

# Neural Networks

## Lecture 11: Autoencoders & GANs

Lesson	Title
1	Introduction
2	Probability and Information Theory
3	Basics of Machine Learning
4	Deep Feed-forward networks
5	Back-propagation
6	Optimization and regularization
7	Convolutional Networks
8	Sequence Modeling: recurrent networks
9	Applications
10	Software and Practical methods
11	Representational Learning
12	Deep Generative Models
13	Deep Reinforcement Learning
14	Deep Learning and the Brain
15	Conclusions and Future Perspective
16	Project presentations

Foundations

Basics

Advanced

Selected topics

Lesson	Title
1	Introduction
2	Probability and Information Theory
3	Basics of Machine Learning
4	Deep Feed-forward networks
5	Back-propagation
6	Optimization and regularization
7	Convolutional Networks
8	Sequence Modeling: recurrent networks
9	Applications
10	Software and Practical methods
11	Autoencoders & GANs
12	Project Development
13	Deep Reinforcement Learning
14	Deep Learning and the Brain
15	Conclusions and Future Perspective
16	Project presentations

Foundations

Basics

Advanced

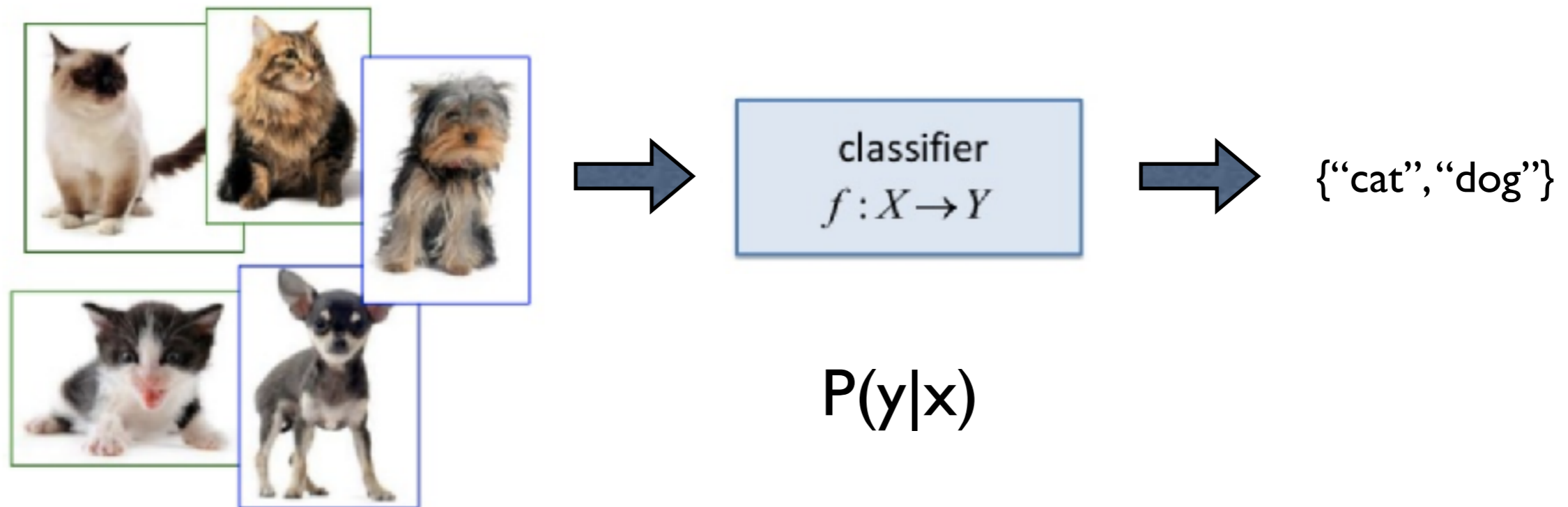
Selected topics

# Learning objectives

- Autoencoders
- Transfer learning and domain adaptation
- Generative adversarial networks

# Types of ML depending of experience

- Supervised: each sample is associated to a label/target



# Supervised learning

**Data:**  $(x, y)$

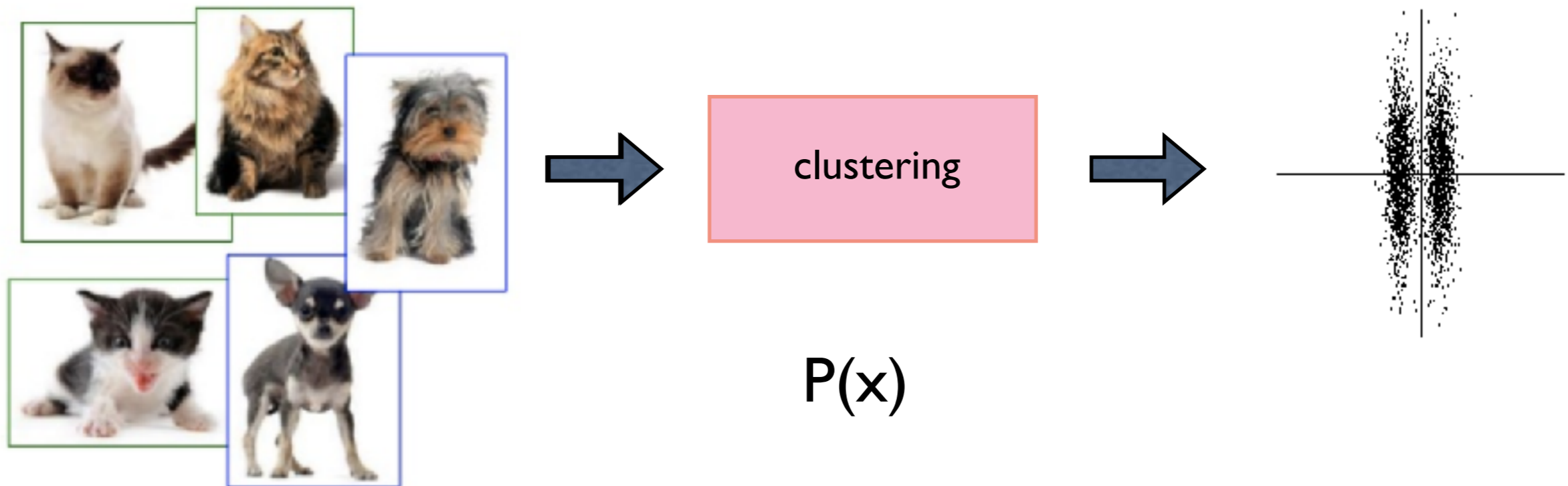
$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

# Types of ML depending of experience

- Supervised
- Unsupervised: each sample contains features but no label



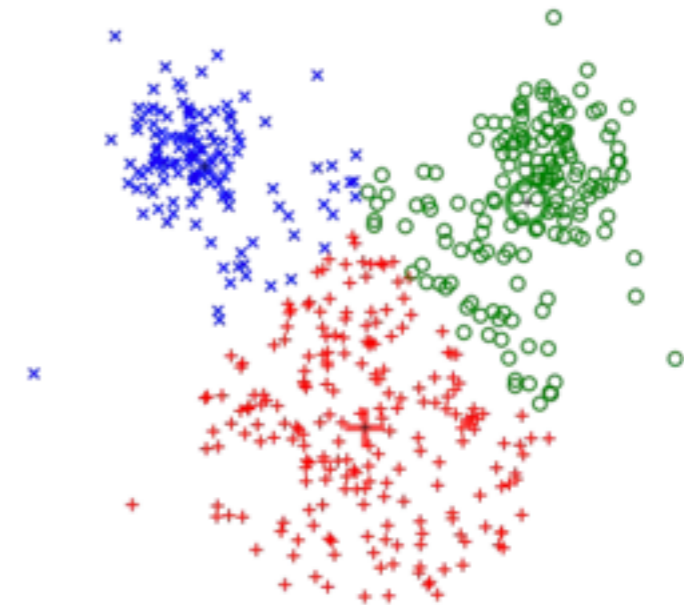
# Unsupervised learning

**Data:**  $x$

just data, no labels

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

**Examples:** clustering,  
dimensionality reduction,  
feature learning,  
density estimation, etc.



K-means clustering

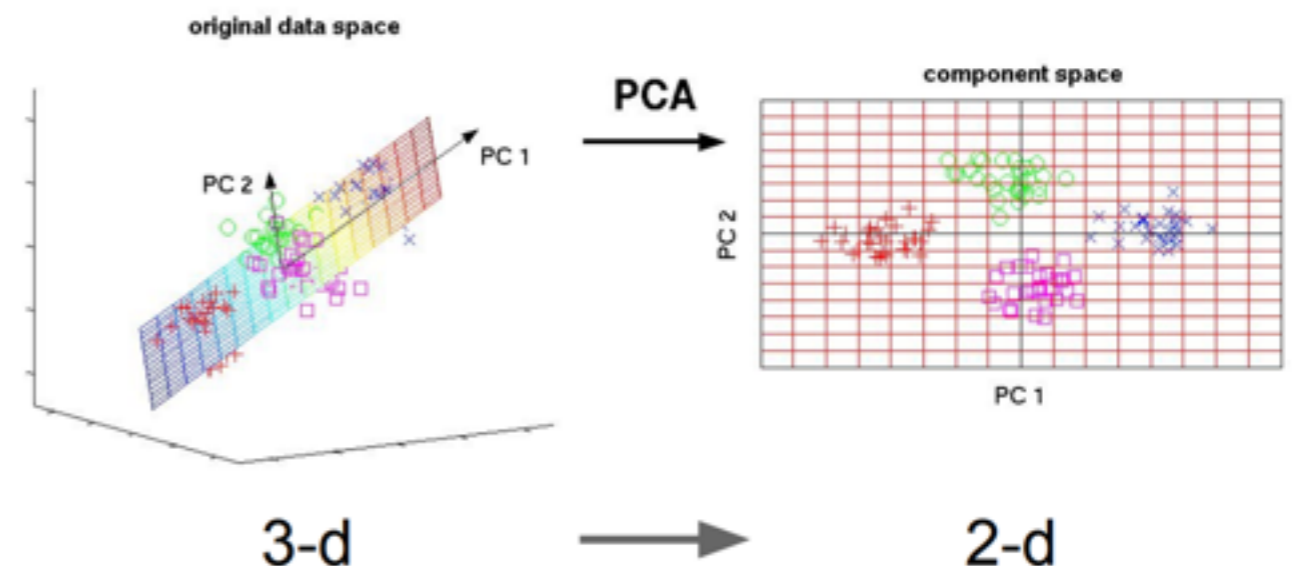
# Unsupervised learning

**Data:**  $x$

just data, no labels

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

**Examples:** clustering, dimensionality reduction, feature learning, density estimation, etc.



Principal component analysis  
(Dimensionality reduction)

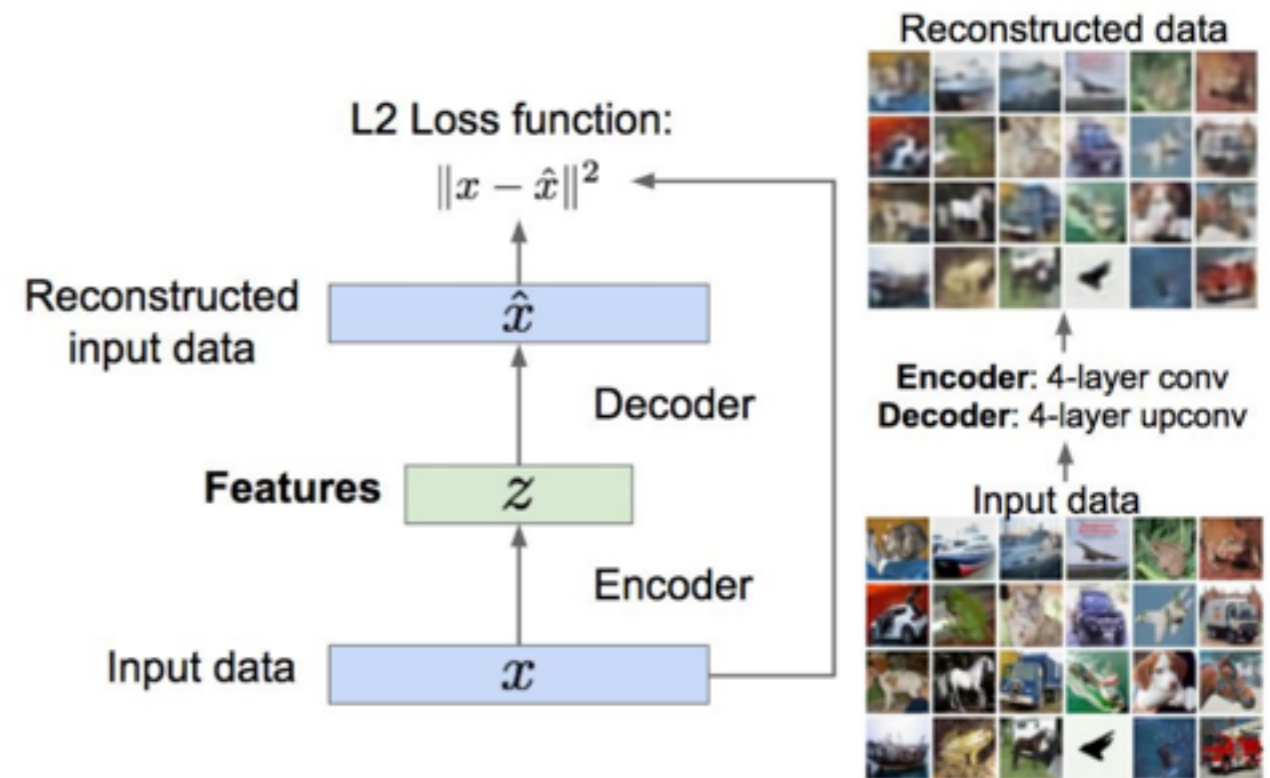
# Unsupervised learning

**Data:**  $x$

just data, no labels

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

**Examples:** clustering, dimensionality reduction, feature learning, density estimation, etc.



Autoencoders  
(Feature learning)

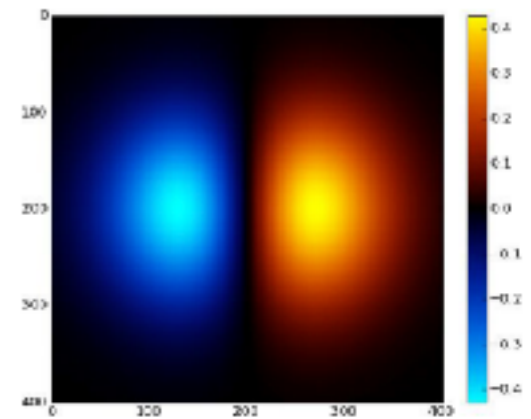
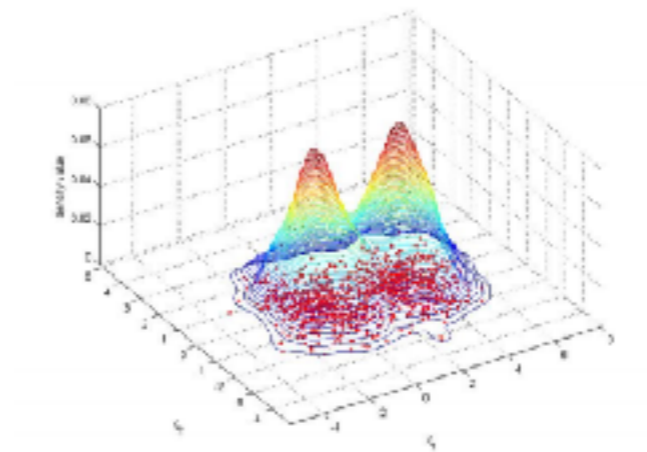
# Unsupervised learning

**Data:**  $x$

just data, no labels

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

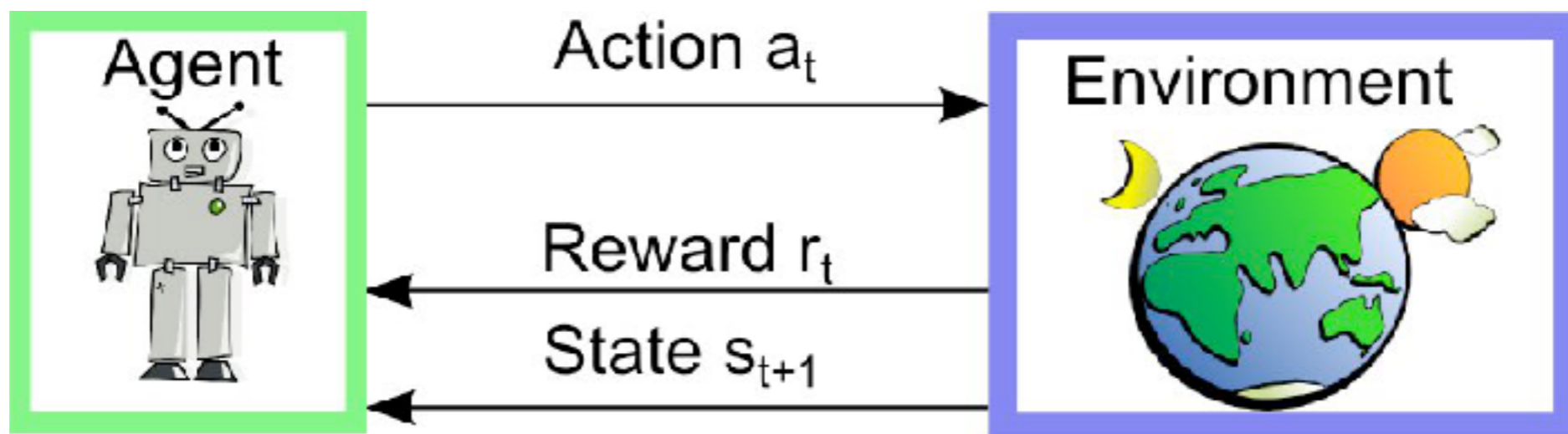
**Examples:** clustering,  
dimensionality reduction,  
feature learning,  
density estimation, etc.



2-d density estimation

# Types of ML depending of experience

- Supervised
- Unsupervised
- Reinforcement: experience via interacting with environment



# Supervised vs Unsupervised learning

## Supervised learning

**Data:**  $(x, y)$

$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

## 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, etc.

# Supervised vs Unsupervised learning

## Supervised learning

**Data:**  $(x, y)$

$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

## Unsupervised learning

Training data is cheap

**Data:**  $x$

just data, no labels!

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

**Examples:** clustering,  
dimensionality reduction,  
feature learning,  
density estimation, etc.

# Supervised vs Unsupervised learning

## Supervised learning

**Data:**  $(x, y)$

$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

## Unsupervised learning

Training data is cheap

**Data:**  $x$

just data, no labels!

Holy grail: solve  
unsupervised learning  
 $\rightarrow$  understand structure  
generating process

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

**Examples:** clustering,  
dimensionality reduction,  
feature learning,  
density estimation, etc.

# Supervised vs Unsupervised learning

## Supervised learning

**Data:**  $(x, y)$

$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

## Unsupervised learning

Training data is cheap

**Data:**  $x$

just data, no labels!

Holy grail: solve  
unsupervised learning  
 $\rightarrow$  understand structure  
generating process

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

**Examples:** clustering,  
dimensionality reduction,  
feature learning,  
density estimation, etc.

# Supervised vs Unsupervised learning

## Supervised learning

**Data:**  $(x, y)$

$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

## Unsupervised learning

Training data is cheap

**Data:**  $x$

just data, no labels!

Holy grail: solve  
unsupervised learning  
→ understand structure  
generating process

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

**Examples:** clustering,  
dimensionality reduction,  
feature learning,  
density estimation, etc.

# Supervised vs Unsupervised learning

## Supervised learning

**Data:**  $(x, y)$

$x$  is data,  $y$  is label

**Goal:** Learn a function  
to map  $x \rightarrow y$

**Examples:** classification,  
regression, object detection,  
segmentation, image captioning,  
etc.

## Unsupervised learning

Training data is cheap

**Data:**  $x$

just data, no labels!

Holy grail: solve  
unsupervised learning  
 $\rightarrow$  understand structure  
generating process

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

**Examples:** clustering,  
dimensionality reduction,

feature learning,

density estimation, etc.

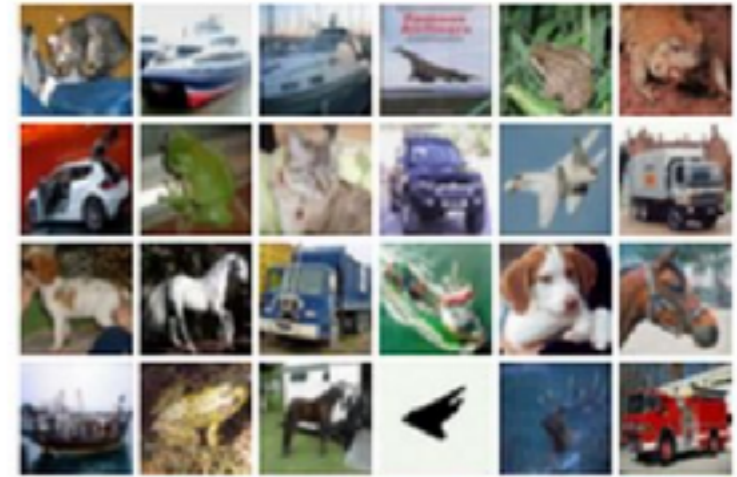
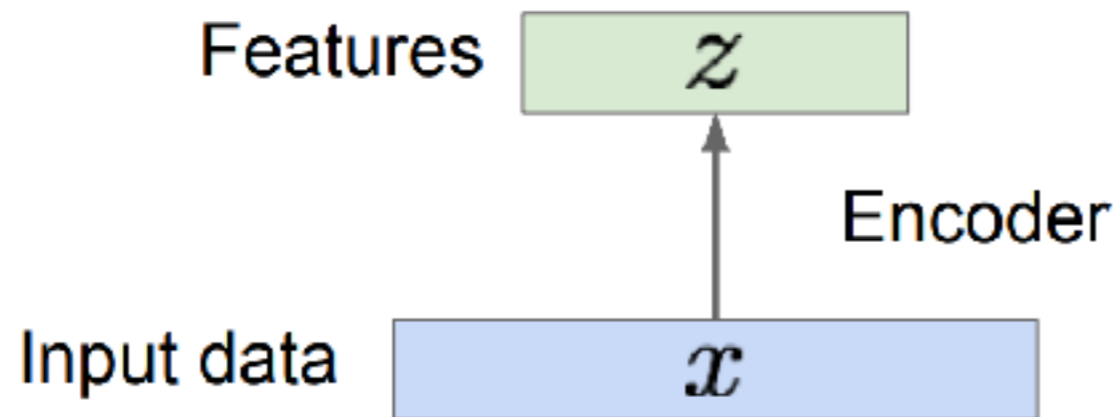
Autoencoders

GANs

# Autoencoders

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

$\mathbf{z}$  usually smaller than  $\mathbf{x}$   
(dimensionality reduction)



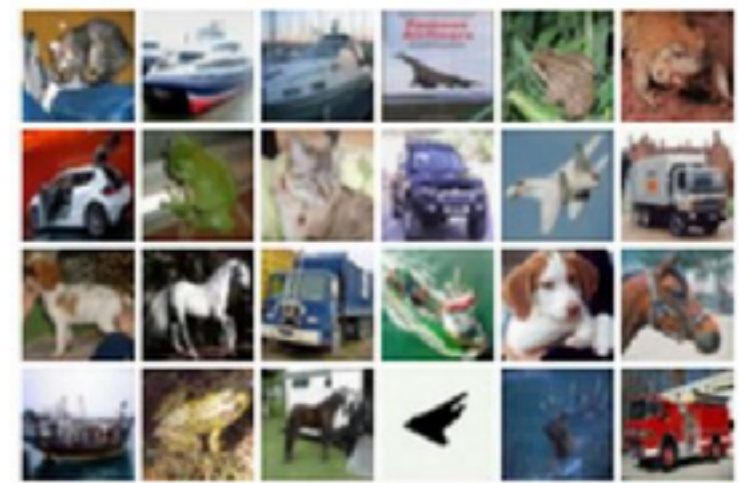
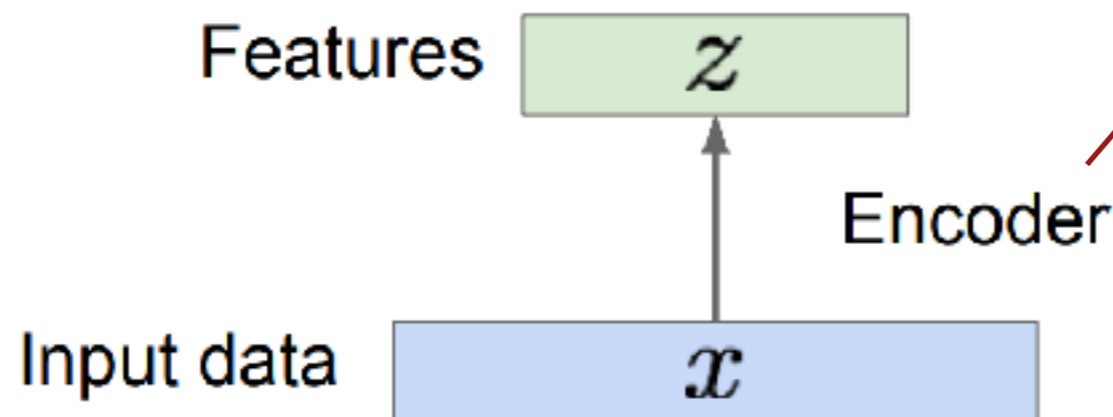
# Autoencoders

Unsupervised approach for learning a lower-dimensional feature Representation from unlabelled training data

**Originally:** Linear + nonlinearity (sigmoid)

**Later:** Deep, fully connected

**Later:** ReLU CNN



# Autoencoders

Unsupervised approach for learning a lower-dimensional feature Representation from unlabelled training data

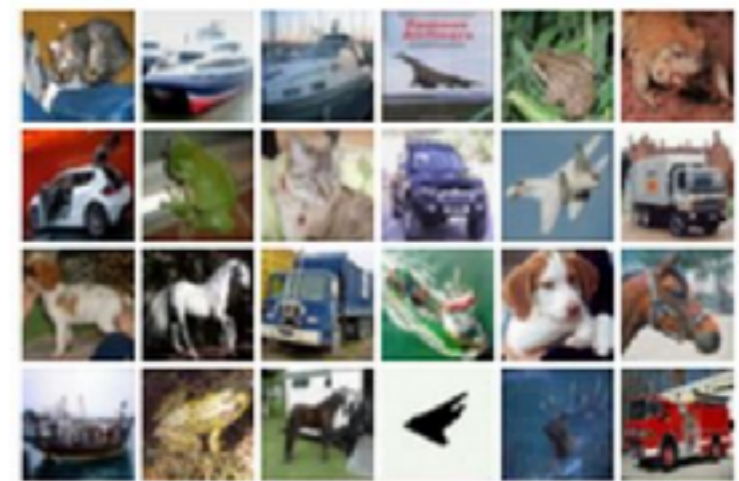
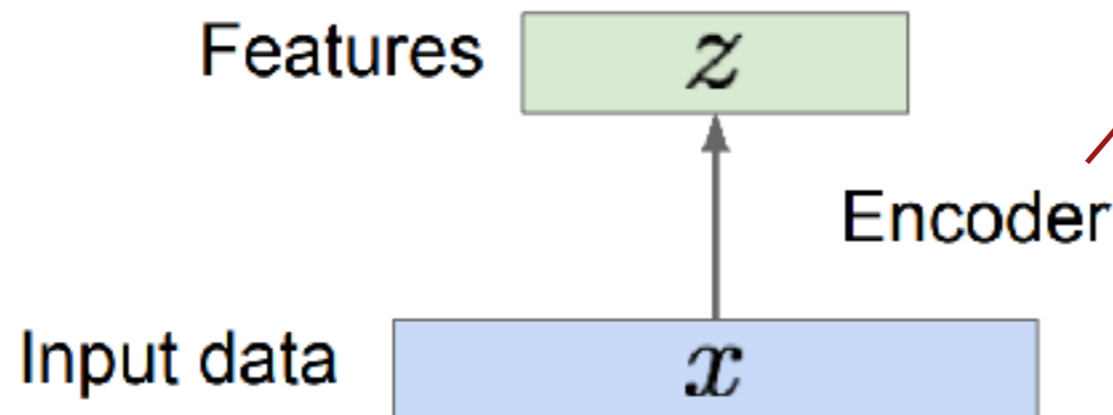
$\mathbf{z}$  usually smaller than  $\mathbf{x}$   
(dimensionality reduction)

Why smaller?

**Originally:** Linear +  
nonlinearity (sigmoid)

**Later:** Deep, fully connected

**Later:** ReLU CNN



# Autoencoders

Unsupervised approach for learning a lower-dimensional feature Representation from unlabelled training data

$\mathbf{z}$  usually smaller than  $\mathbf{x}$   
(dimensionality reduction)

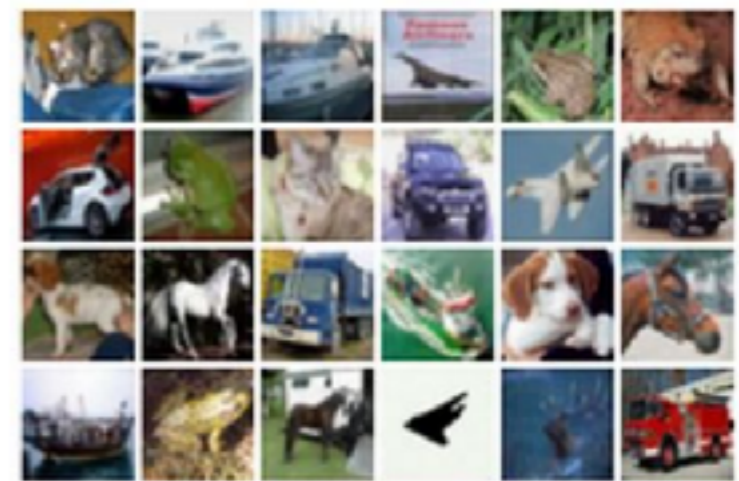
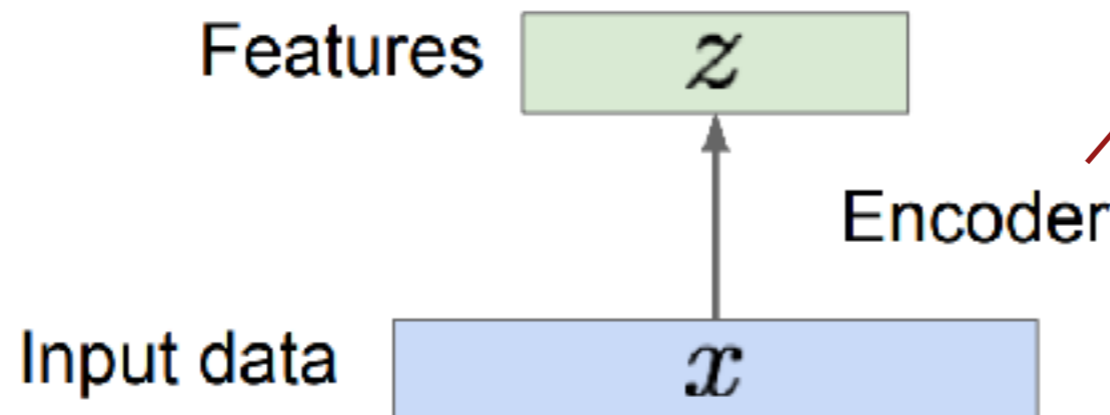
Why smaller?

We want features to capture meaningful factors of variation in data

**Originally:** Linear + nonlinearity (sigmoid)

**Later:** Deep, fully connected

**Later:** ReLU CNN

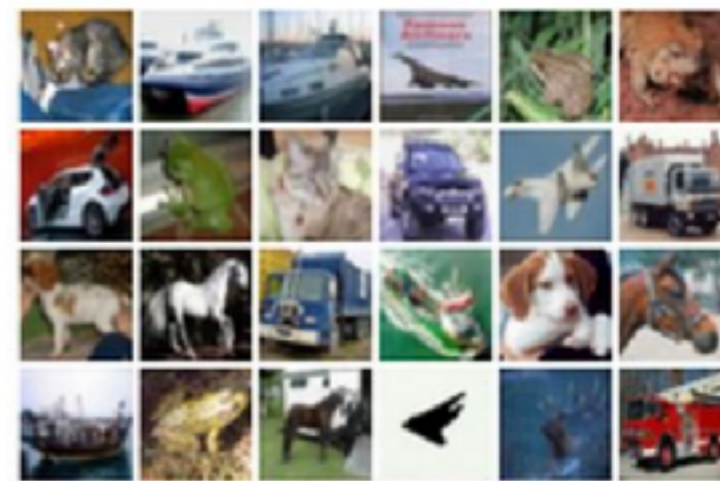
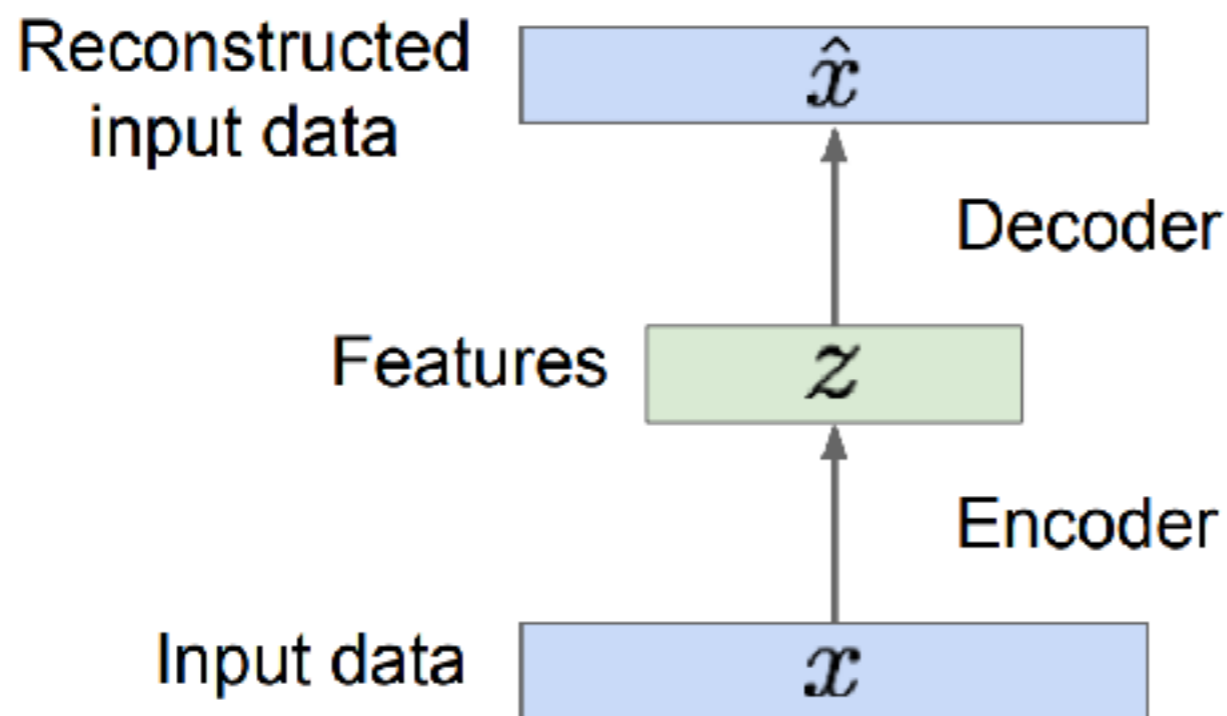


# Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data

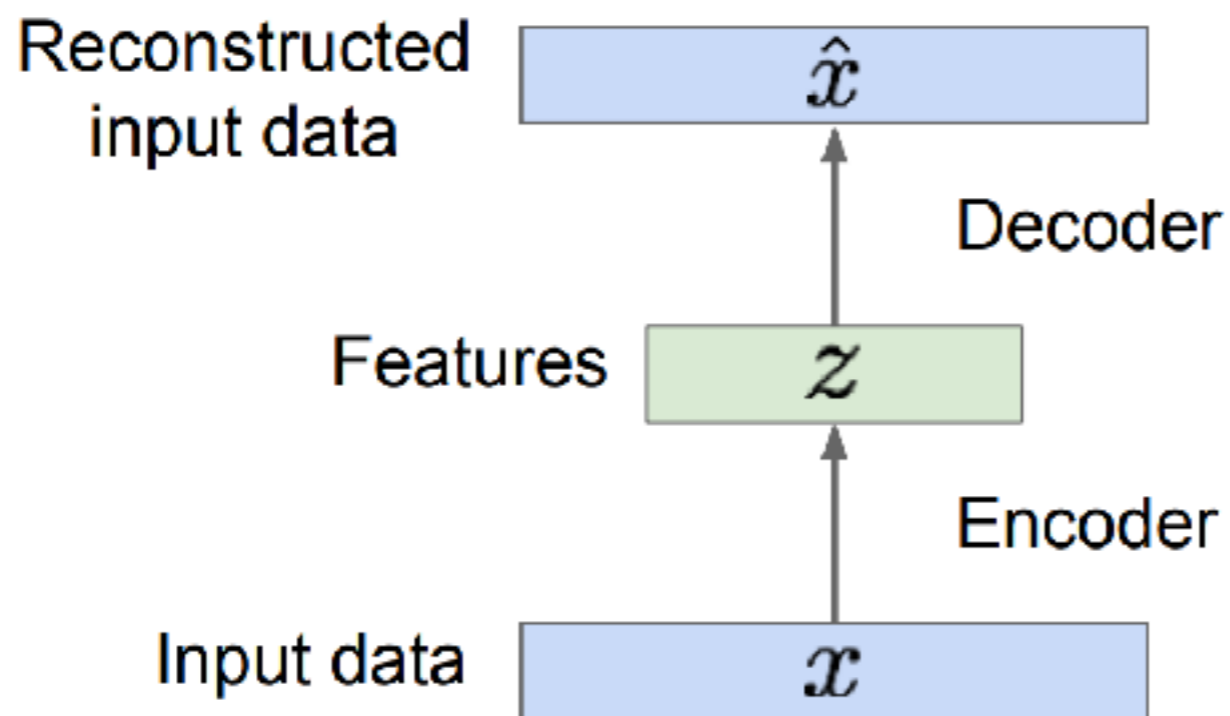
“Autoencoding” - encoding itself



# Autoencoders

How to learn this feature representation?

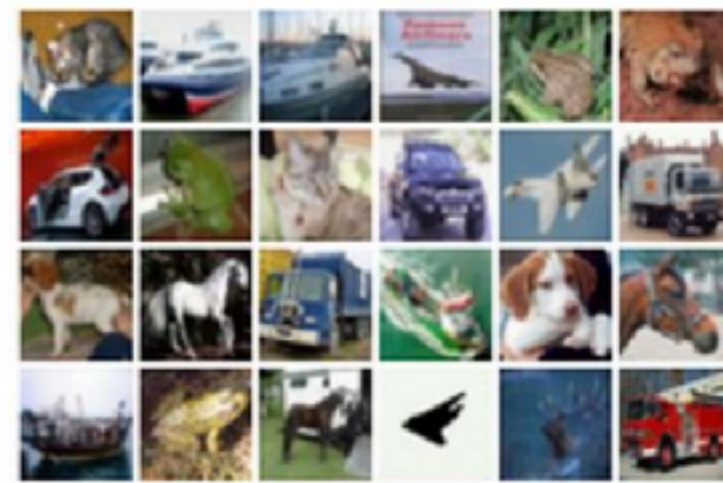
Train such that features can be used to reconstruct original data  
“Autoencoding” - encoding itself



**Originally:** Linear + nonlinearity (sigmoid)

**Later:** Deep, fully connected

**Later:** ReLU CNN (upconv)

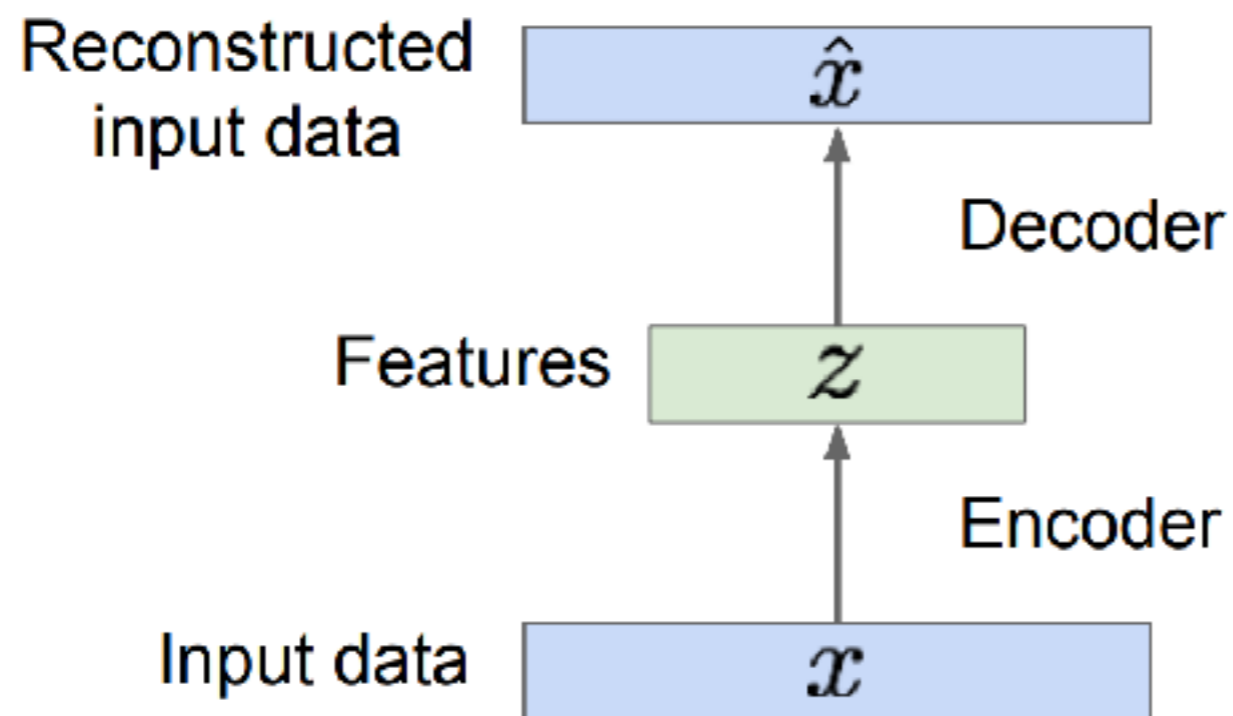


# Autoencoders

How to learn this feature representation?

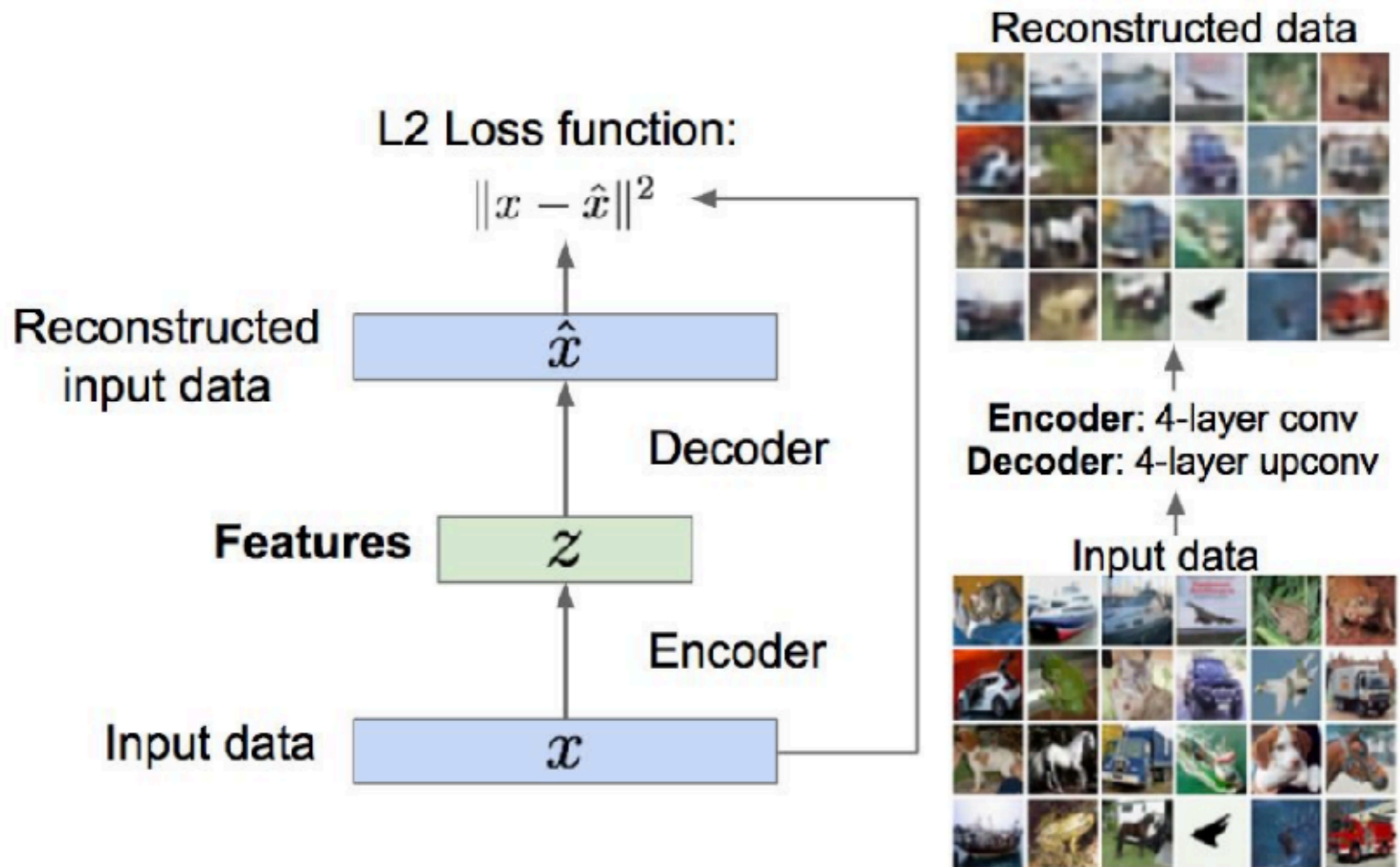
Train such that features can be used to reconstruct original data

“Autoencoding” - encoding itself



# Autoencoders

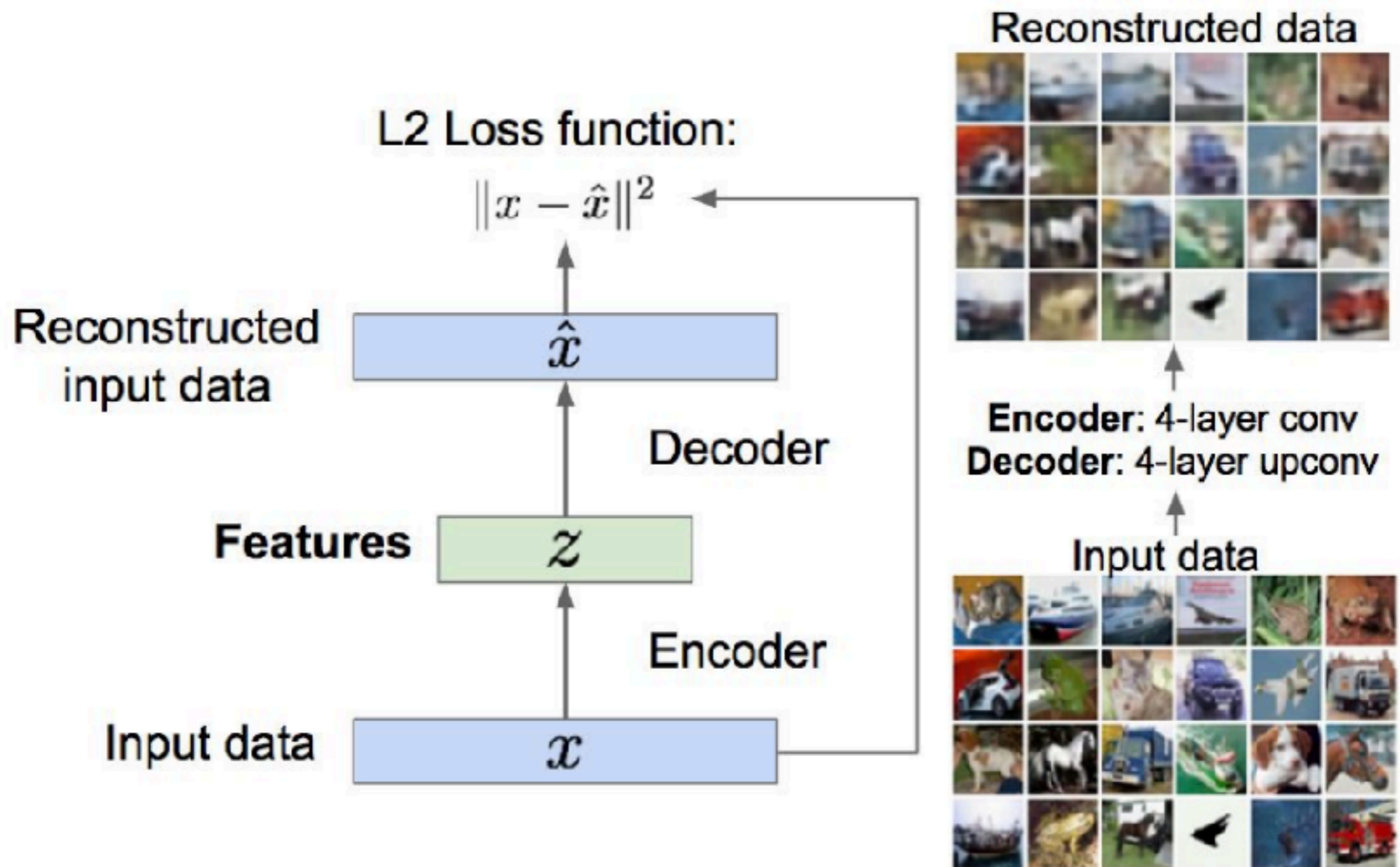
Train such that features  
can be used to  
reconstruct original data



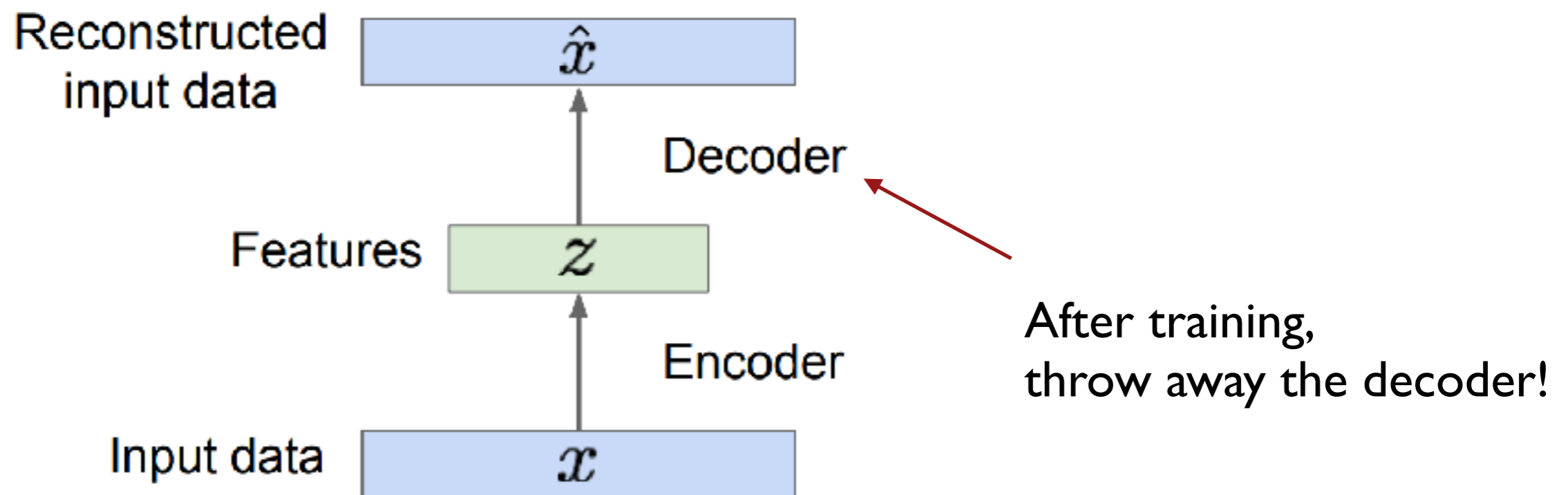
# Autoencoders

Train such that features  
can be used to  
reconstruct original data

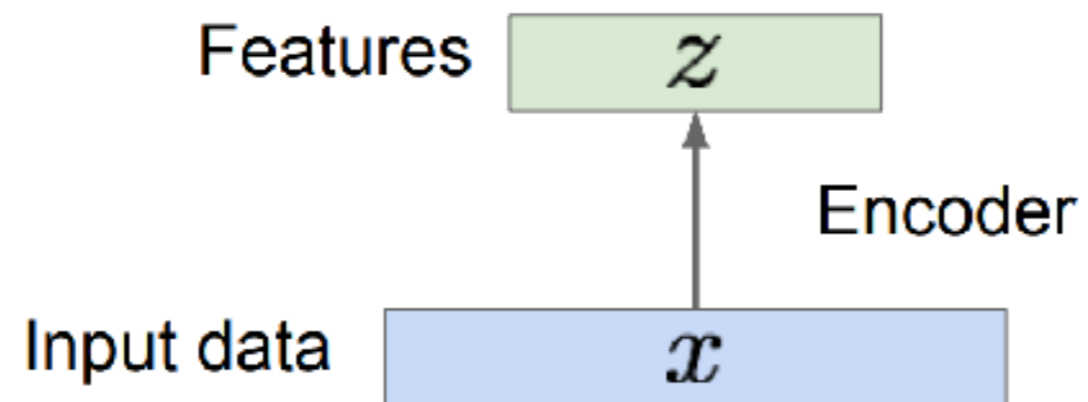
Doesn't use labels!



# Autoencoders

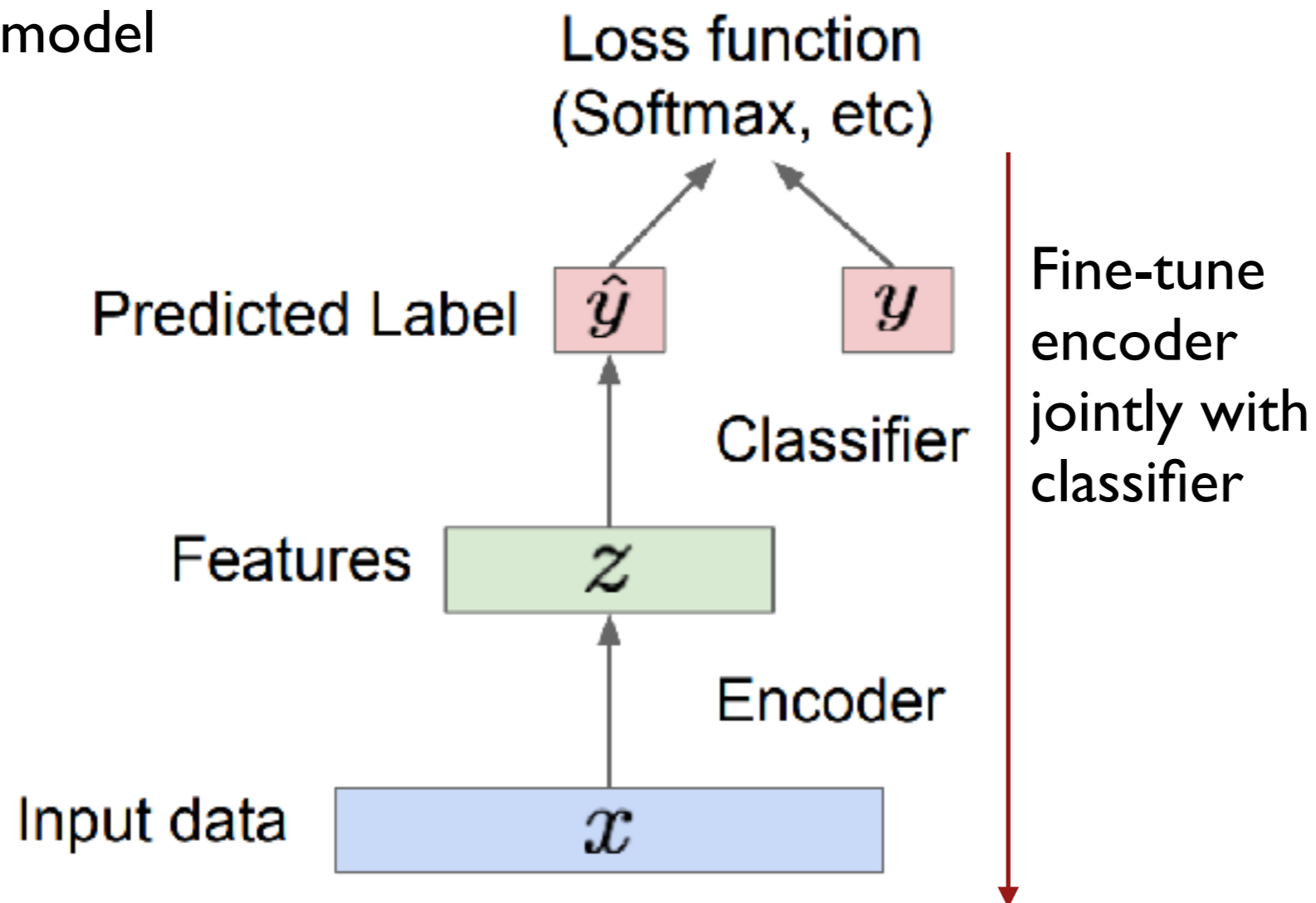


# Autoencoders



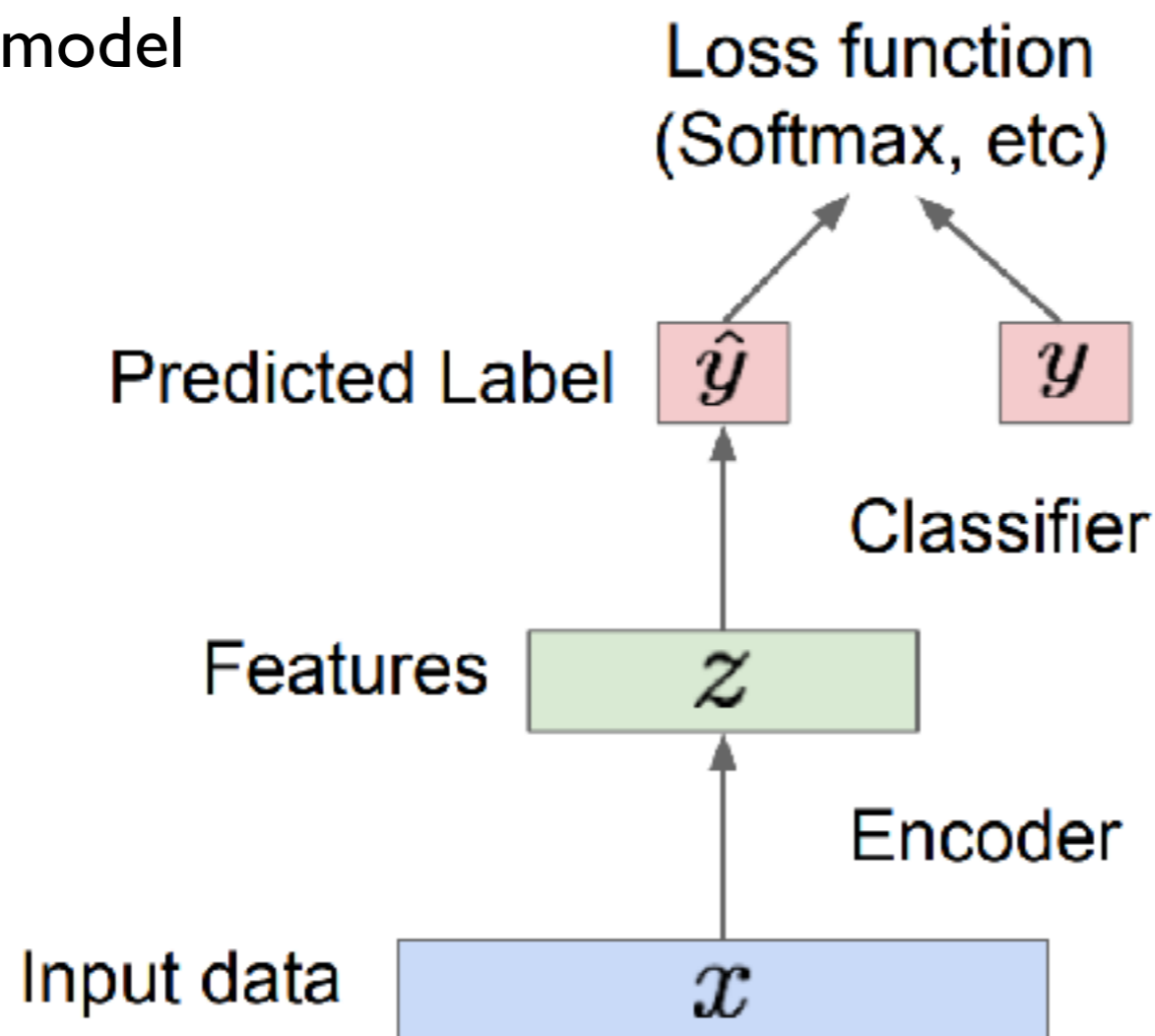
# Autoencoders

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



# Autoencoders

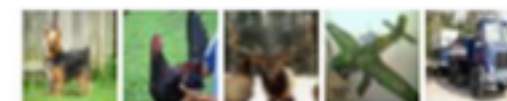
Encoder can be used to initialize a **supervised** model



Fine-tune  
encoder  
jointly with  
classifier

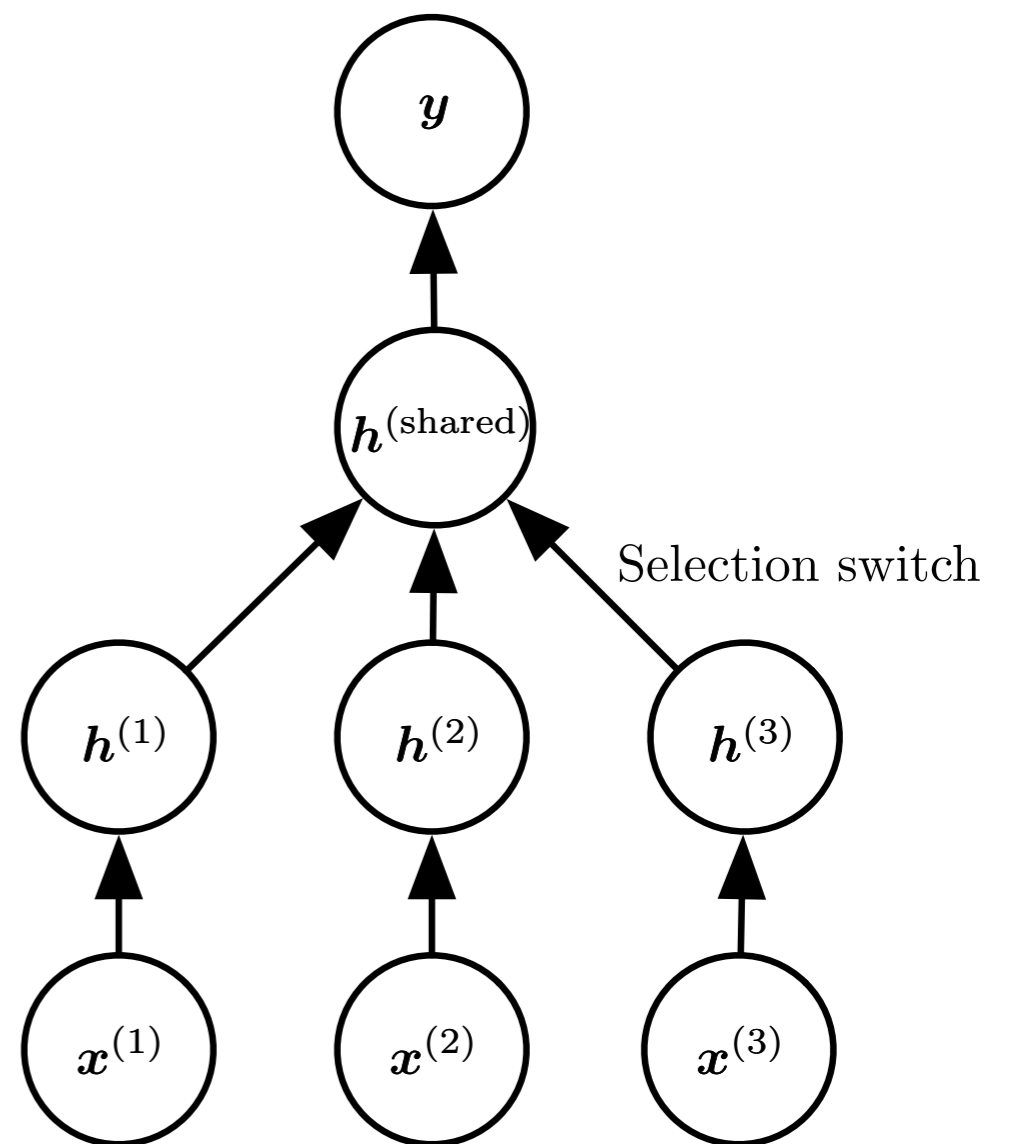
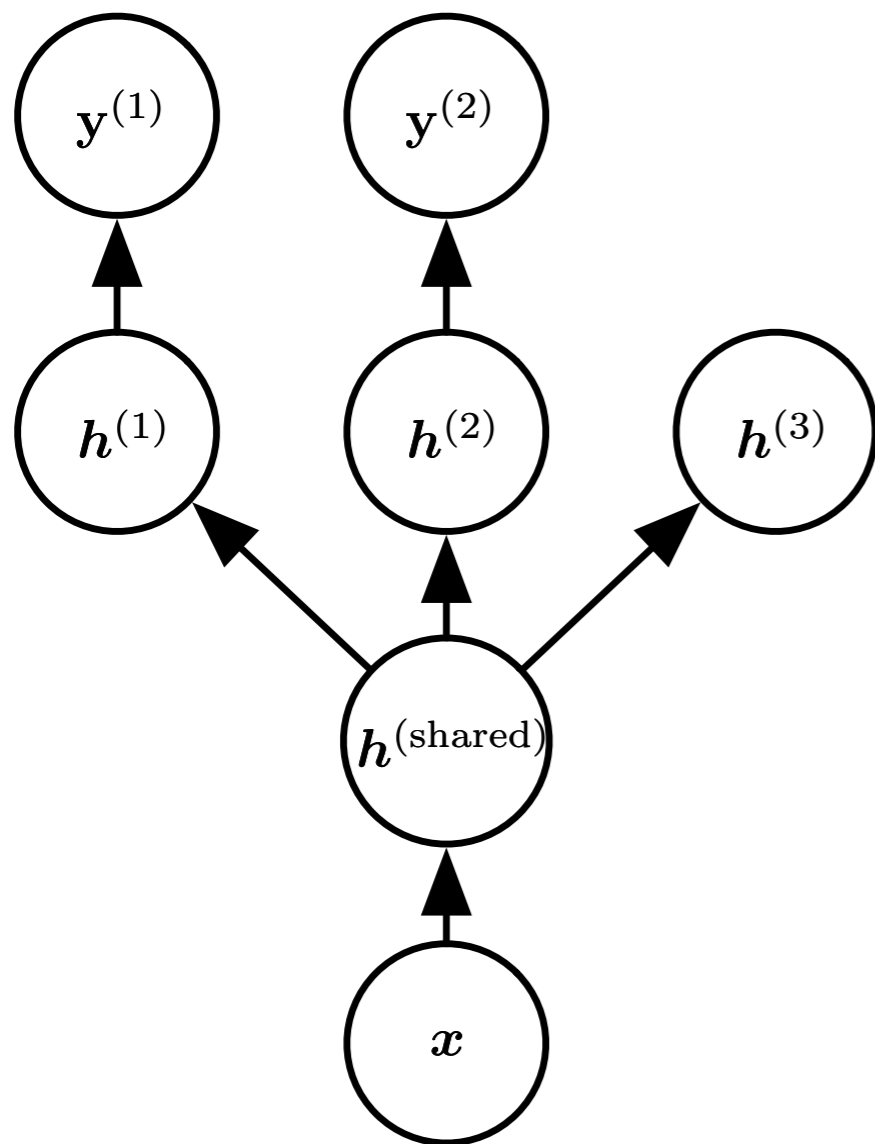
bird plane  
dog deer truck

Train for final task  
(sometimes with  
small data)



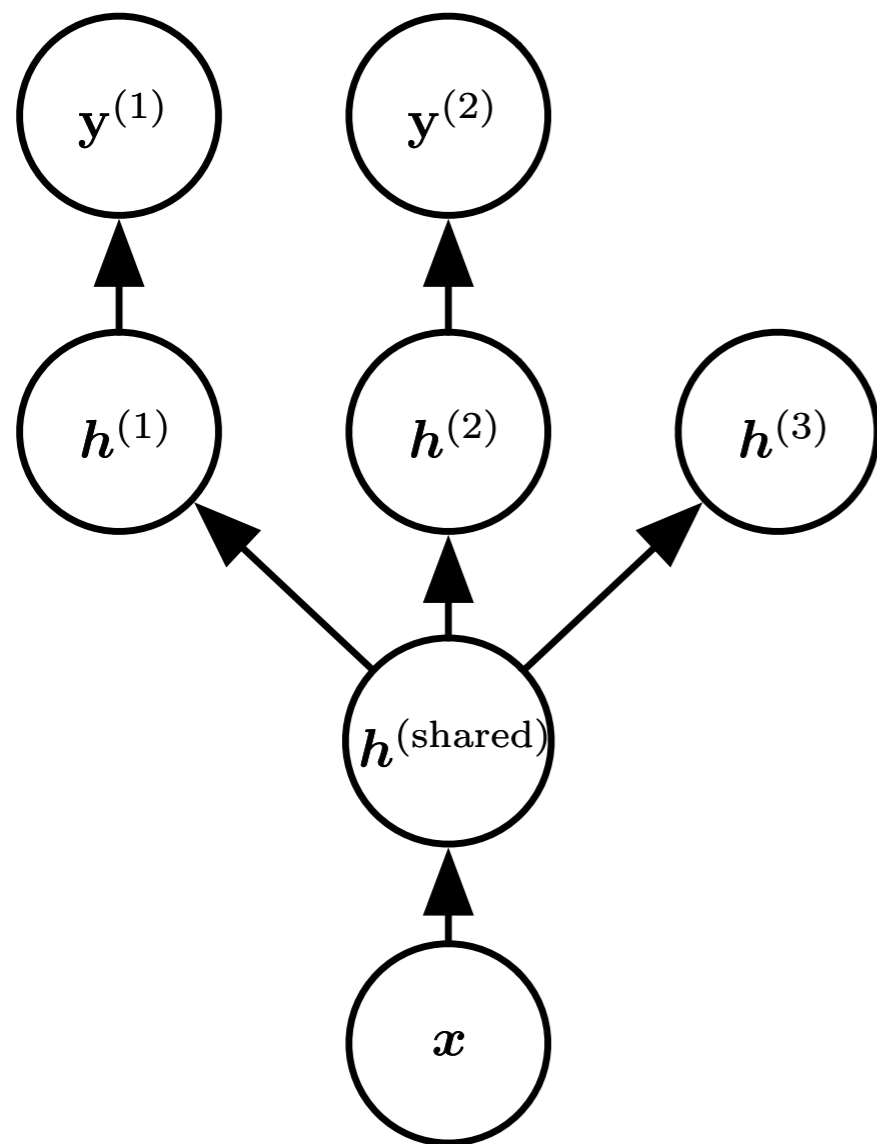
# Transfer learning (sharing representations)

One representation used for many tasks or input formats



# Transfer learning (sharing representations)

One representation used for many tasks or input formats



Task 1: women vs men

Task 2: young vs old

Input: images

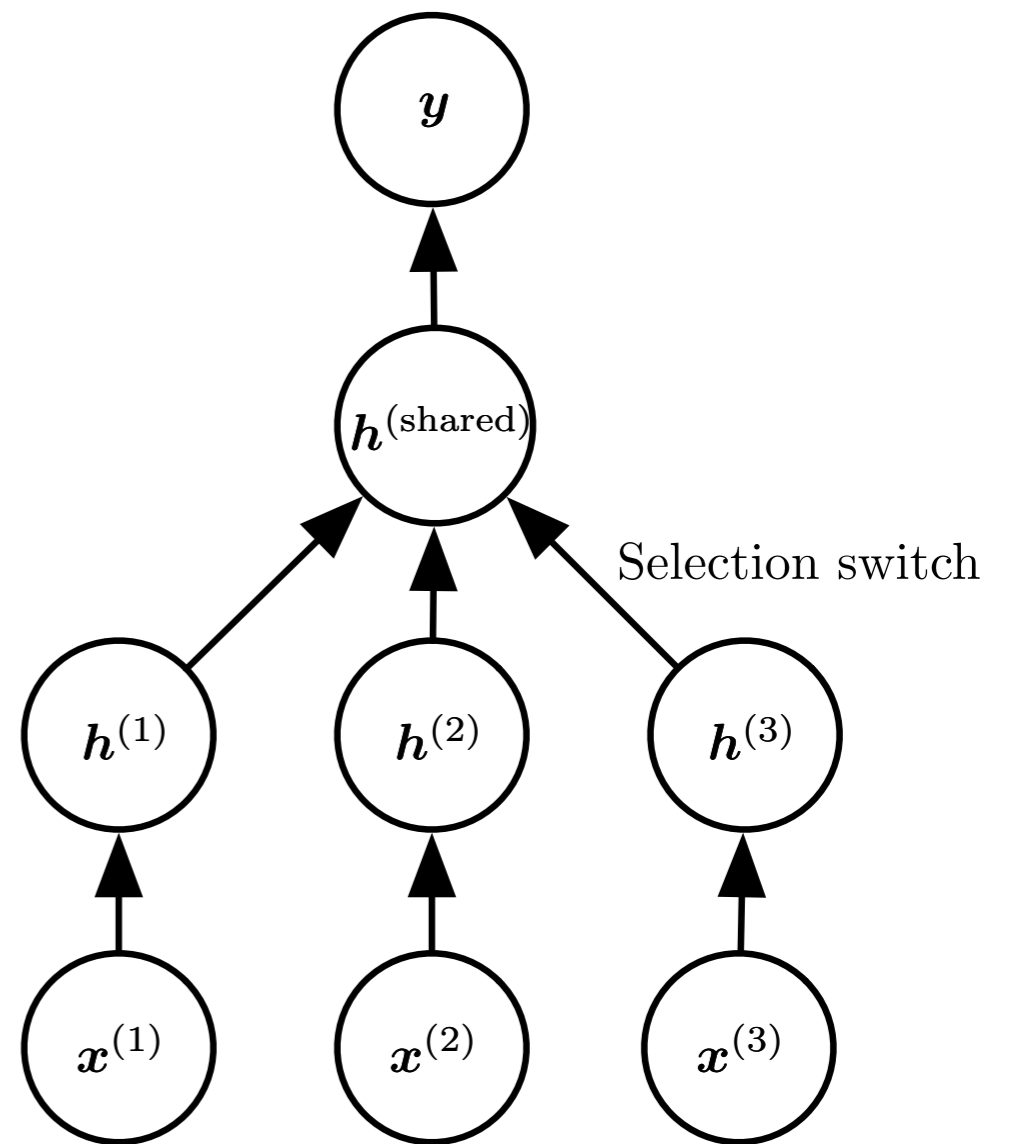
# Transfer learning (sharing representations)

One representation used for many tasks or input formats

Task: sentiment analysis

Input 1: book reviews

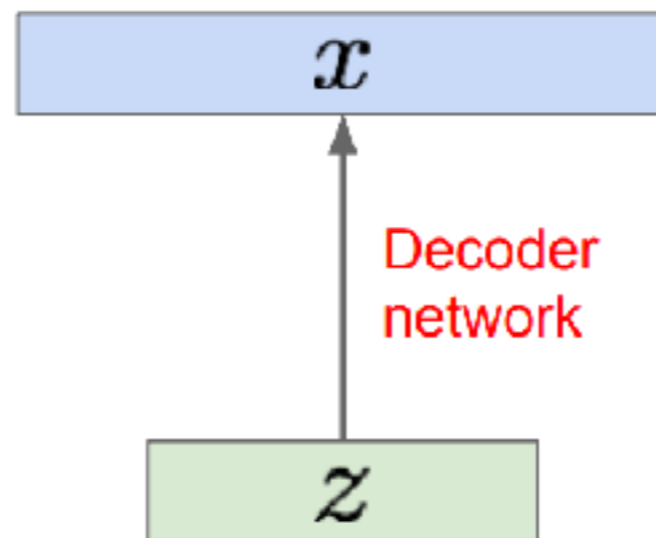
Input 2: music reviews



# Autoencoders for generating data

Autoencoders can reconstruct data, and can learn features to initialize a supervised model

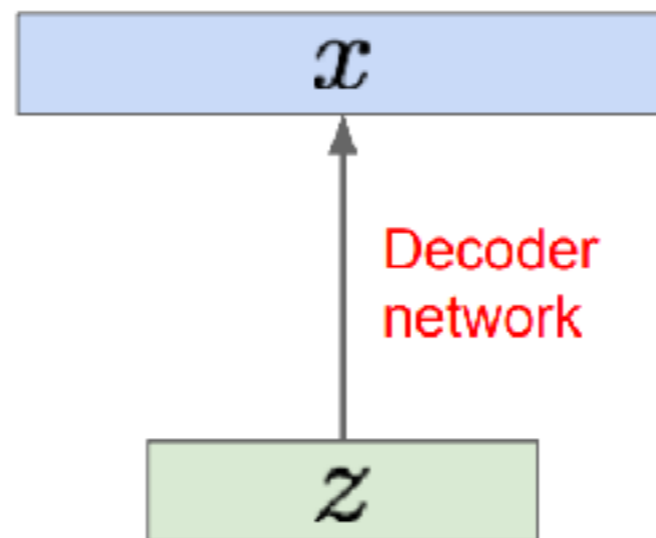
Features capture factors of variation in training data.  
Can we generate new images from an auto encoder?



# Autoencoders for generating data

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 auto encoder?



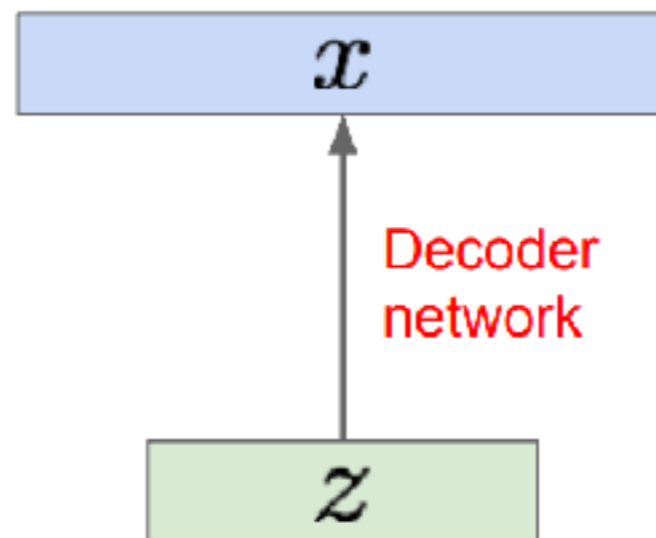
**Intuition:**  $x$  is an image,  $z$  is latent factors used to generate  $x$ : attributes, orientation, etc.

# Autoencoders for generating data

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

Autoencoders can reconstruct data, and can learn features to initialize a supervised model

Obtain  $p(x|z)$



Features capture factors of variation in training data.  
Can we generate new images from an auto encoder?

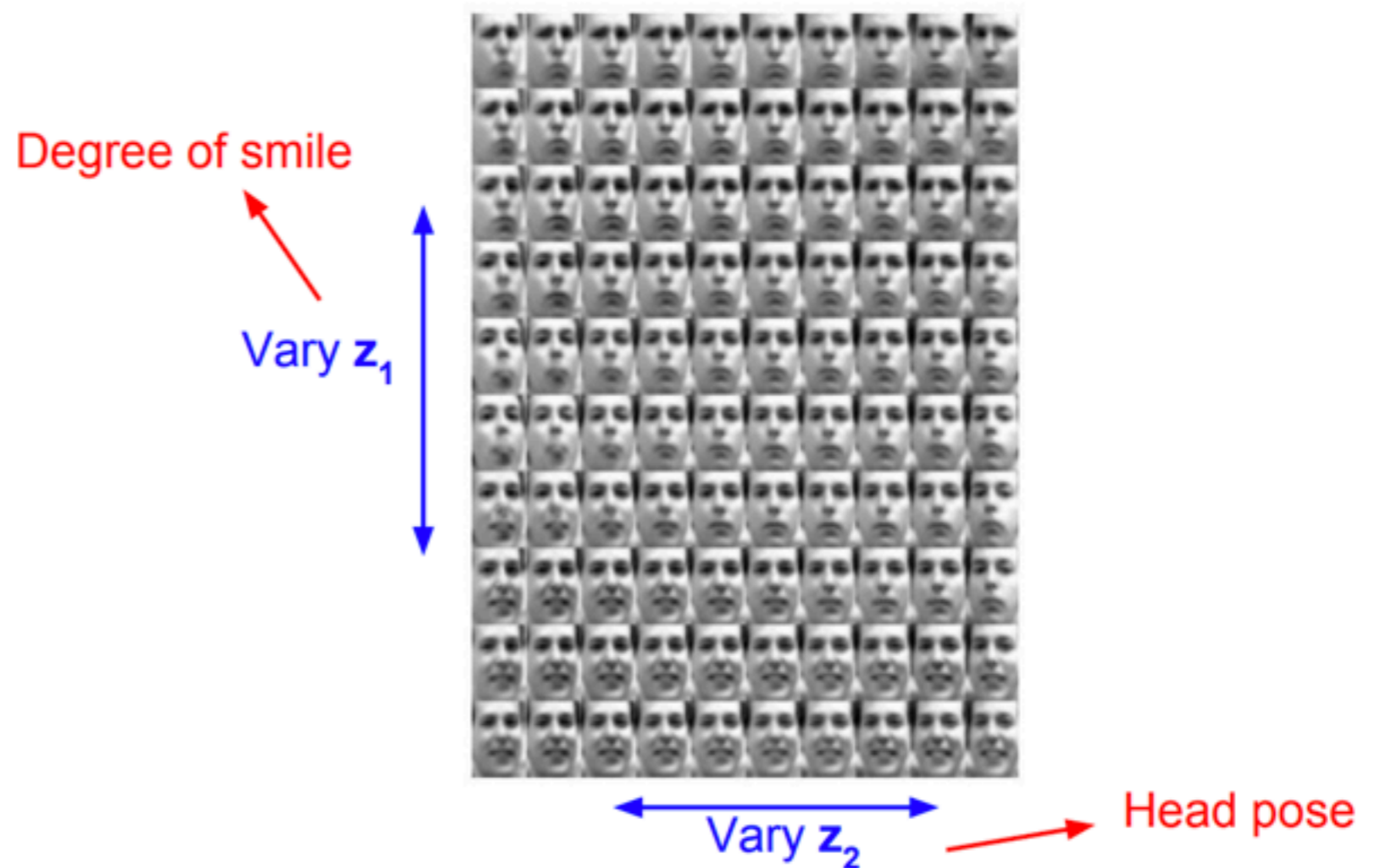
Sample from some prior  $p(z)$

**Intuition:**  $x$  is an image,  $z$  is latent factors used to generate  $x$ : attributes, orientation, etc.

# Autoencoders for generating data

Assume training data is generated from underlying unobserved (latent) representation  $\mathbf{z}$

Different dimensions of  $\mathbf{z}$   
encode  
interpretable factors  
of variation

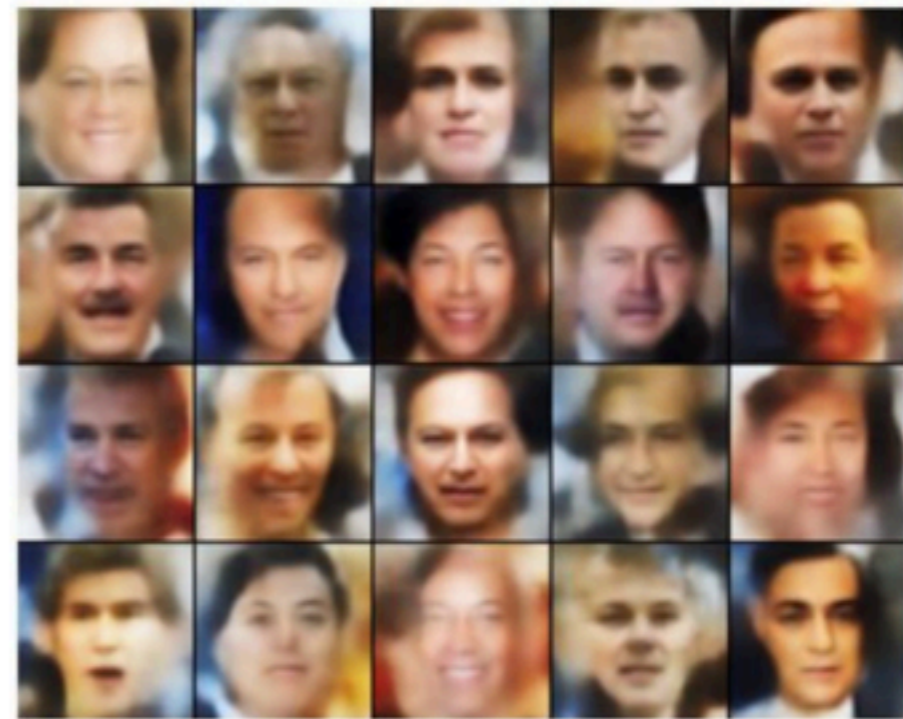


# Autoencoders for generating data

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

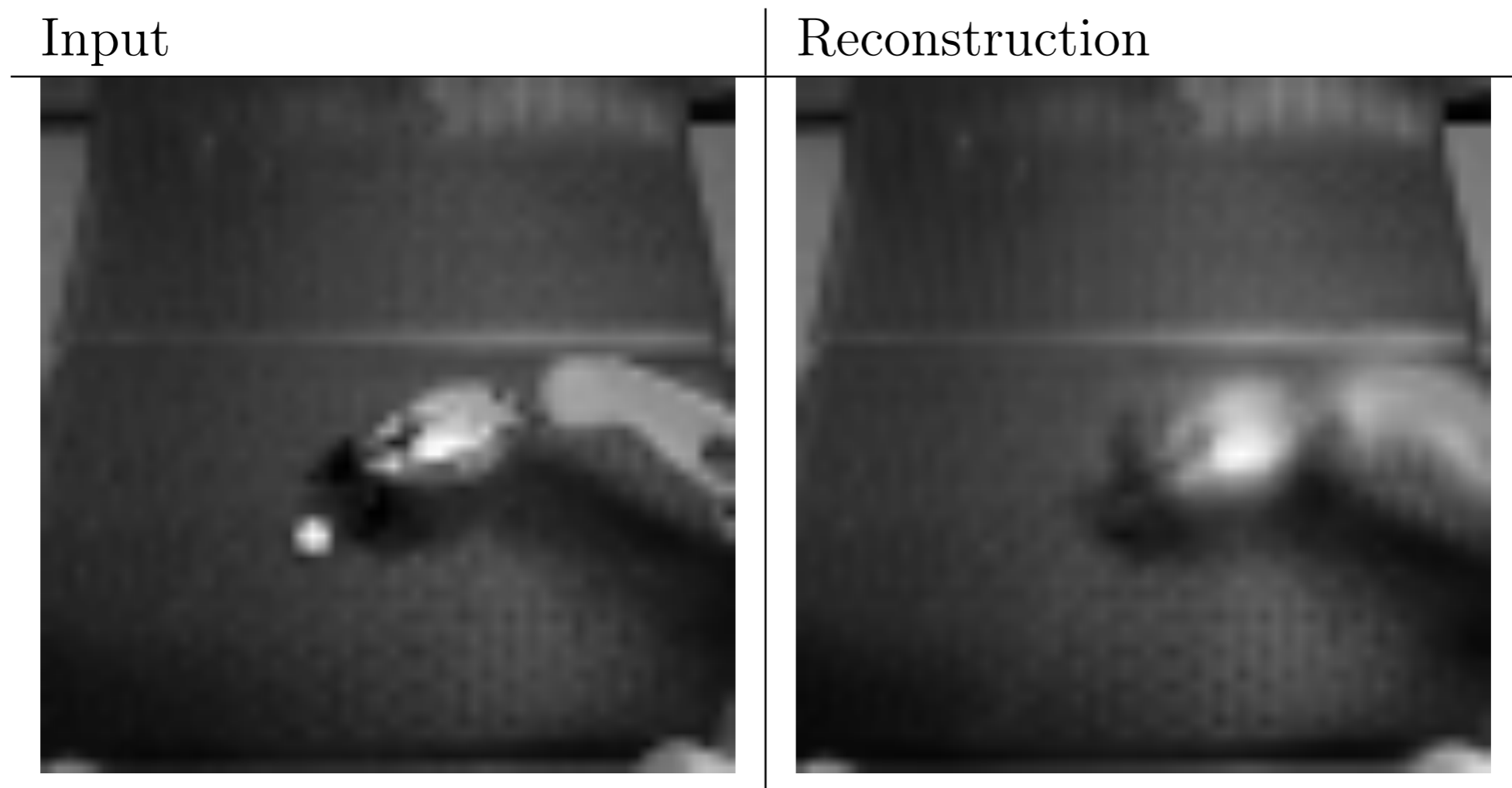


32x32 CIFAR-10



Labeled Faces in the Wild

# Mean Squared Error Can Ignore Small but Task-Relevant Features (Autoencoders)



The ping pong ball vanishes because it is not large enough to significantly affect the mean squared error

# Generative adversarial networks

**Problem:** want to sample from complex, high-dimensional training distribution. No direct way to do this!

**Solution:** sample from a simple distribution, e.g. random noise. Learn transformation to target distribution.

What can we use to  
represent this complex  
transformation?

# Generative adversarial networks

**Problem:** want to sample from complex, high-dimensional training distribution. No direct way to do this!

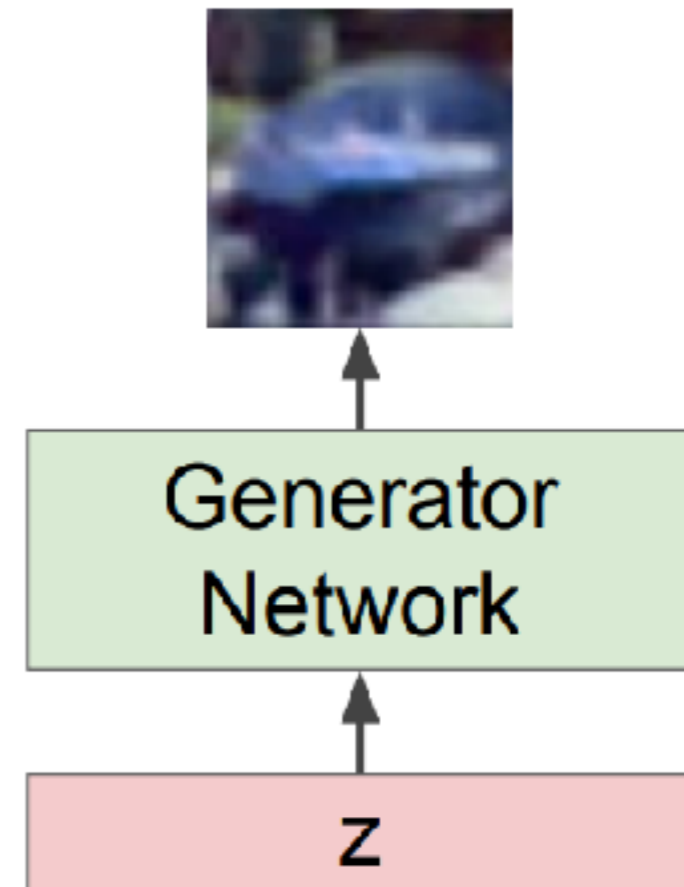
**Solution:** sample from a simple distribution, e.g. random noise. Learn transformation to target distribution.

What can we use to represent this complex transformation?

A neural network!

Output: sample from training distribution

Input: Random noise



# Training GANs: Two-player game

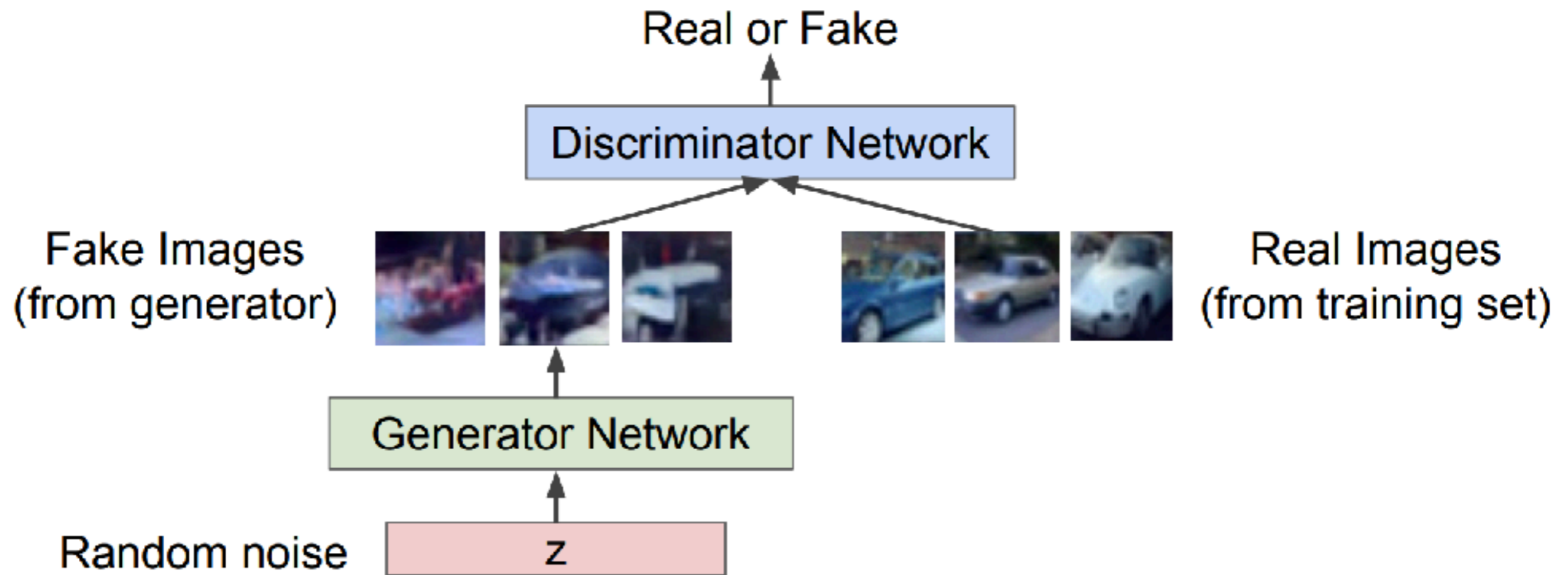
**Generator network:** try to fool the discriminator by generating real-looking pictures

**Discriminator network:** try to distinguish between real and fake images

# Training GANs: Two-player game

**Generator network:** try to fool the discriminator by generating real-looking pictures

**Discriminator network:** try to distinguish between real and fake images



# Training GANs: Two-player game

**Generator network:** try to fool the discriminator by generating real-looking pictures

**Discriminator network:** try to distinguish between real and fake images

Train jointly in minimax game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

# Training GANs: Two-player game

**Generator network:** try to fool the discriminator by generating real-looking pictures

**Discriminator network:** try to distinguish between real and fake images

Train jointly in minimax game

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\substack{\text{Discriminator output} \\ \text{for real data } x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\substack{\text{Discriminator output for} \\ \text{for generated fake data } G(z)}}) \right]$$

# Training GANs: Two-player game

**Generator network:** try to fool the discriminator by generating real-looking pictures

**Discriminator network:** try to distinguish between real and fake images

Train jointly in minimax game

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \underbrace{\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x)}_{\text{Discriminator output for real data } x} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data } G(z)}) \right]$$

- Discriminator wants to **maximize objective** such that  $D(x)$  is close to 1 (real) and  $D(G(z))$  is close to 0 (fake)
- Generator wants to **minimize objective** such that  $D(G(z))$  is close to 1 (discriminator is fooled into thinking generated  $G(z)$  is real)

# Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1 Gradient ascent on discriminator

$$\max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2 Gradient descent on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

# Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1 Gradient ascent on discriminator

$$\max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2 **Instead:** Gradient **ascent** on generator, **different objective**

$$\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$$

# Training GANs: Two-player game

Putting it together: GAN training algorithm

**for** number of training iterations **do**

**for**  $k$  steps **do**

- Sample minibatch of  $m$  noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$  from noise prior  $p_g(z)$ .
- Sample minibatch of  $m$  examples  $\{x^{(1)}, \dots, x^{(m)}\}$  from data generating distribution  $p_{\text{data}}(x)$ .
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[ \log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]$$

**end for**

- Sample minibatch of  $m$  noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$  from noise prior  $p_g(z)$ .
- Update the generator by ascending its stochastic gradient (improved objective):

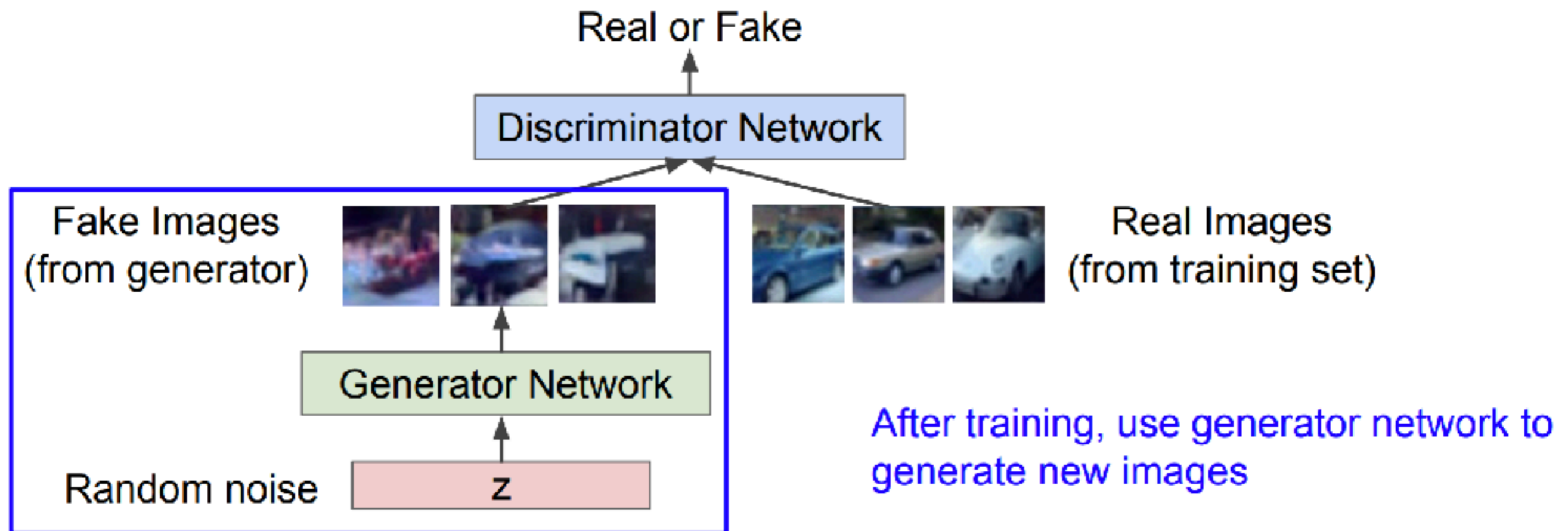
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))$$

**end for**

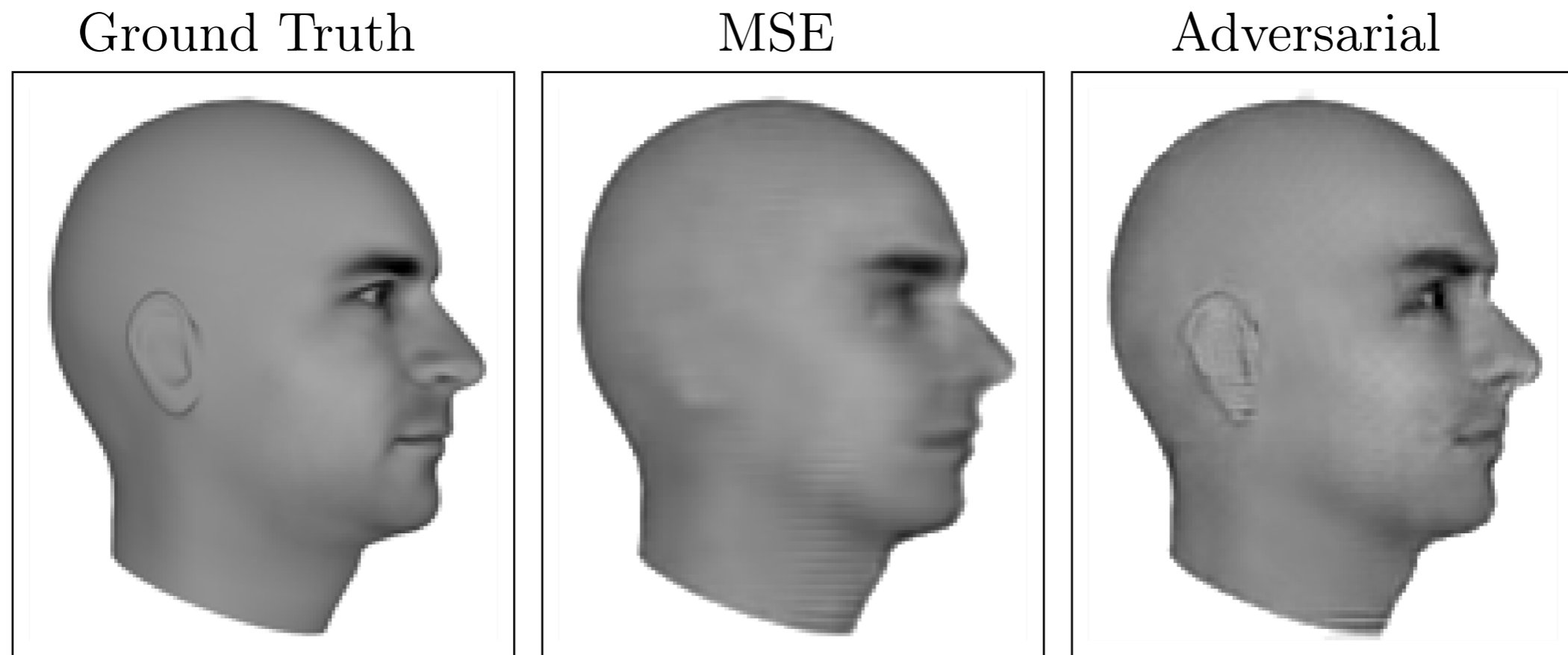
# Training GANs: Two-player game

**Generator network:** try to fool the discriminator by generating real-looking pictures

**Discriminator network:** try to distinguish between real and fake images



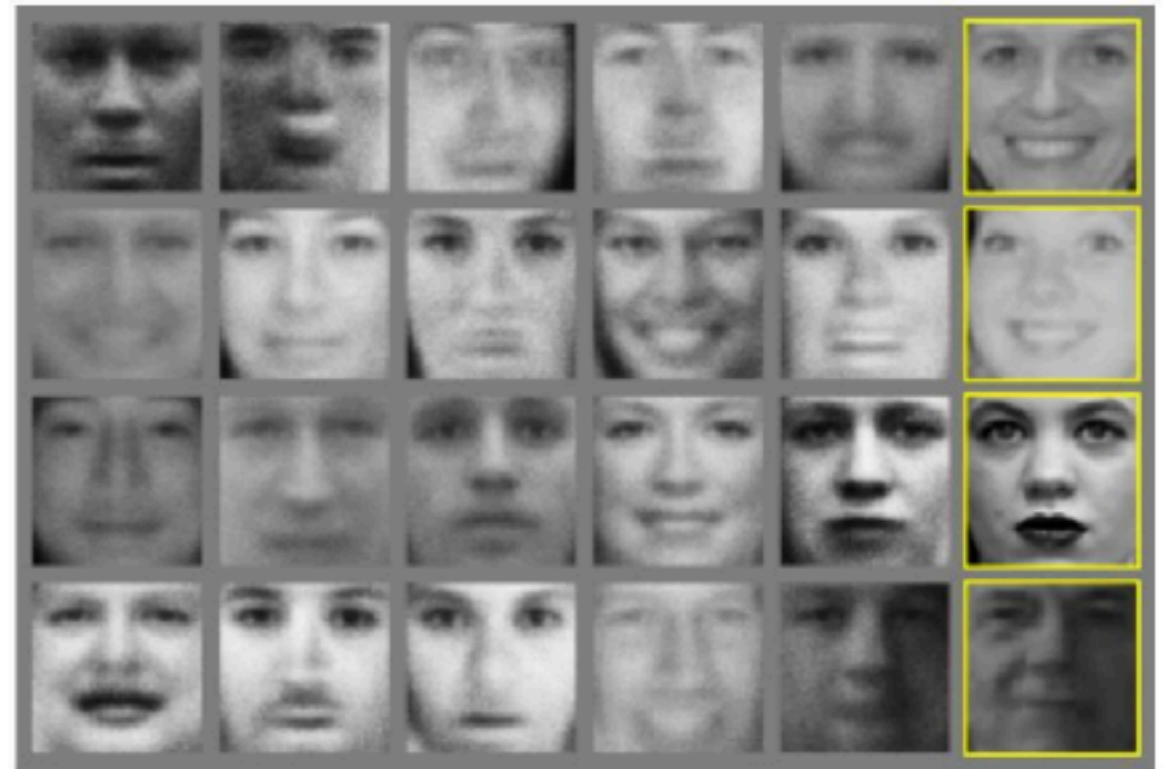
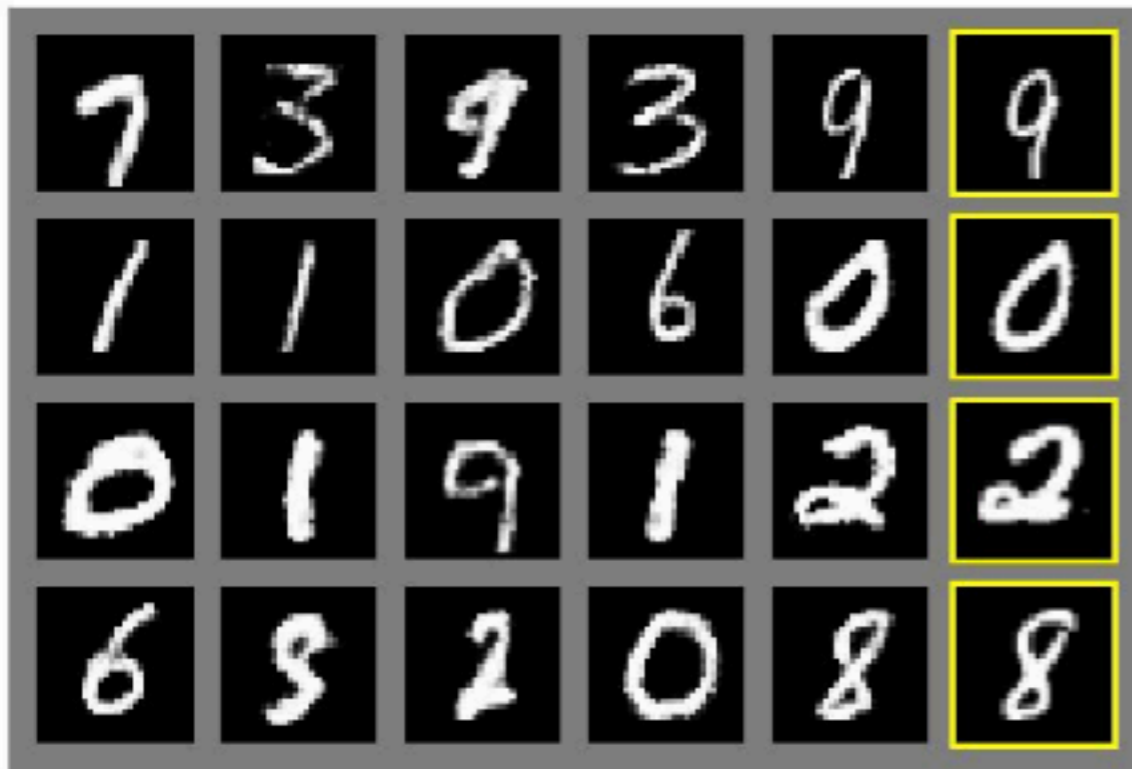
# Adversarial Losses Preserve Any Features with Highly Structured Patterns



Mean squared error loses the ear because it causes a small change in few pixels. Adversarial loss preserves the ear because it is easy to notice its absence.

# GANs

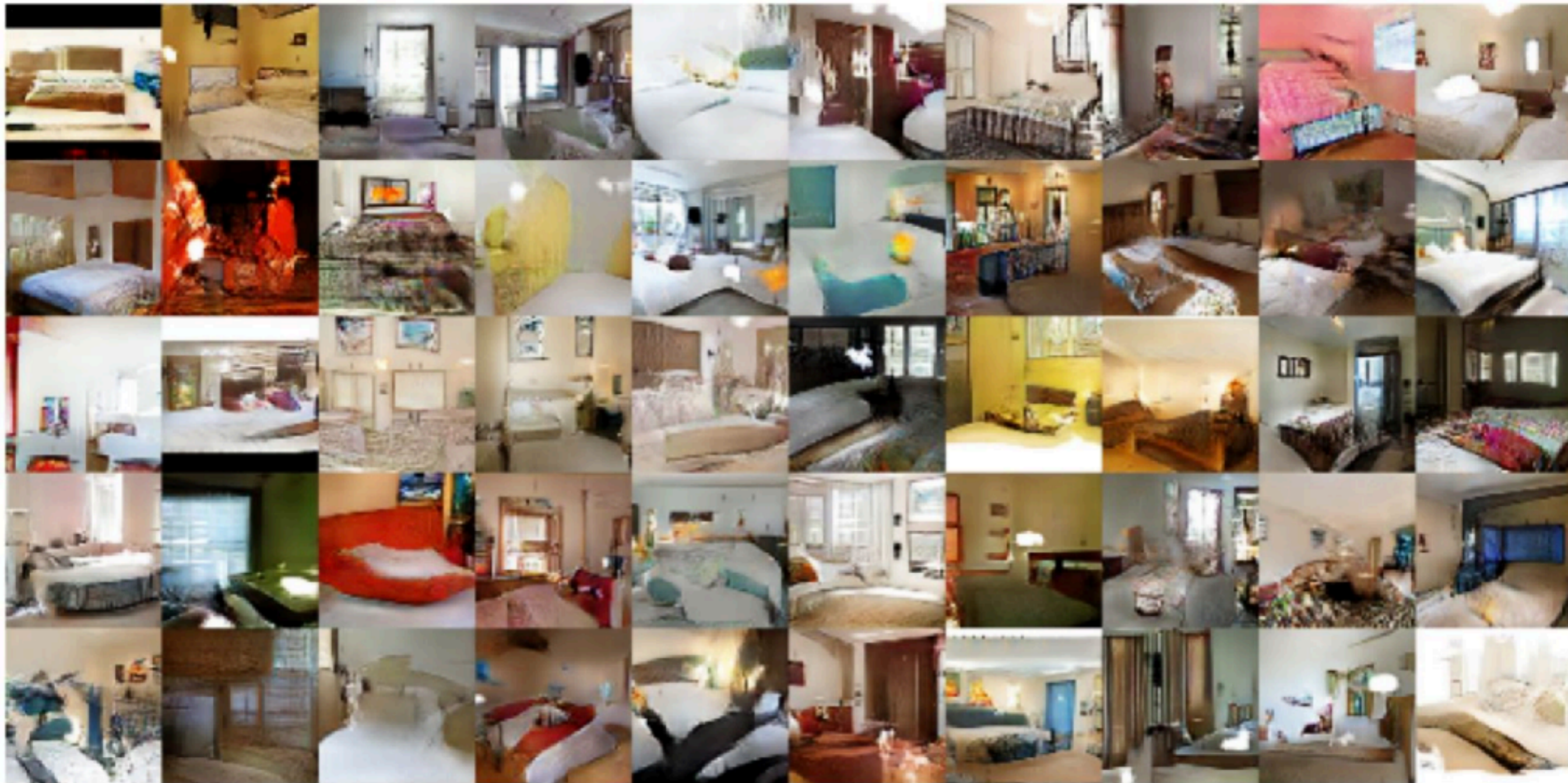
## Generated samples



Nearest neighbours from  
training set

# GANs

Generated samples



# GANs learn vector spaces that support semantic arithmetic

Glasses  
man



No glasses  
man



No glasses  
woman

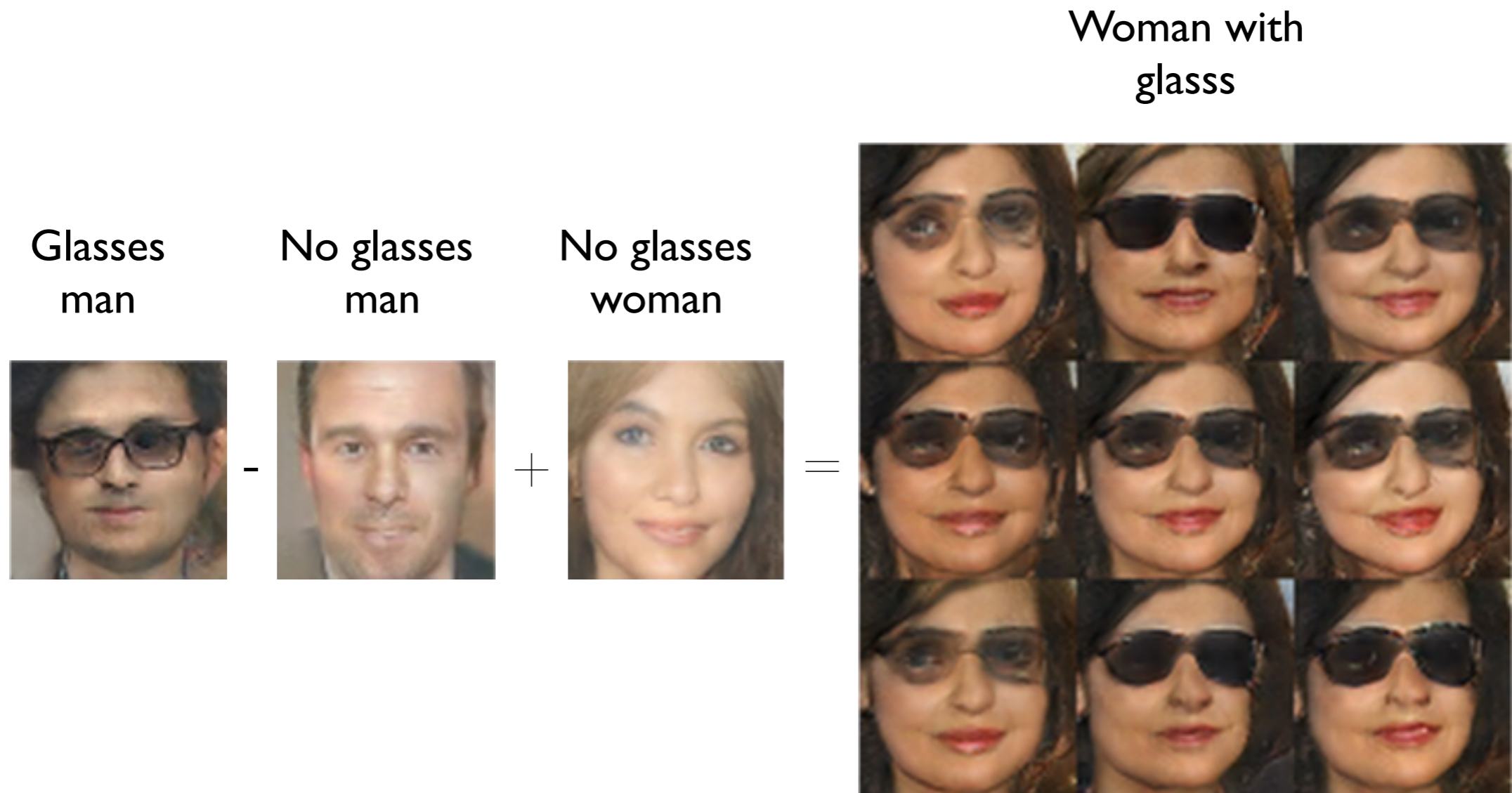


-

+

=

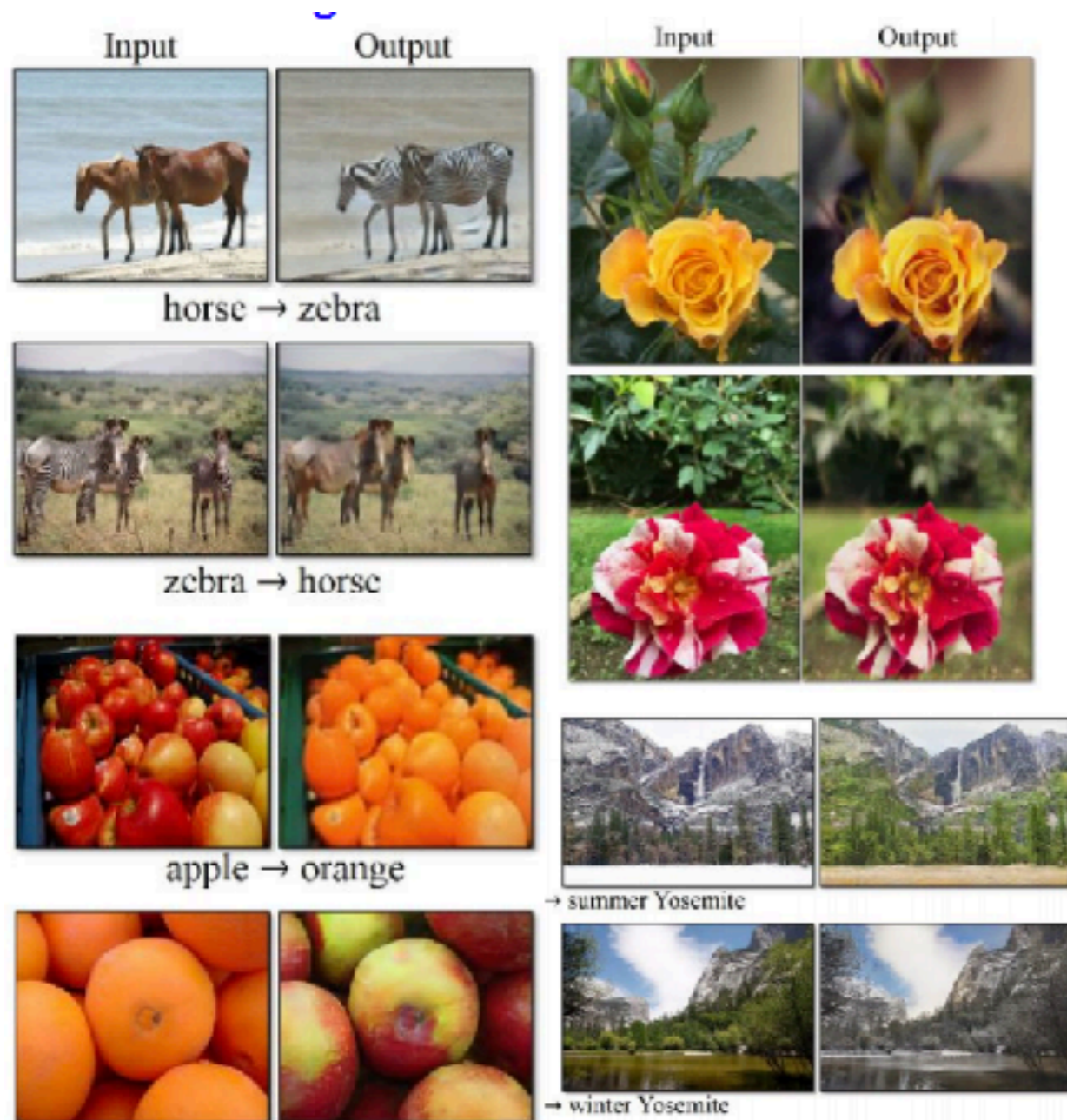
# GANs learn vector spaces that support semantic arithmetic



# 2017: year of the GAN

## Many GAN applications

Source-> Target domain transfer



Text -> images synthesis

this small bird has a pink  
breast and crown, and black  
primaries and secondaries.



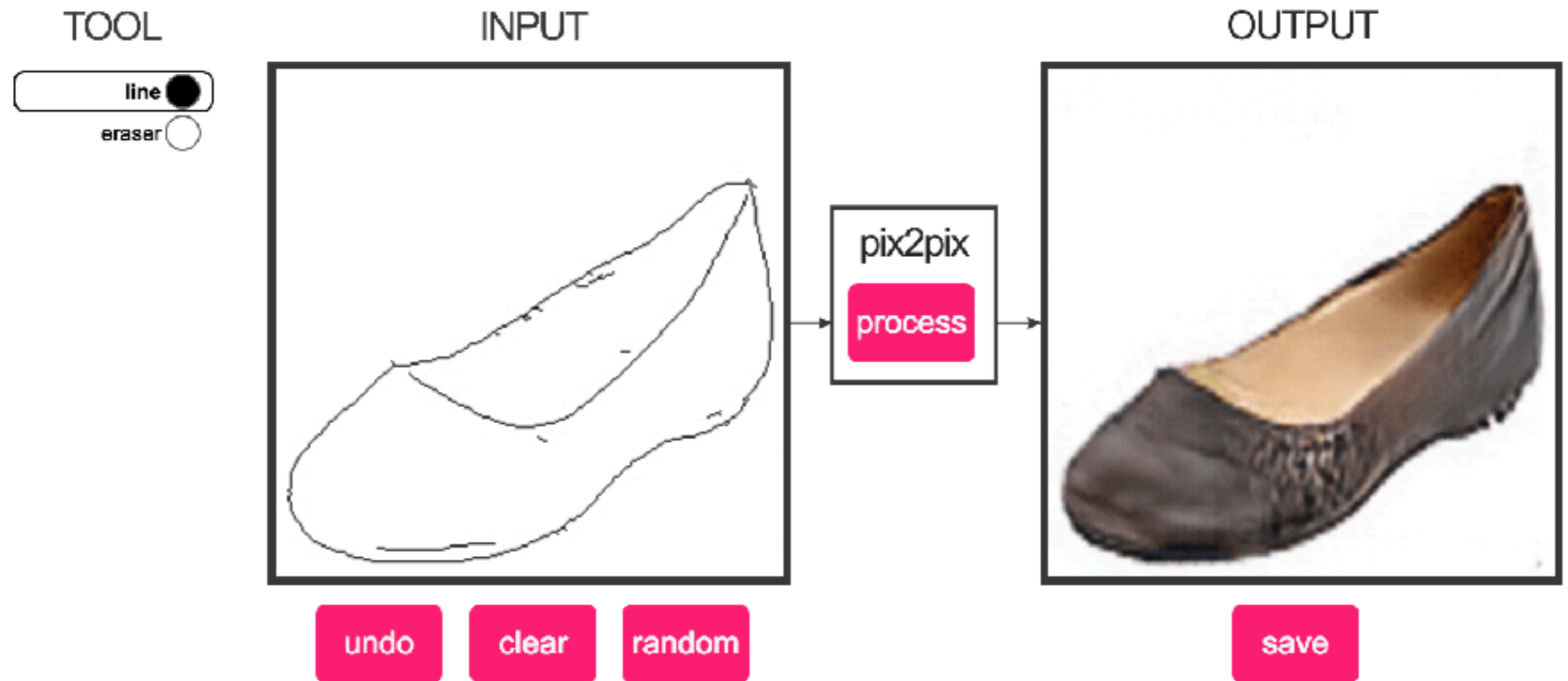
this magnificent fellow is  
almost all black with a red  
crest, and white cheek patch.



Fill in your image!

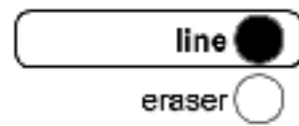


# edges2shoes



# edges2handbags

TOOL



INPUT

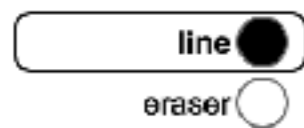


OUTPUT

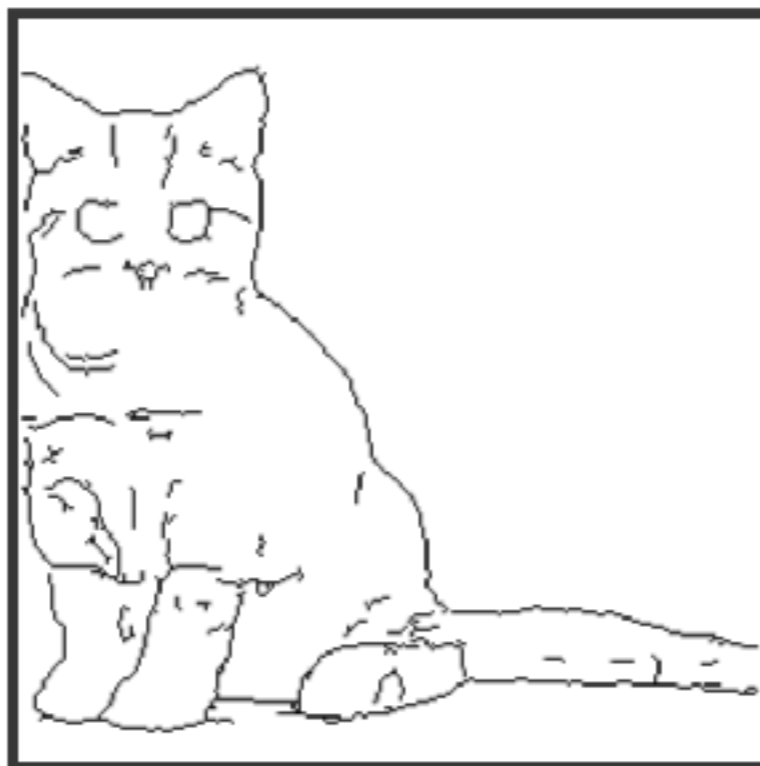


# edges2cats

TOOL



INPUT



pix2pix

process

OUTPUT

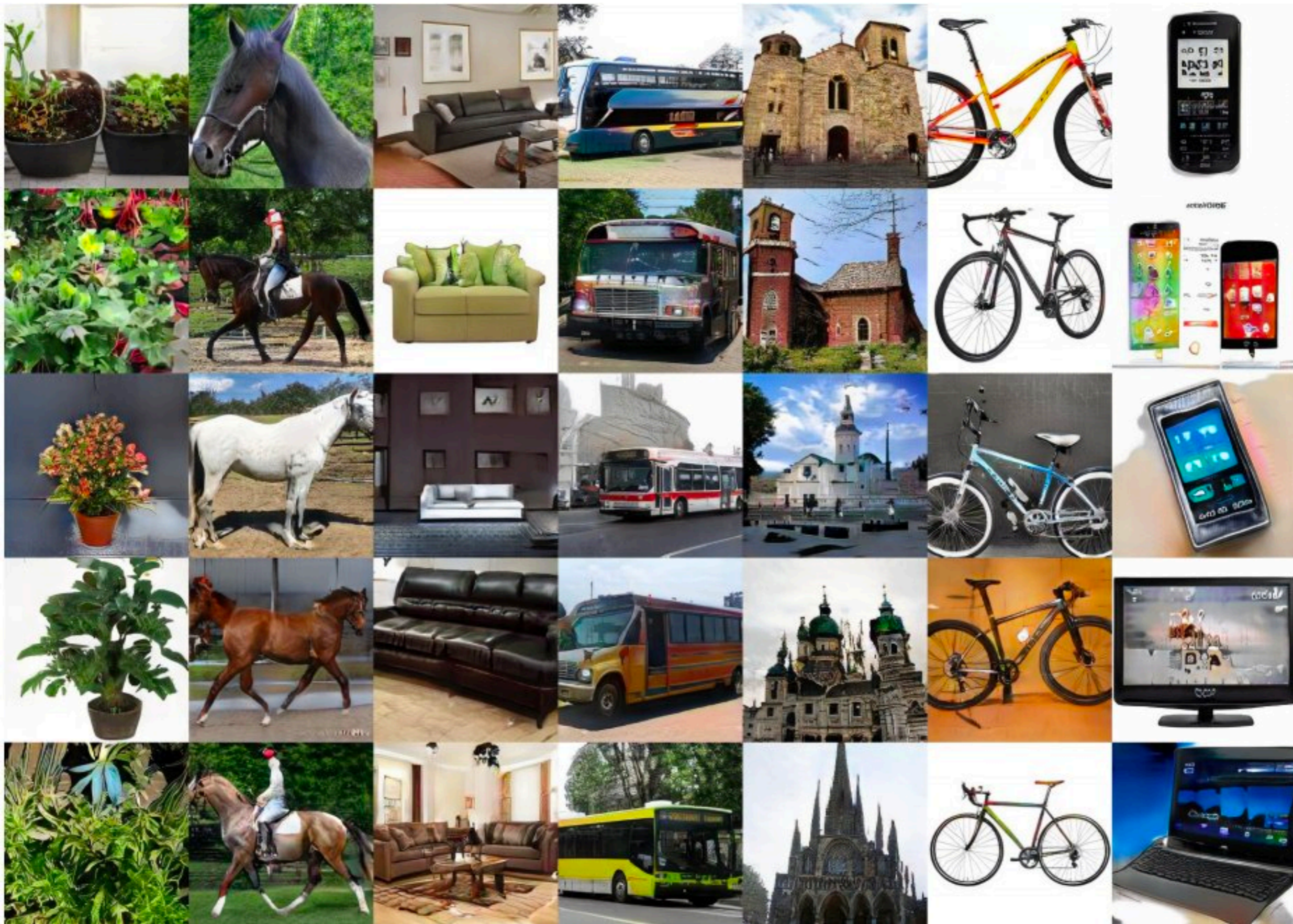


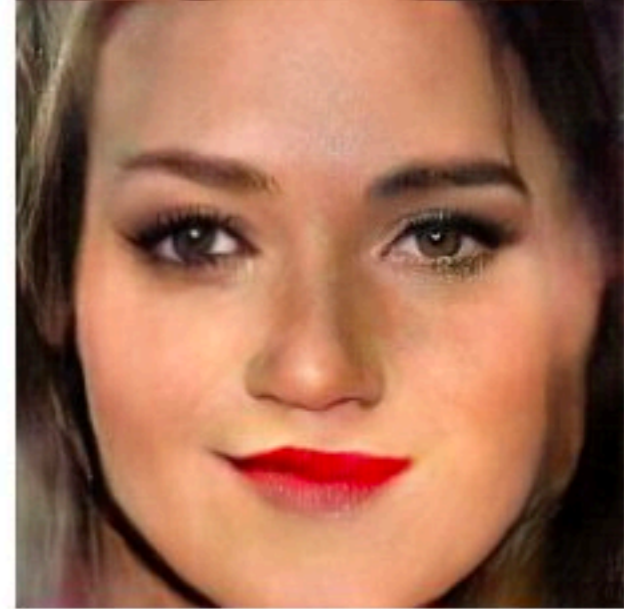
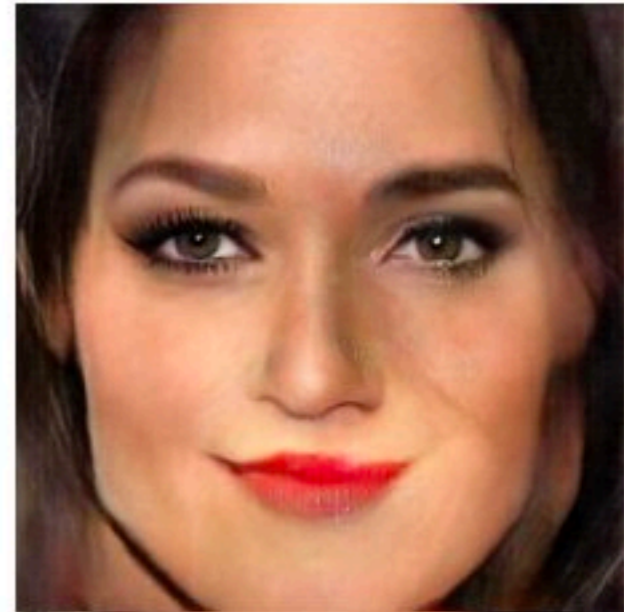
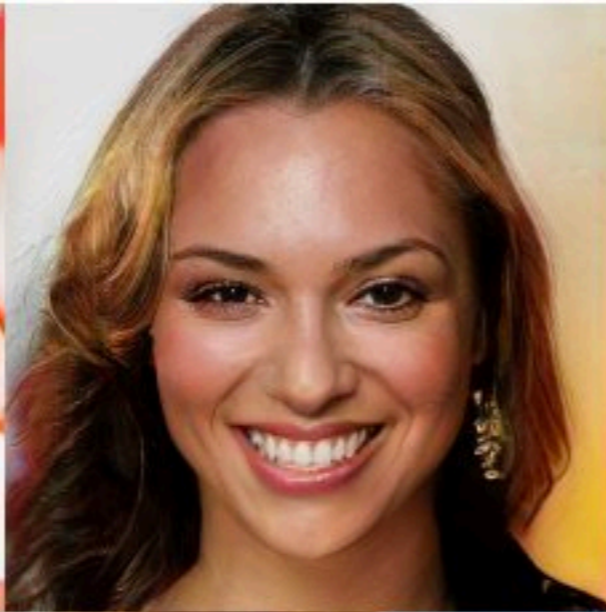
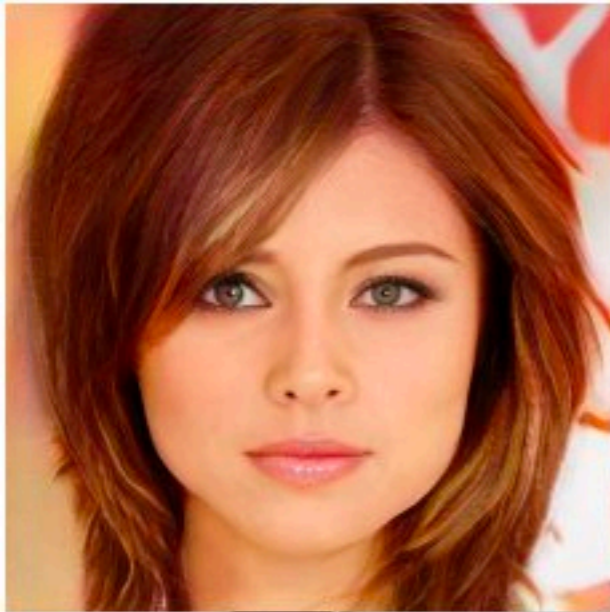
undo

clear

random

save





Lesson	Title
1	Introduction
2	Probability and Information Theory
3	Basics of Machine Learning
4	Deep Feed-forward networks
5	Back-propagation
6	Optimization and regularization
7	Convolutional Networks
8	Sequence Modeling: recurrent networks
9	Applications
10	Software and Practical methods
11	Autoencoders & GANs
12	Project Development
13	Deep Reinforcement Learning
14	Deep Learning and the Brain
15	Conclusions and Future Perspective
16	Project presentations

Foundations

Basics

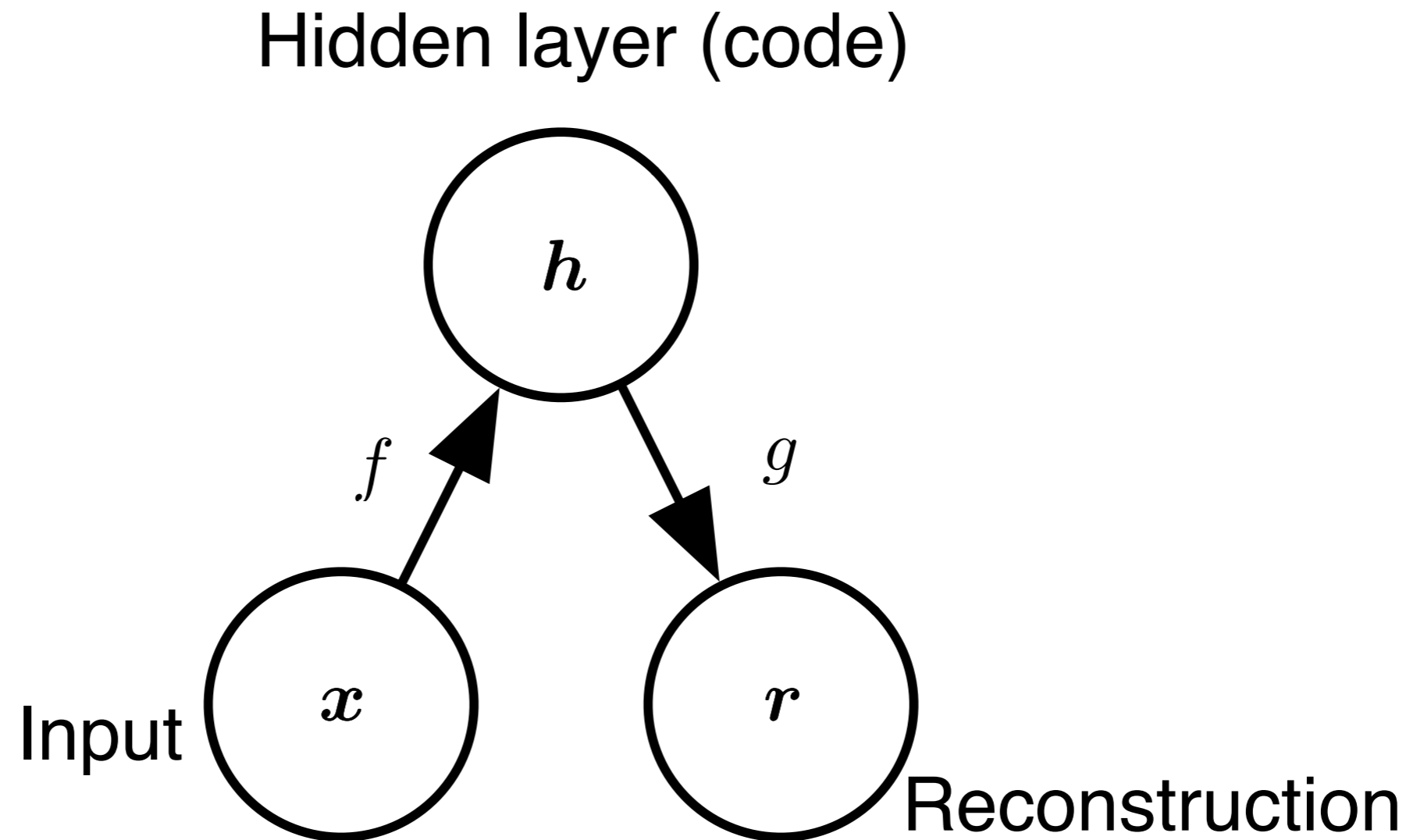
Advanced

Selected topics

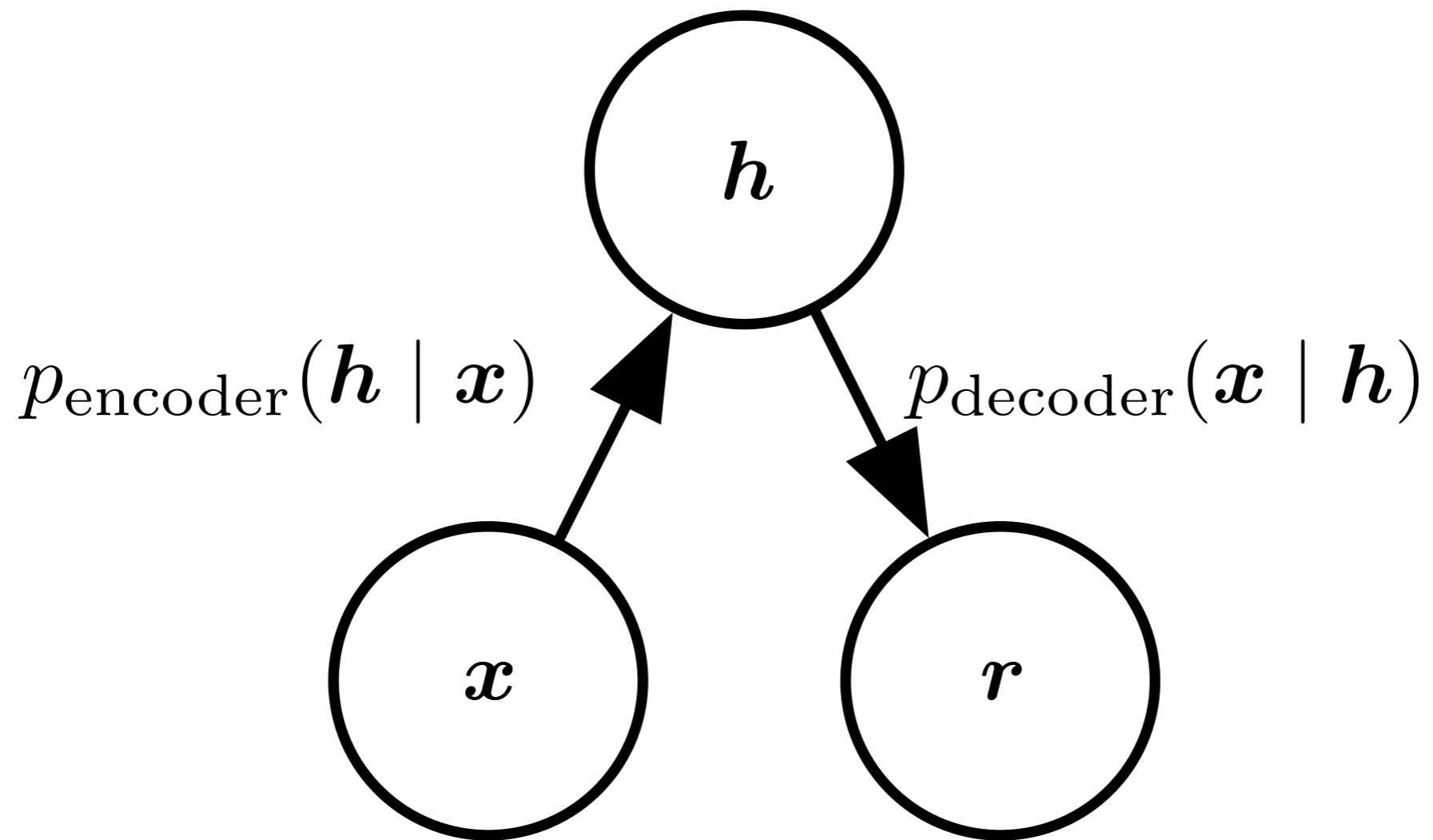




# Structure of an autoencoder



# Structure of an autoencoder



# Avoiding trivial identity

- **Undercomplete autoencoders**

- $h$  has lower dimension than  $x$
- $f$  or  $g$  has low capacity (e.g., linear  $g$ )
- Must discard some information in  $h$

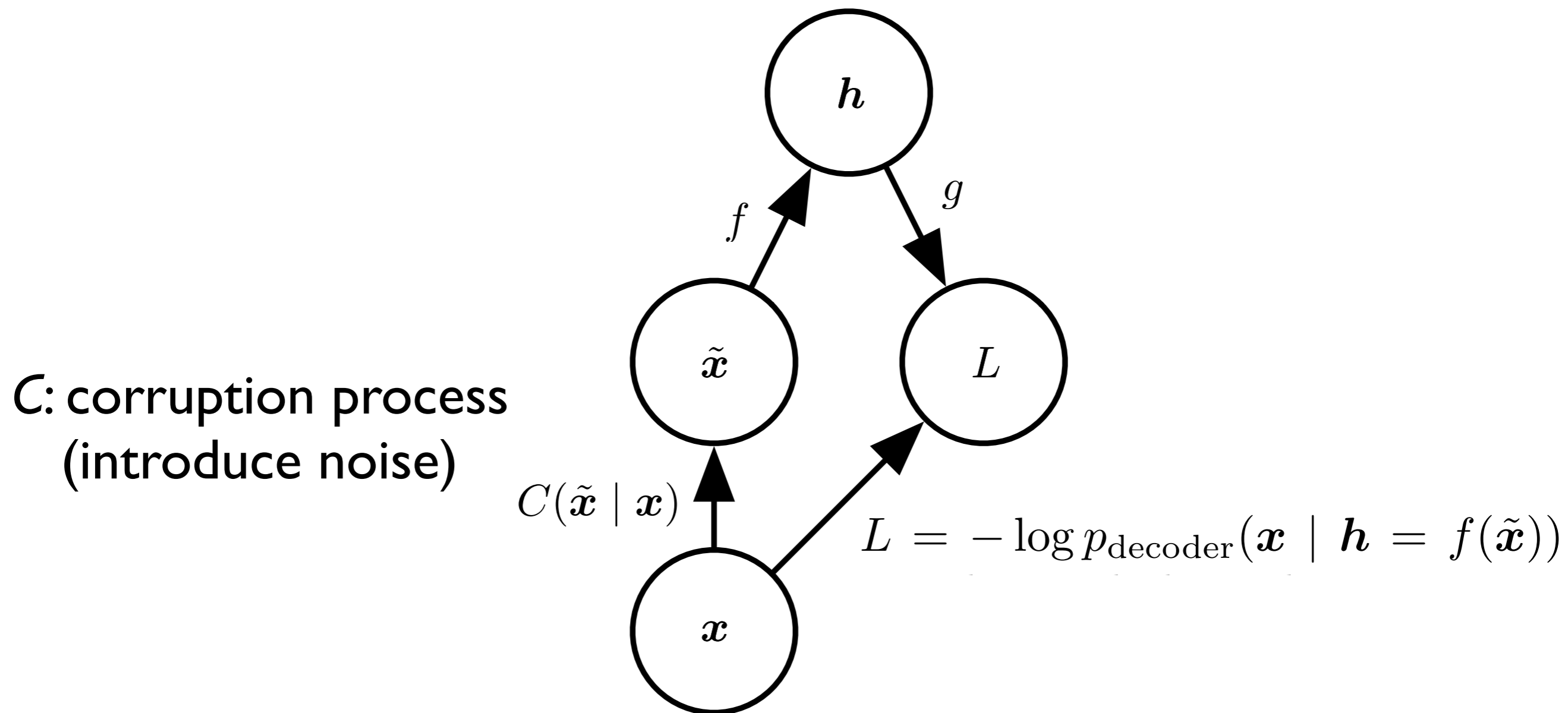
- **Overcomplete autoencoders**

- $h$  has higher dimension than  $x$
- Must be regularized

# Regularized autoencoders

- Sparse autoencoders
- **Denoising autoencoders**
- Autoencoders with dropout on the hidden layer
- Contractive autoencoders

# Denoising autoencoder



# Denoising autoencoder learns a manifold

