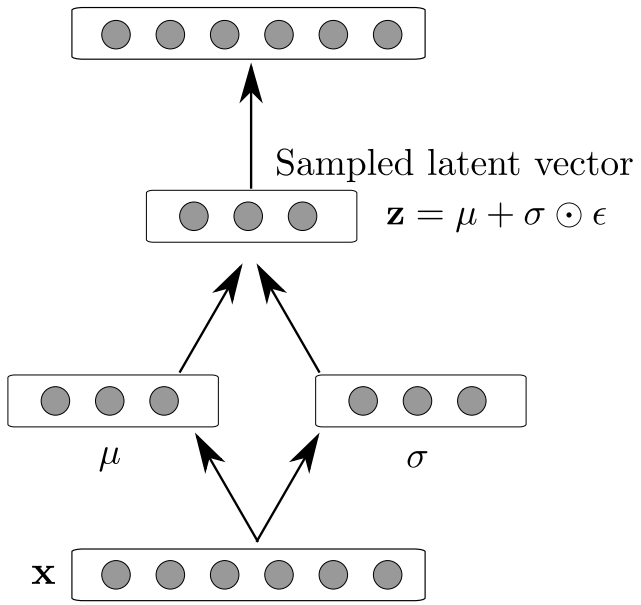
Book: 20.10.3, 20.9, Better: `https://www.jeremyjordan.me/variational-autoencoders/`

Q: How could you sample (generate samples) from an auto encoder?
A: Cannot: Undefined range.
Idea: Make hidden layer a distribution which is easy to sample from: $\mathcal{N}(0, I)$.

Autoencoder:



Variational autoencoder:



Reparameterization trick: Compute sampler gradients, draw $\epsilon \sim \mathcal{N}(0, 1)$. (Book: 20.9).

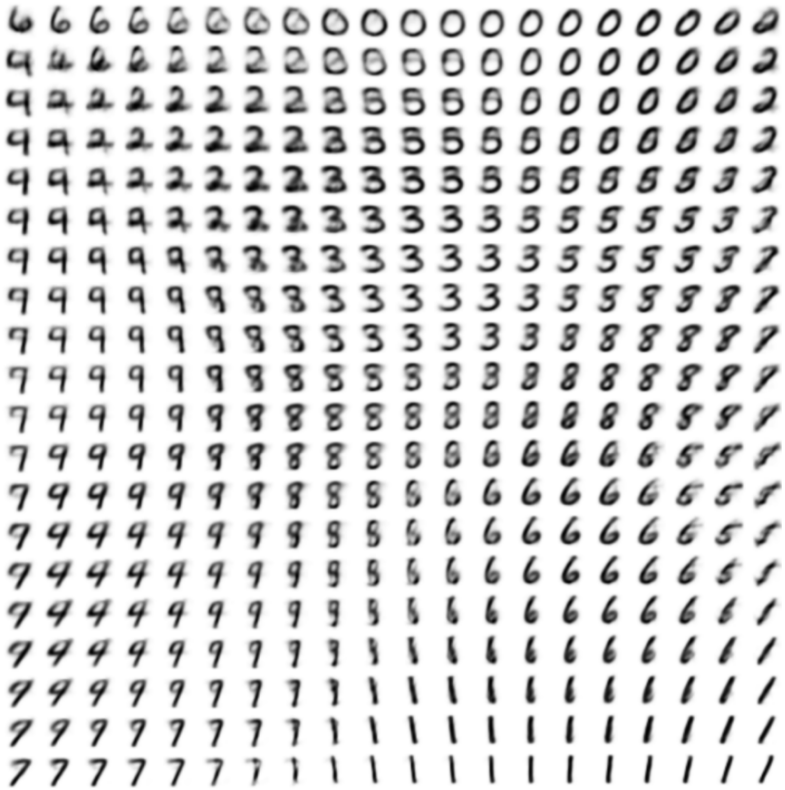
Variational autoencoder: Examples in 2D

Faces:

Variational autoencoder: Examples in 2D

Faces:

MNIST:



Generative Adversarial Networks

Book: Chapter 20.10.4

Problem: No direct way to sample from high dimensional training distribution.

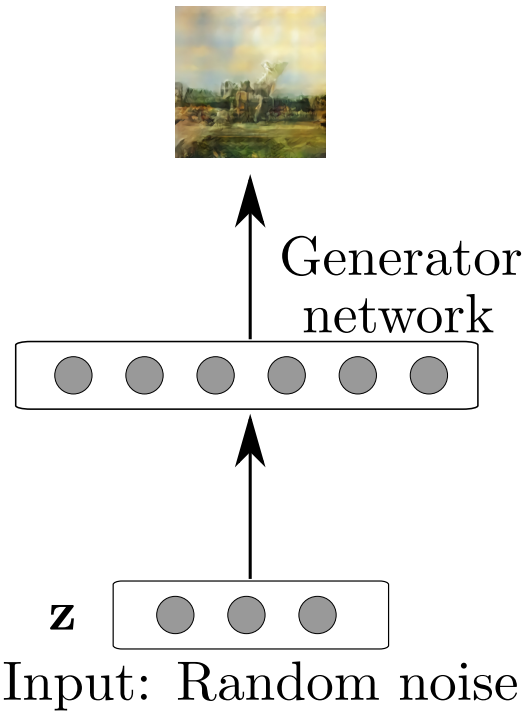
Solution: Sample from simple distribution, such as random noise, and learn transformation from noise to training distribution.

Generative Adversarial Networks

Book: Chapter 20.10.4

Problem: No direct way to sample from high dimensional training distribution.

Solution: Sample from simple distribution, such as random noise, and learn transformation from noise to training distribution.



Training GANS: Two player game

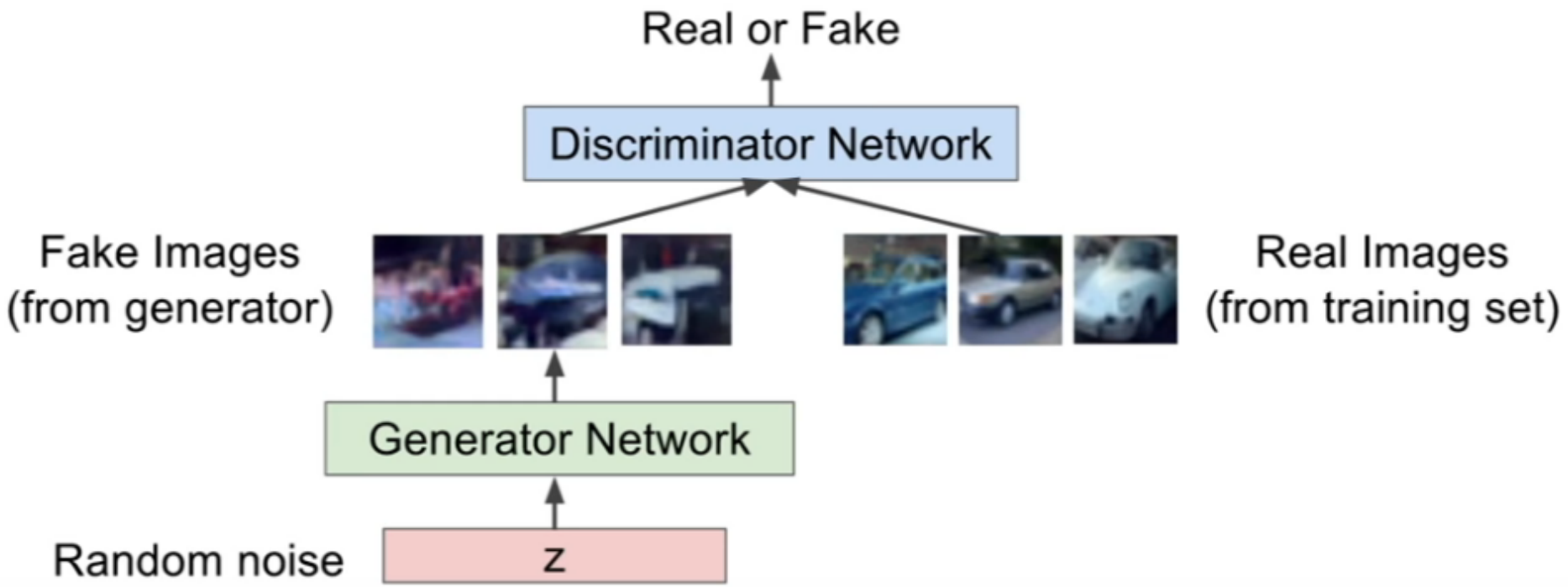
Generator network:
Try to fool the discriminator by generating real looking images.

Discriminator network:
Try to discriminate between real and fake images.

Training GANS: Two player game

Generator network:
Try to fool the discriminator by generating real looking images.

Discriminator network:
Try to discriminate between real and fake images.



Training GANS: Two player game

Generator network:
Try to fool the discriminator by generating real looking images.

Discriminator network:
Try to discriminate between real and fake images (between 0 and 1).

Training GANS: Two player game

Generator network:
Try to fool the discriminator by generating real looking images.

Discriminator network:
Try to discriminate between real and fake images (between 0 and 1).

Objective:

$$\arg\min_{\theta_G}\max_{\theta_D}\mathbb{E}_{x\sim p_{\text{data}}}\log D_{\theta_D}(x)+\mathbb{E}_{x\sim p(z)}\log(1-D_{\theta_D}(G_{\theta_G}(z)))$$

Q: What values for which terms does each network aim to obtain?

Training GANS: Two player game

Generator network:
Try to fool the discriminator by generating real looking images.

Discriminator network:
Try to discriminate between real and fake images (between 0 and 1).

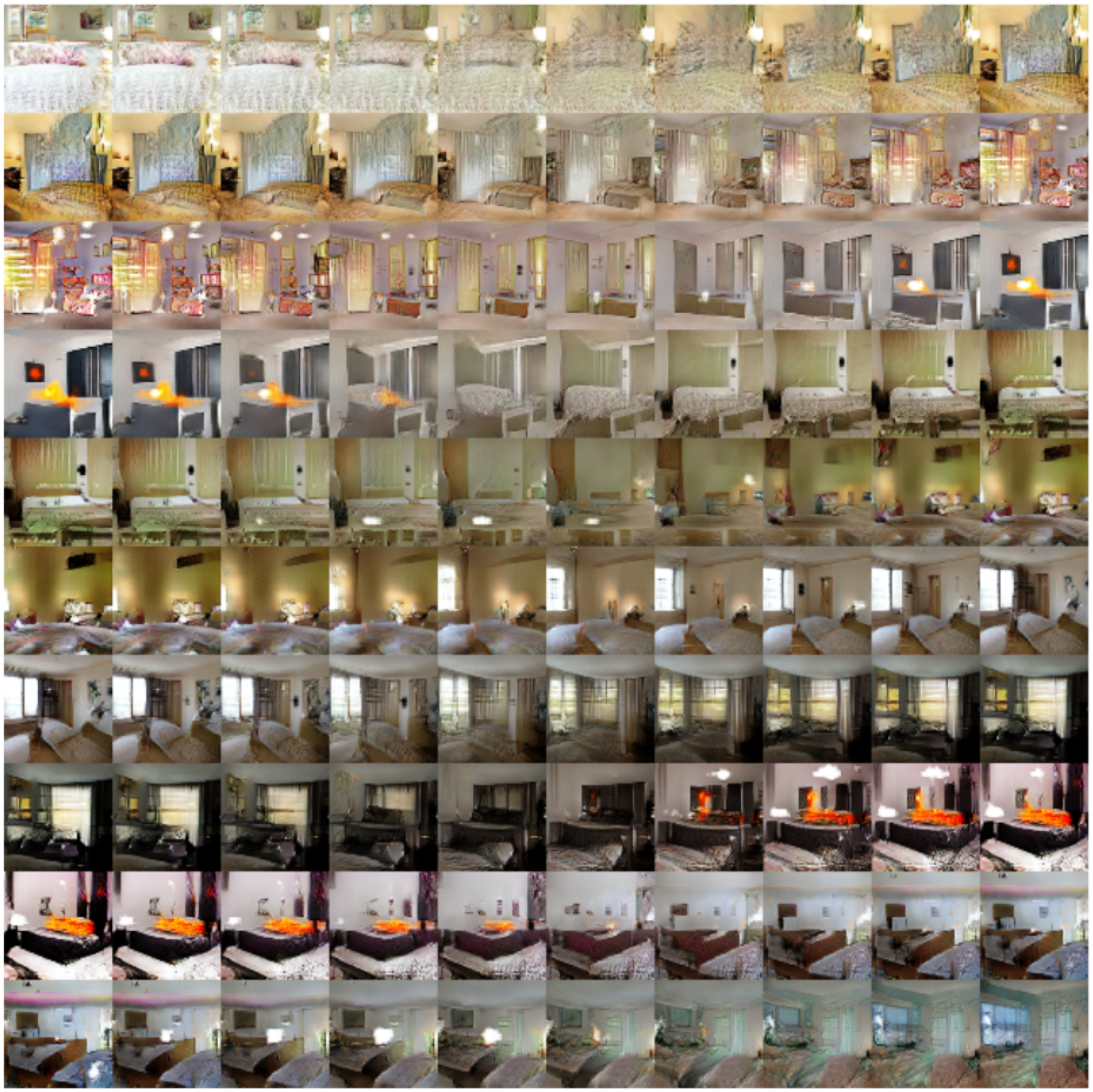
Objective:

$$\arg\min_{\theta_G}\max_{\theta_D} \mathbb{E}_{x\sim p_{\text{data}}} \underbrace{\log D_{\theta_D}(x)}_{\text{D for real data x}} + \mathbb{E}_{x\sim p(z)} \log(1 - \underbrace{D_{\theta_D}(G_{\theta_G}(z))}_{\text{D for fake data}})$$

Q: What values for which terms does each network aim to obtain?
Discriminator D_{θ_D} aims to **maximize** so that $D_{\theta_D}(x)$ is close to 1 for real data and $D_{\theta_D}(G_{\theta_G}(x))$ is 0 for fakes.

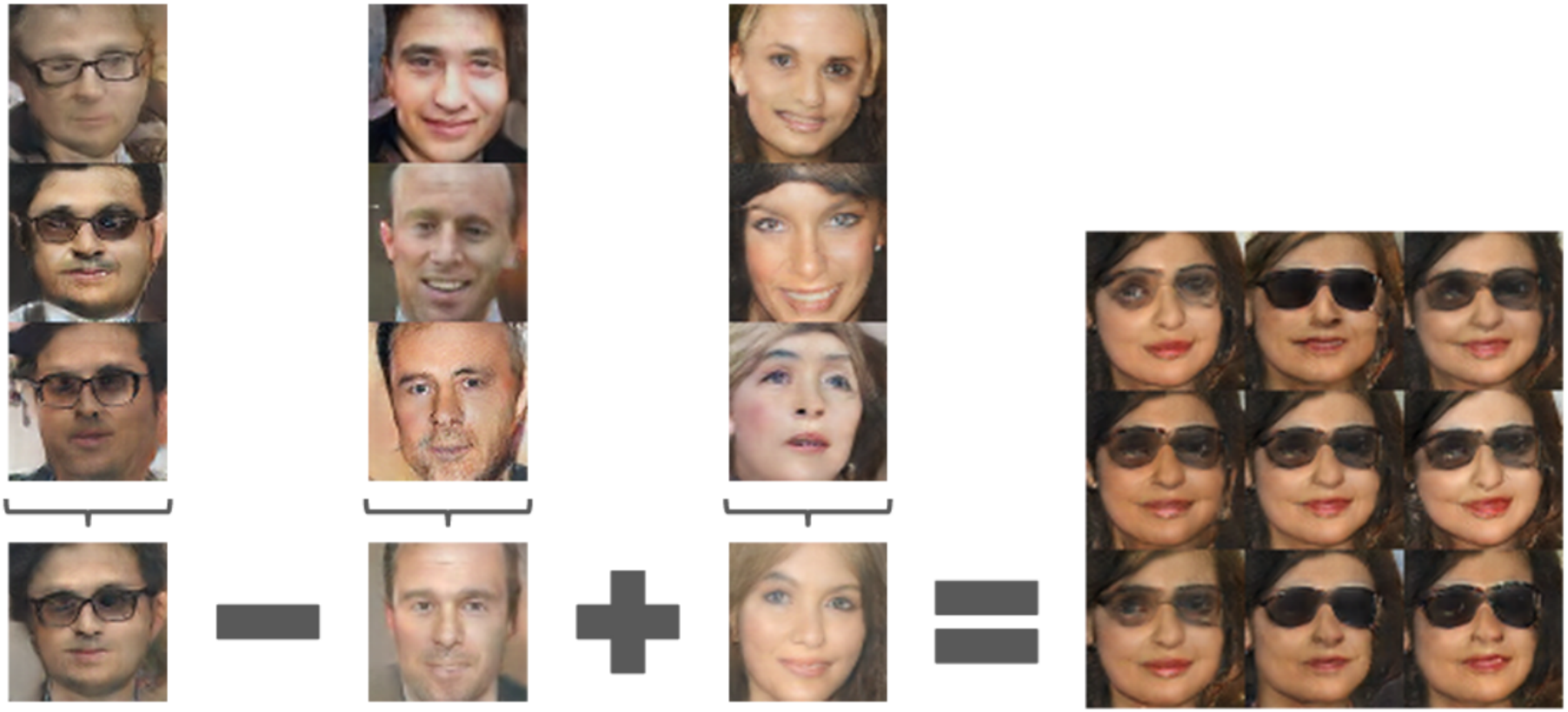
Generator G_{θ_G} aims to **minimize** so that $D_{\theta_D}(G_{\theta_G}(x))$ is close to 1 (fooling the discriminator)

Interpolation in latent space



Vector arithmetics

Vector arithmetics



How to evaluate generative models?

Book: Chapter 20.14

- Q: Can't we just evaluate likelihoods?

How to evaluate generative models?

Book: Chapter 20.14

- Q: Can't we just evaluate likelihoods?
  A: Higher likelihoods is not necessarily visually better than a lower one.

- Q: Can't we let external evaluators evaluate model quality?

How to evaluate generative models?

Book: Chapter 20.14

- Q: Can't we just evaluate likelihoods?
  A: Higher likelihoods is not necessarily visually better than a lower one.
- Q: Can't we let external evaluators evaluate model quality?
  A: Just copying the training set will do very well
- Q: Can't we use the unsupervised features to evaluate another task (e.g., classification accuracy)?

How to evaluate generative models?

Book: Chapter 20.14

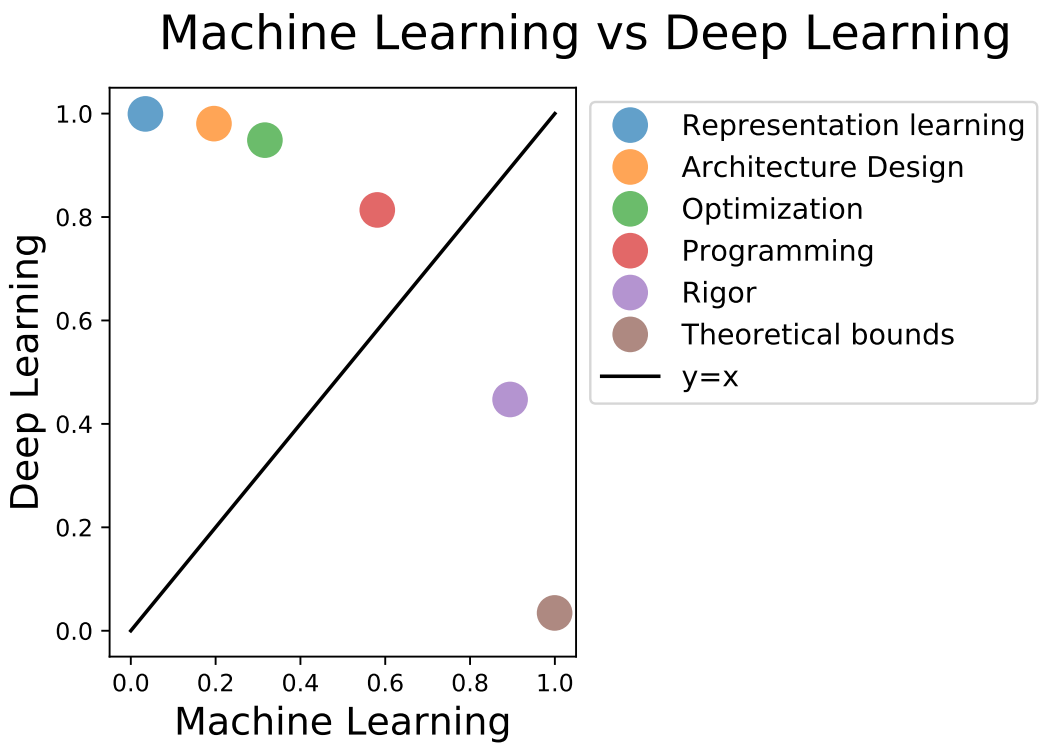
- Q: Can't we just evaluate likelihoods?
  A: Higher likelihoods is not necessarily visually better than a lower one.
- Q: Can't we let external evaluators evaluate model quality?
  A: Just copying the training set will do very well
- Q: Can't we use the unsupervised features to evaluate another task (e.g., classification accuracy)?
  A: Yes, but that does limit the use of generative models

How to evaluate generative models is still a research topic.

Questions?

Deep learning and Machine learning

Difference between
machine learning and deep learning



- Deep learning: coupled to the application
- Machine learning: application independent
- Lets discuss this slide again after the course