Deep Generative Models

Part1: Introduction

Mohamed Elhoseiny

KAUST

mohamed.elhoseiny@kaust.edu.sa

What to Expect ?

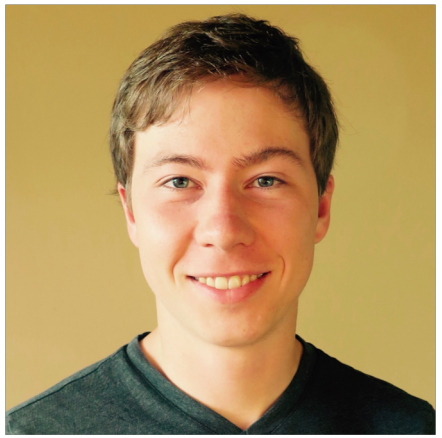
- **Part 1:** Learn what are generative models and about key modeling approaches including Generative Adversarial Networks and Variational Auto Encoders.

- **Part 2:** Learn more about Deep Generative Models capabilities to solve problems like recognizing unseen objects and creating unseen likable images.

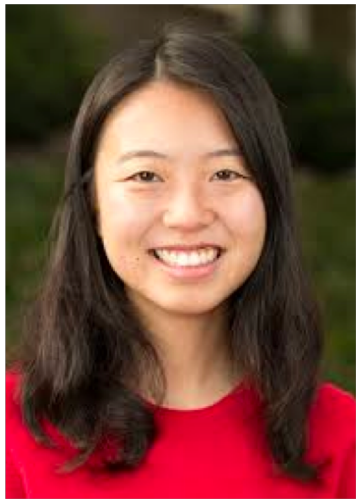
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Credit for Part 1

Fei-Fei Li



Justin Johnson



Serena Yeung

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Part1: Introduction to Deep Generative Models

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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.

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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.

Cat

Classification

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

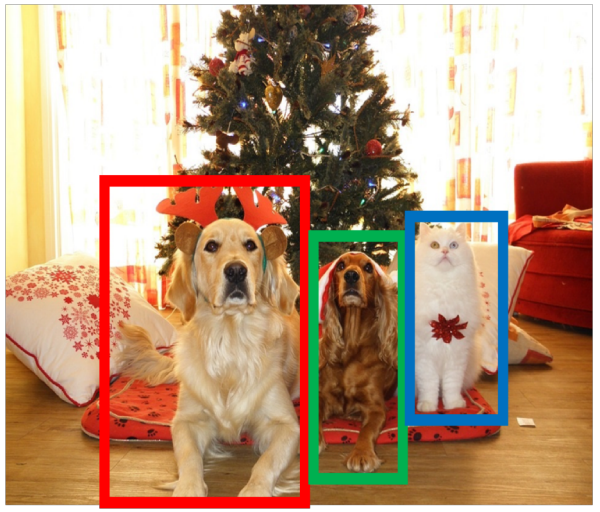
Supervised vs Unsupervised Learning

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.



DOG, **DOG**, **CAT**

Object Detection

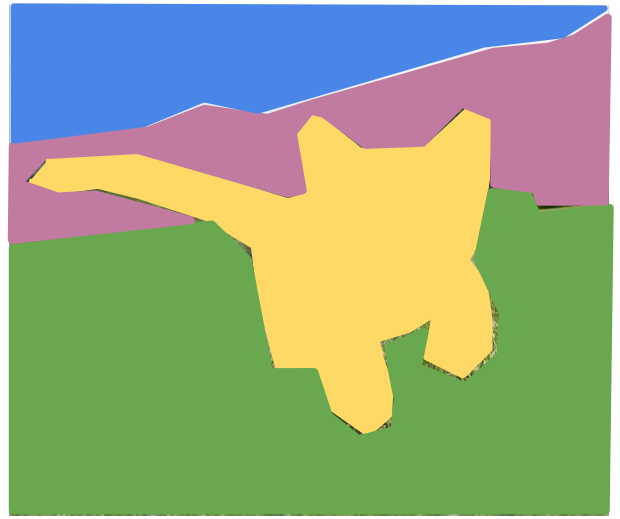
Supervised vs Unsupervised Learning

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.



GRASS, CAT, TREE, SKY

Semantic Segmentation

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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.

A cat sitting on a suitcase on the floor

Image captioning

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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, etc.

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

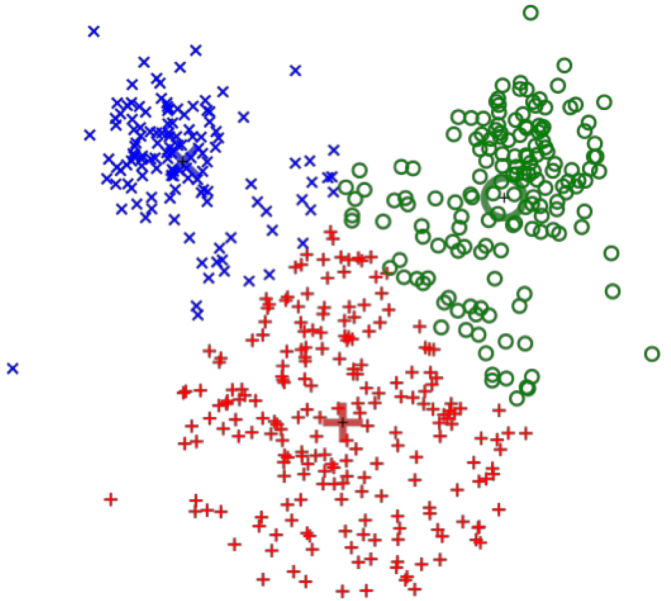
Supervised vs Unsupervised Learning

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, etc.



K-means clustering

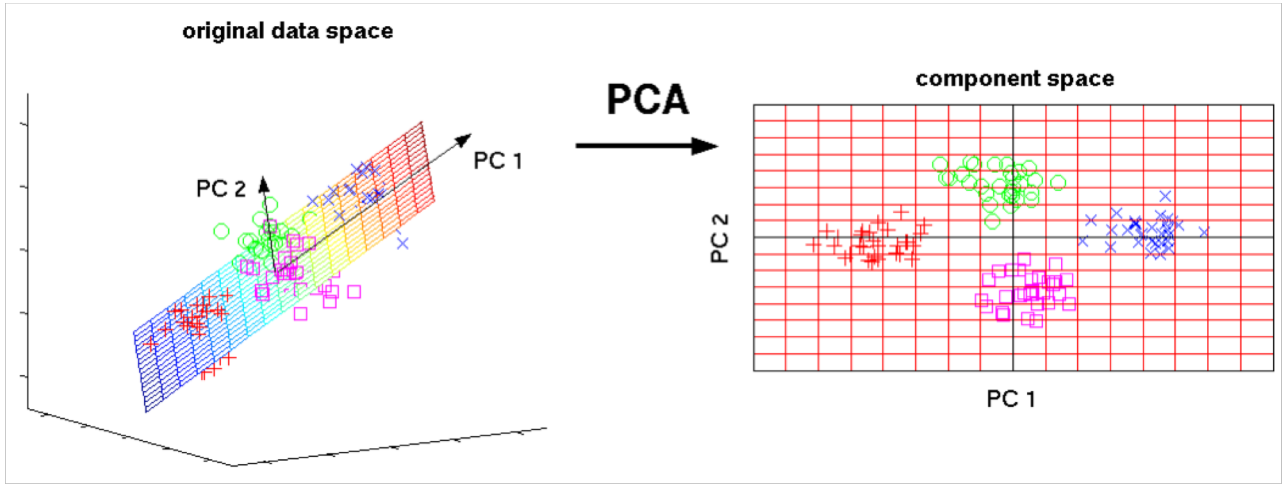
Supervised vs Unsupervised Learning

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, etc.



3-d → 2-d

Principal Component Analysis
(Dimensionality reduction)

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

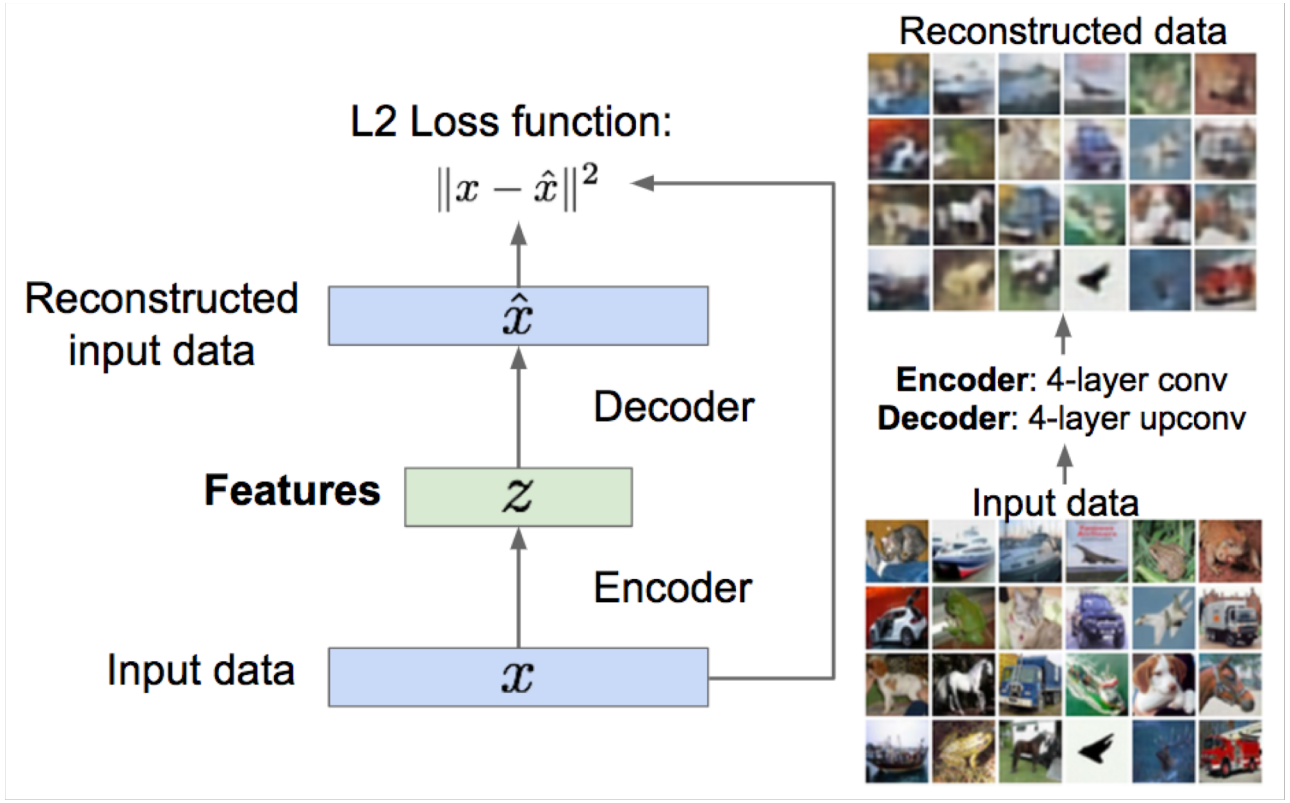
Supervised vs Unsupervised Learning

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, etc.



Autoencoders
(Feature learning)

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Supervised vs Unsupervised Learning

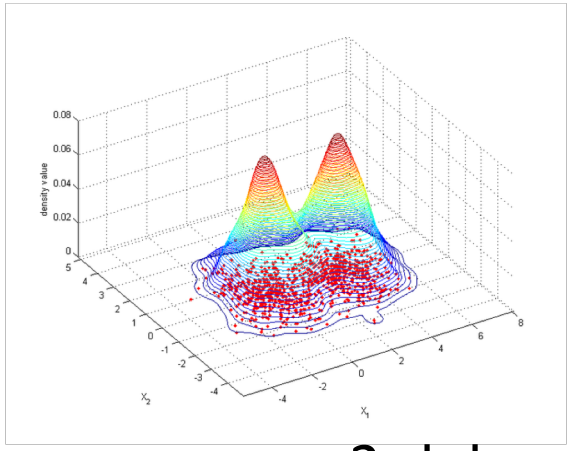
Unsupervised Learning

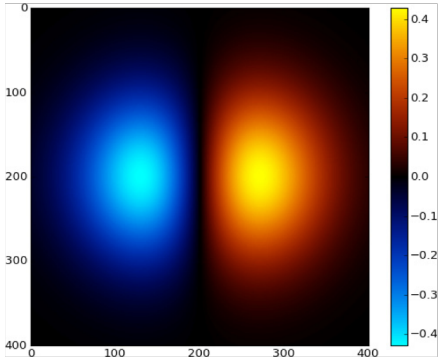
Data: x
Just data, no labels!

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

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

1-d density estimation





2-d density estimation

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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.

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.

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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.

Unsupervised Learning

Training data is cheap

Data: x
Just data, no labels!

Holy grail: Solve unsupervised learning => understand structure of visua world

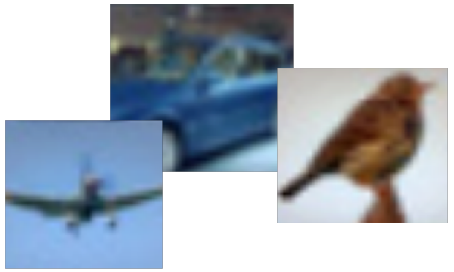
Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

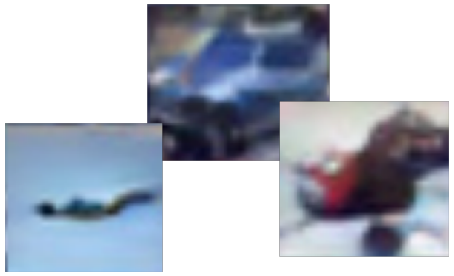
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Generative Models

Given training data, generate new samples from same distribution



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



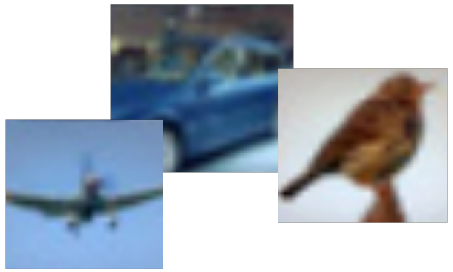
Generated samples ~ $p_{model}(x)$

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

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Generative Models

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 $p_{data}(x)$

Addresses density estimation, a core problem in unsupervised learning

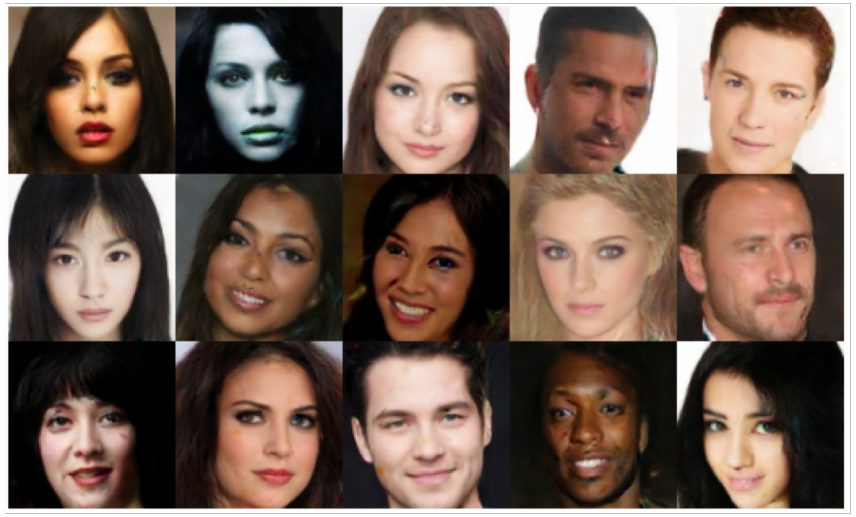
Several flavors:

- Explicit density estimation: explicitly define and solve for $p_{model}(x)$
- Implicit density estimation: learn model that can sample from $p_{model}(x)$ w/o explicitly defining it

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

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

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Taxonomy of Generative Models

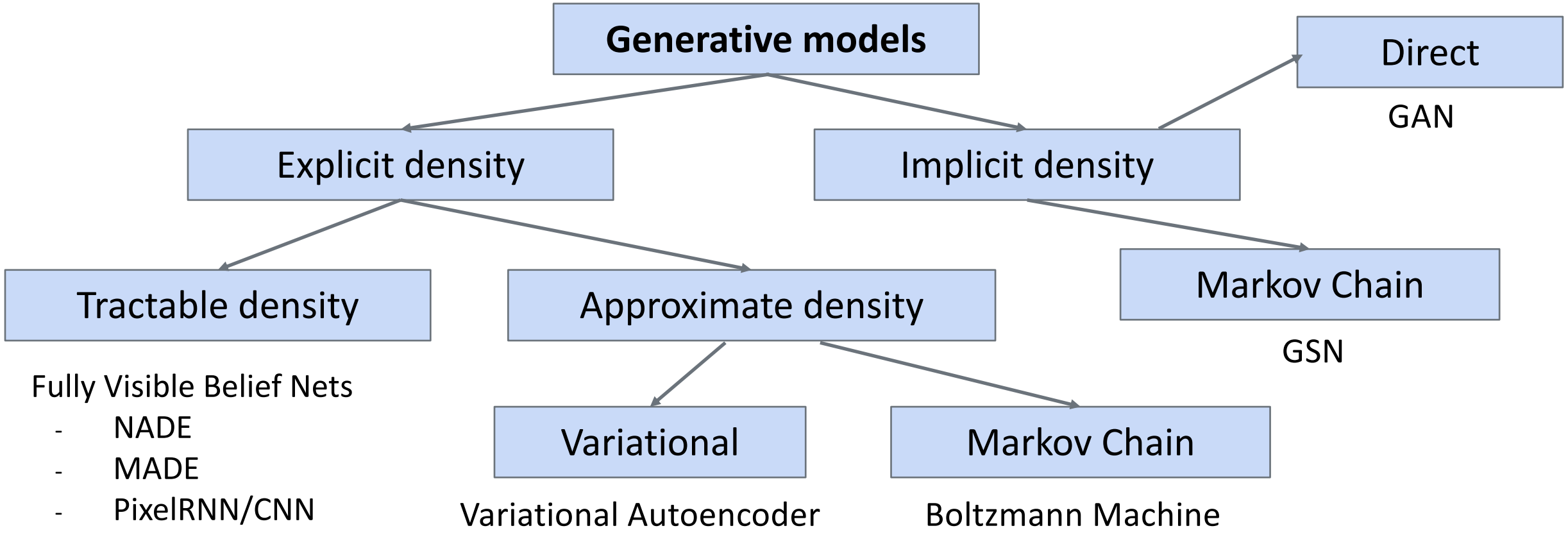
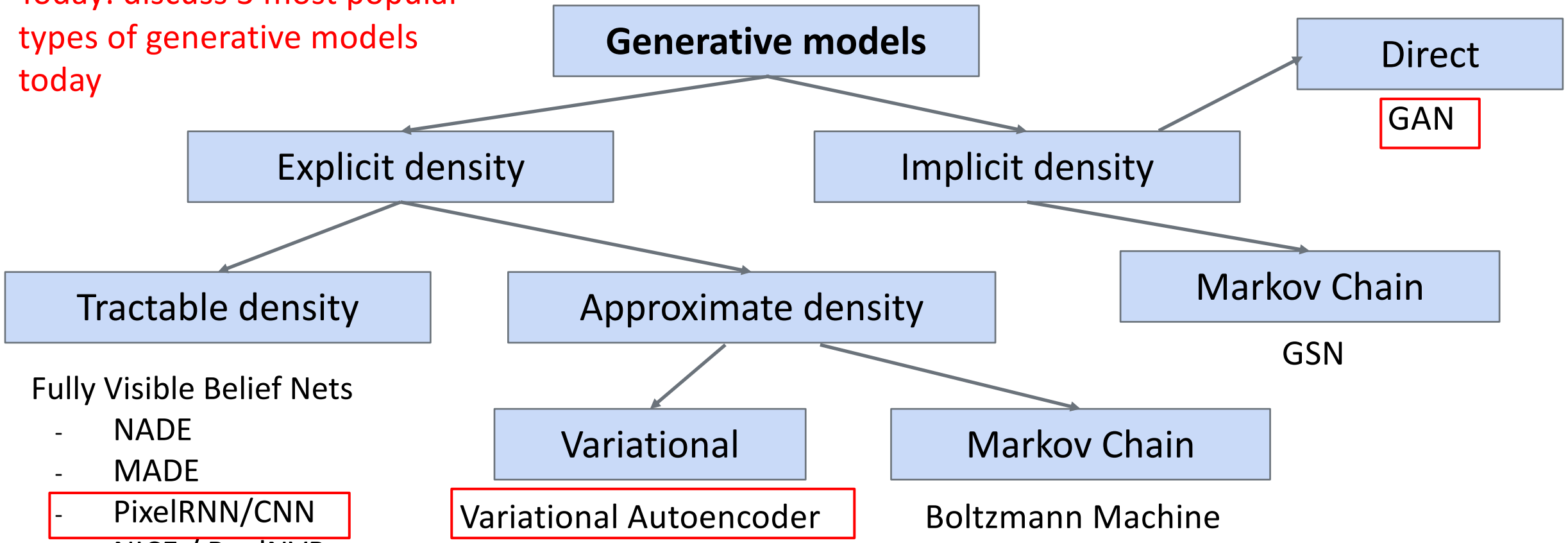


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Taxonomy of Generative Models

Today: discuss 3 most popular types of generative models today



Fully Visible Belief Nets
- NADE
- MADE
- PixelRNN/CNN
- NICE / RealNVP
- Glow
- Ffjord

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

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

PixelRNN and PixelCNN

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Fully visible belief network

Explicit density model

Use chain rule to decompose likelihood of an image x into product of 1-d distributions:

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

Likelihood of image x

Probability of i'th pixel value given all previous pixels

Then maximize likelihood of training data

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Fully visible belief network

Explicit density model

Use chain rule to decompose likelihood of an image x into product of 1-d distributions:

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

Likelihood of image x

Probability of i'th pixel value given all previous pixels

Complex distribution over pixel values
=> Express using a neural network!

Then maximize likelihood of training data

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Fully visible belief network

Explicit density model

Use chain rule to decompose likelihood of an image x into product of 1-d distributions:

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

Likelihood of image x

Probability of i'th pixel value given all previous pixels

Will need to define ordering of "previous pixels"

Then maximize likelihood of training data

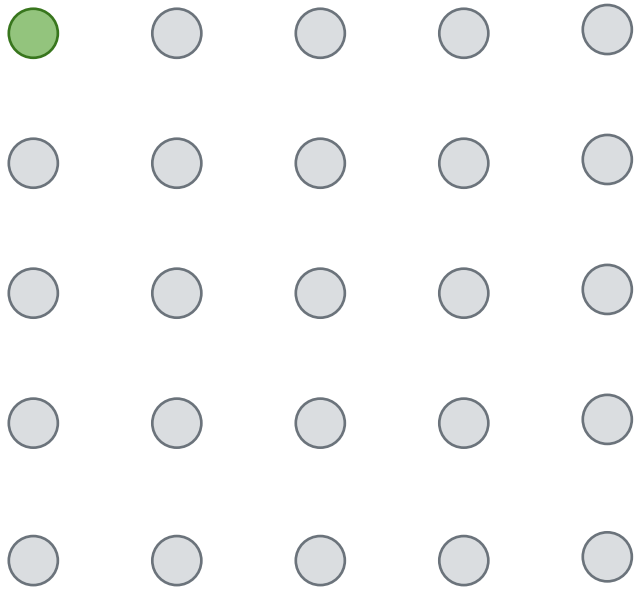
Complex distribution over pixel values
=> Express using a neural network!

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

PixelRNN *[van der Oord et al. 2016]*

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)

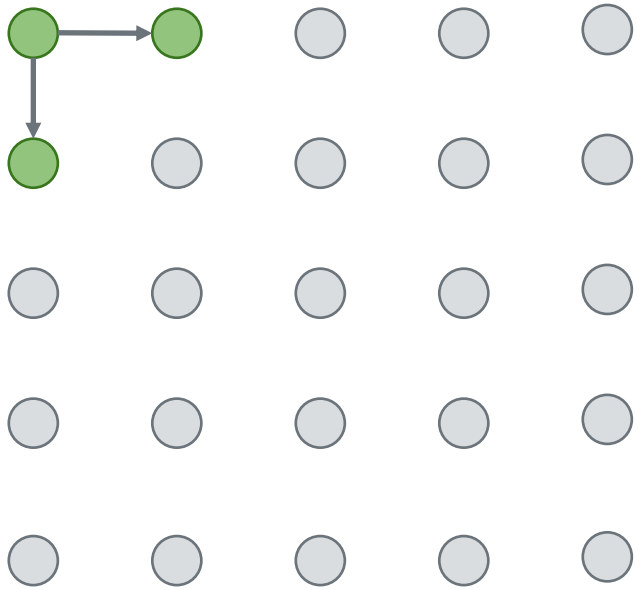


Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

PixelRNN *[van der Oord et al. 2016]*

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)

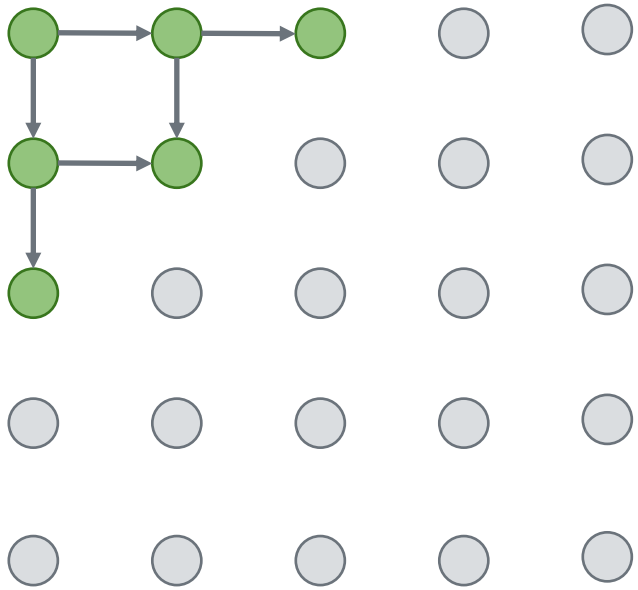


Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

PixelRNN *[van der Oord et al. 2016]*

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)

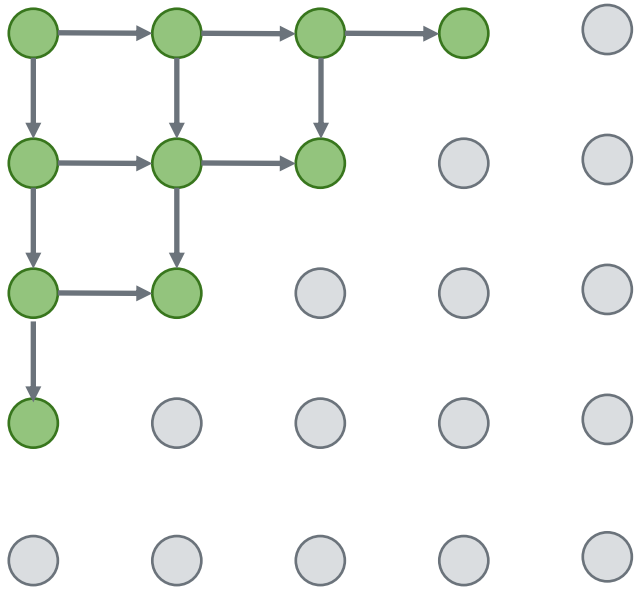


Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

PixelRNN *[van der Oord et al. 2016]*

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)



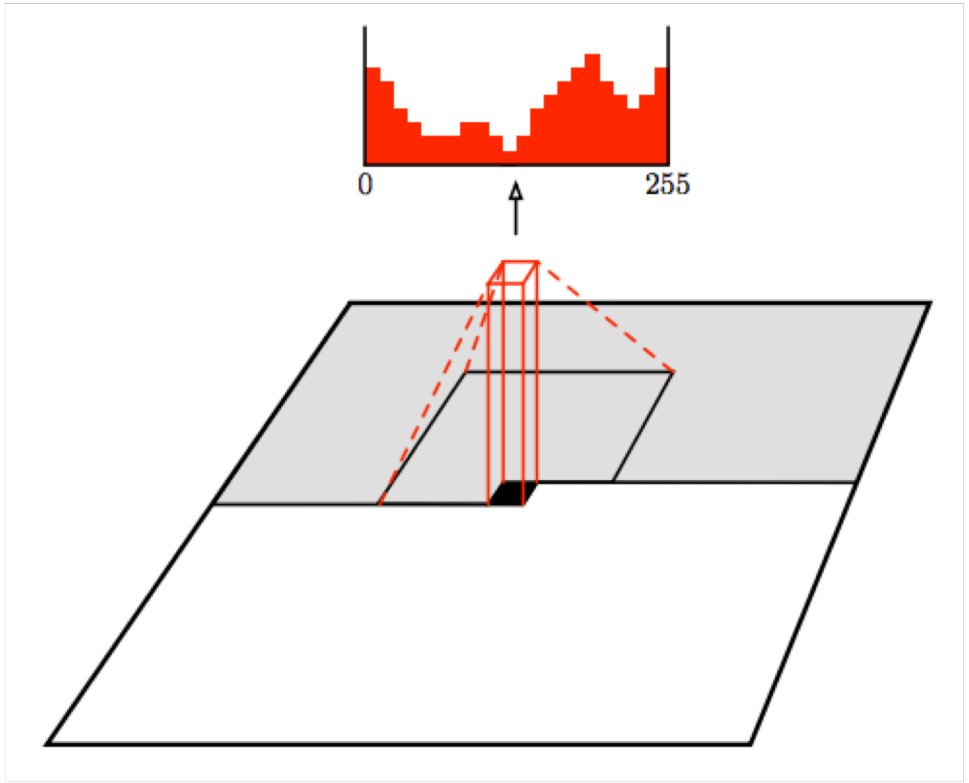
Drawback: sequential generation is slow!

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

PixelCNN *[van der Oord et al. 2016]*

Still generate image pixels starting from corner

Dependency on previous pixels now modeled using a CNN over context region



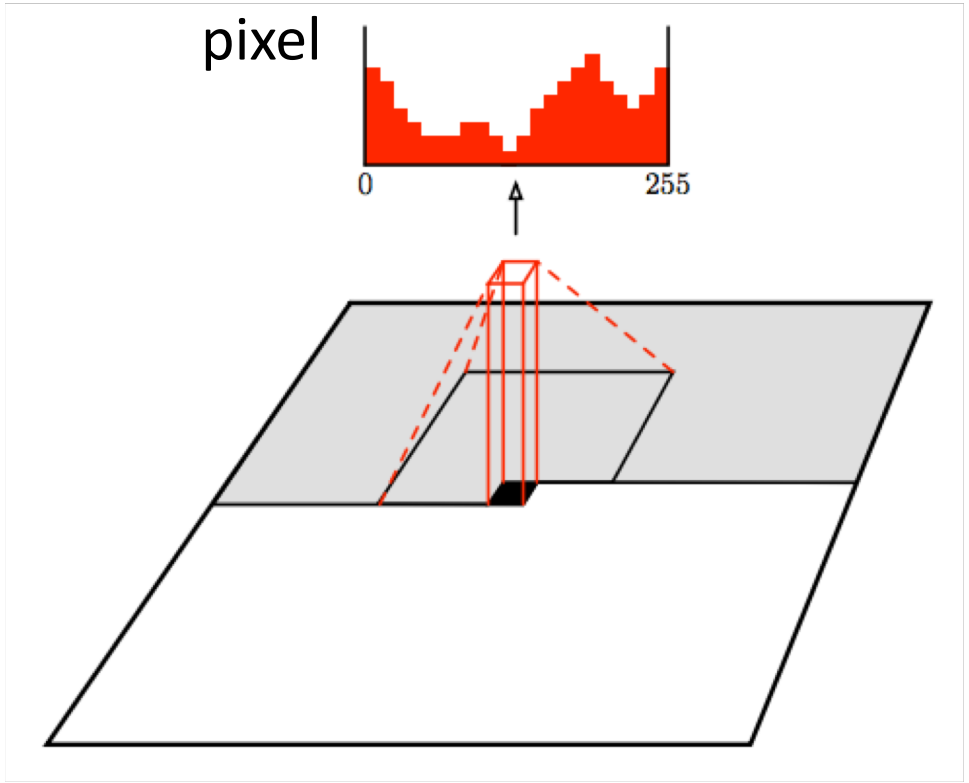
PixelCNN *[van der Oord et al. 2016]*

Still generate image pixels starting from corner

Dependency on previous pixels now modeled using a CNN over context region

Training: maximize likelihood of training images

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$



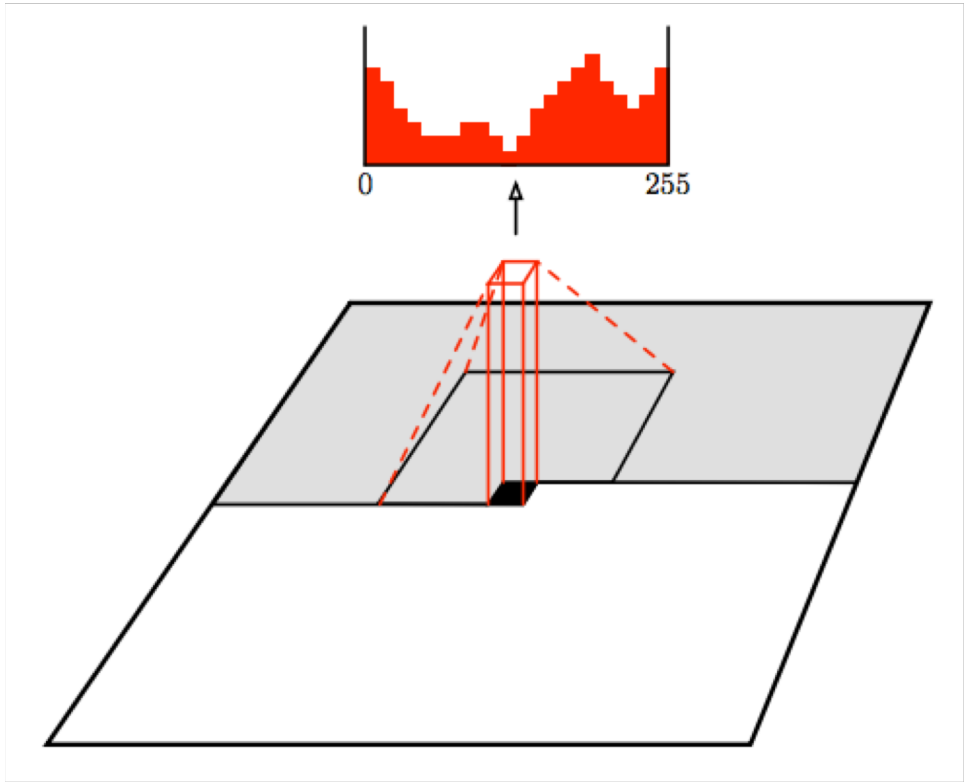
PixelCNN *[van der Oord et al. 2016]*

Still generate image pixels starting from corner

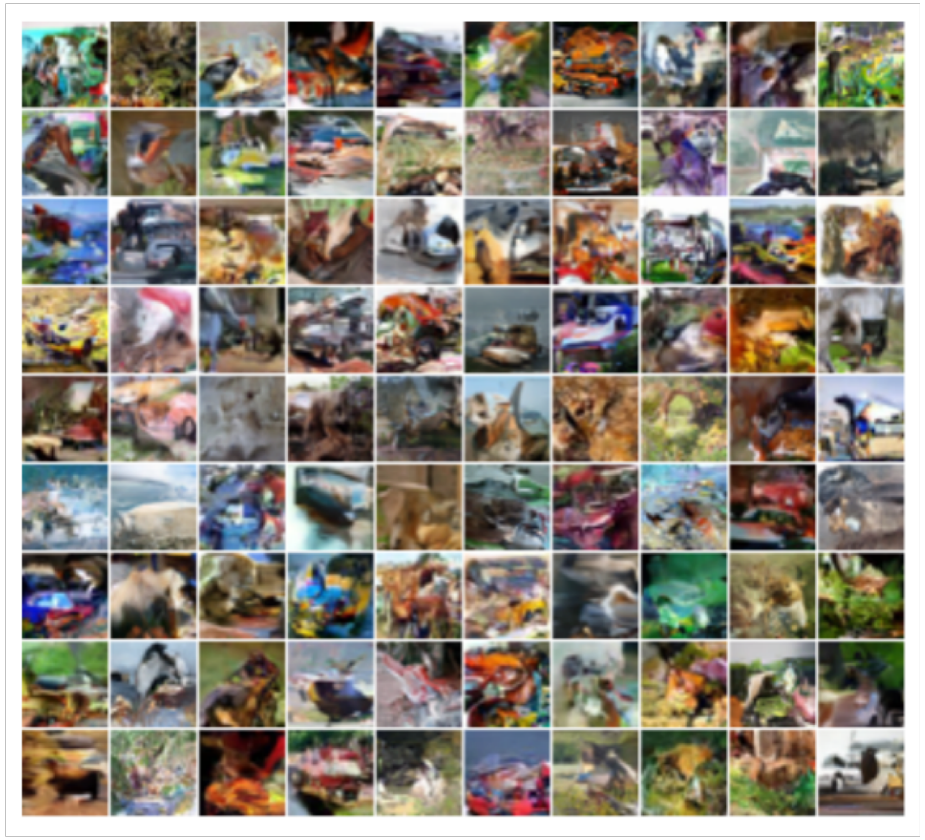
Dependency on previous pixels now modeled using a CNN over context region

Training is faster than PixelRNN
(can parallelize convolutions since context region values known from training images)

Generation must still proceed sequentially
=> still slow



Generation Samples



32x32 CIFAR-10

32x32 ImageNet

PixelRNN and PixelCNN

Pros:

- Can explicitly compute likelihood p(x)
- Explicit likelihood of training data gives good evaluation metric
- Good samples

Con:

- Sequential generation => slow

Improving PixelCNN performance

- Gated convolutional layers
- Short-cut connections
- Discretized logistic loss
- Multi-scale
- Training tricks
- Etc…

See

- Van der Oord et al. NIPS 2016
- Salimans et al. 2017 (PixelCNN++)

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Variational Autoencoders (VAE)

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Variational Autoencoders

So far...

PixelCNNs define tractable density function, optimize likelihood of training data:

$$p_\theta(x) = \prod_{i=1}^{n} p_\theta(x_i | x_1, ..., x_{i-1})$$

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Variational Autoencoders

So far...

PixelCNNs define tractable density function, optimize likelihood of training data:

$$p_\theta(x) = \prod_{i=1}^{n} p_\theta(x_i | x_1, ..., x_{i-1})$$

VAEs define intractable density function with latent **z**:

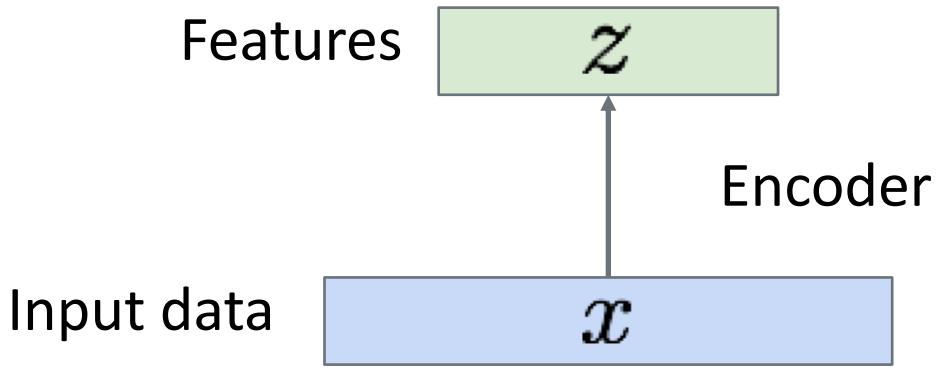
$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

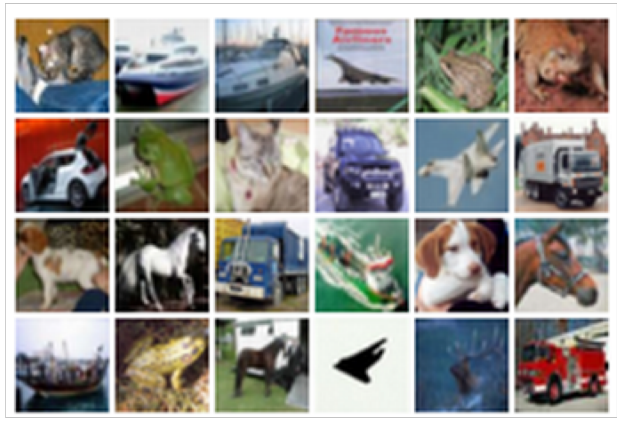
Cannot optimize directly, derive and optimize lower bound on likelihood instead

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

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



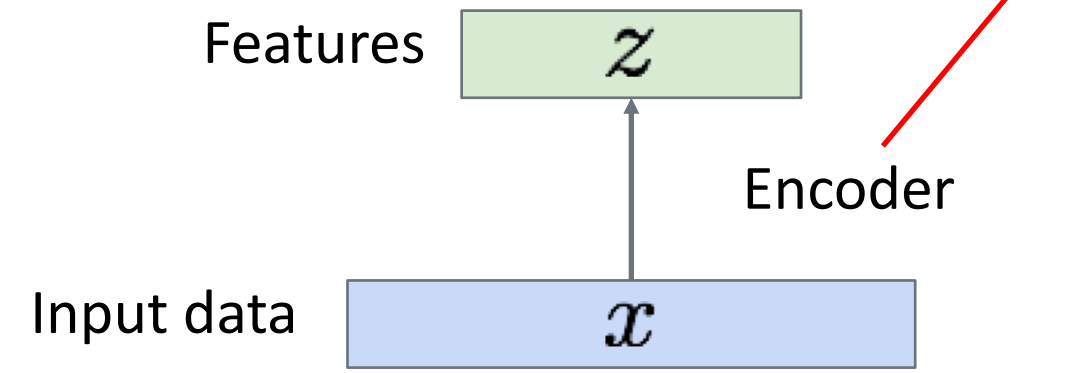


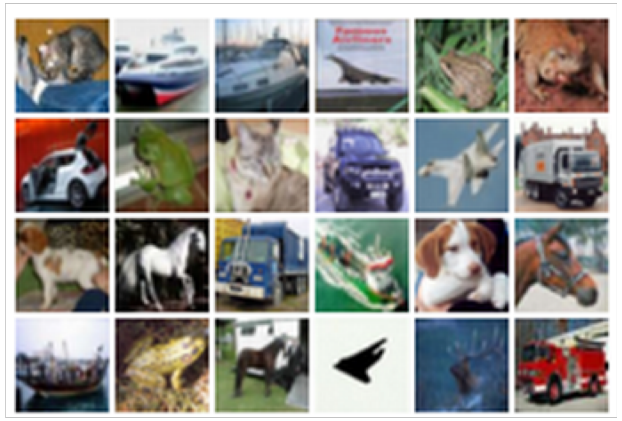
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

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

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





Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

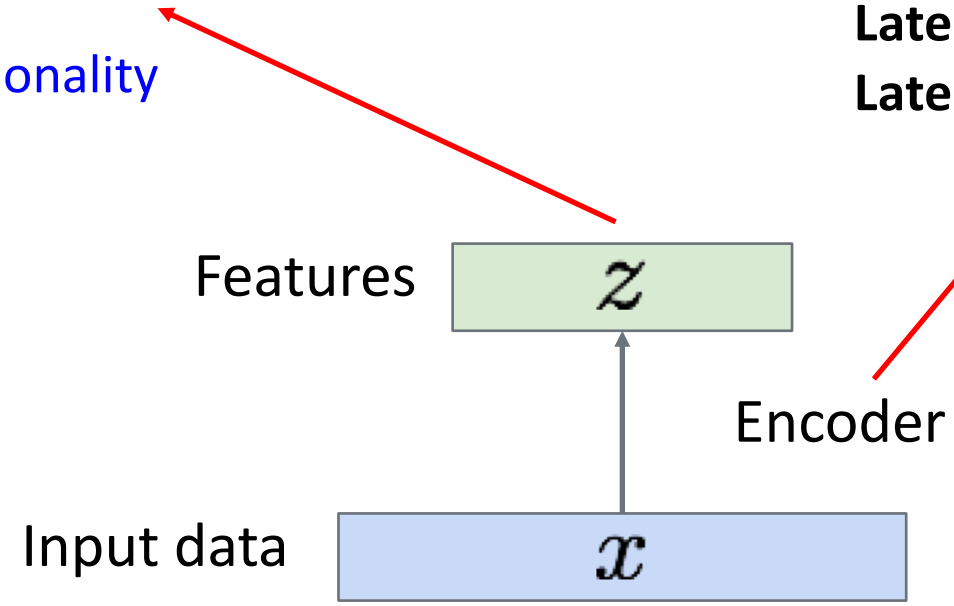
Some background first: Autoencoders

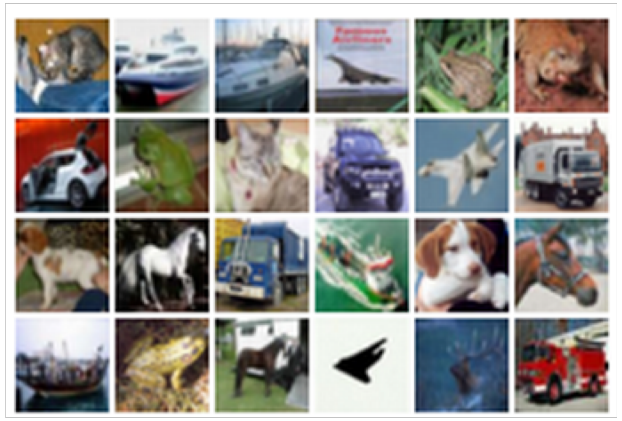
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

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

Q: Why dimensionality reduction?

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





Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

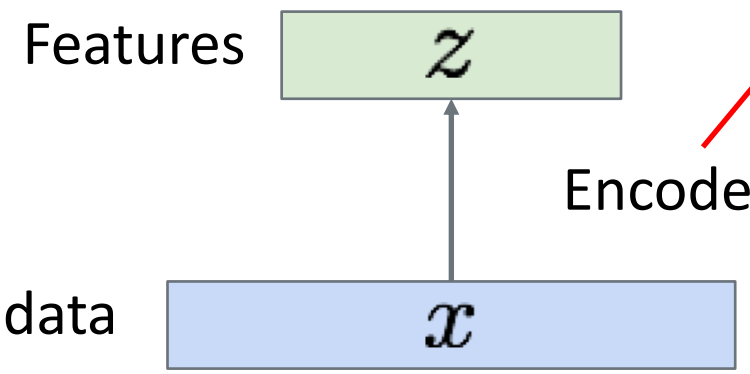
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

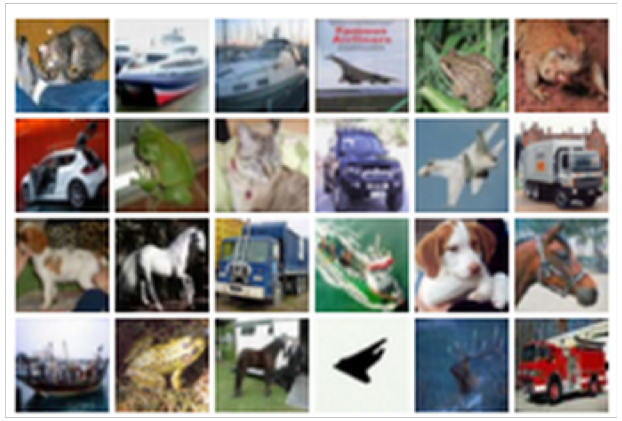
z usually smaller than **x**
(dimensionality reduction)

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

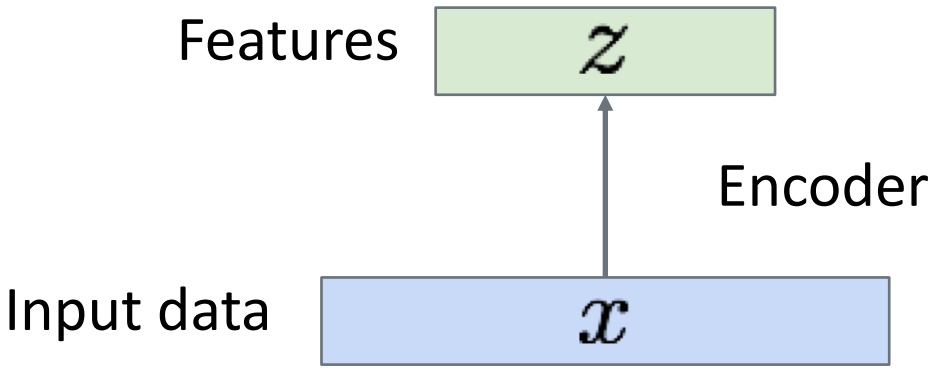


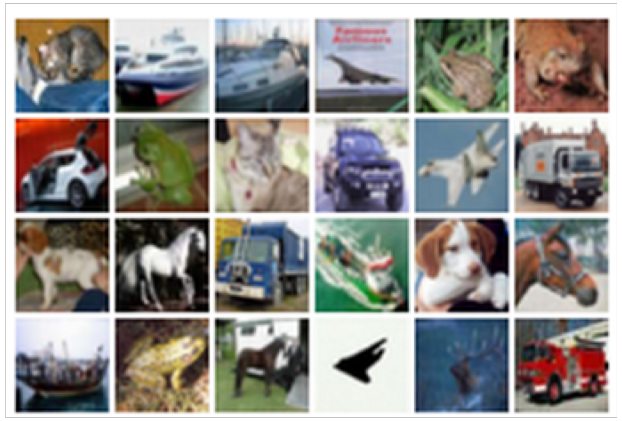


Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

How to learn this feature representation?





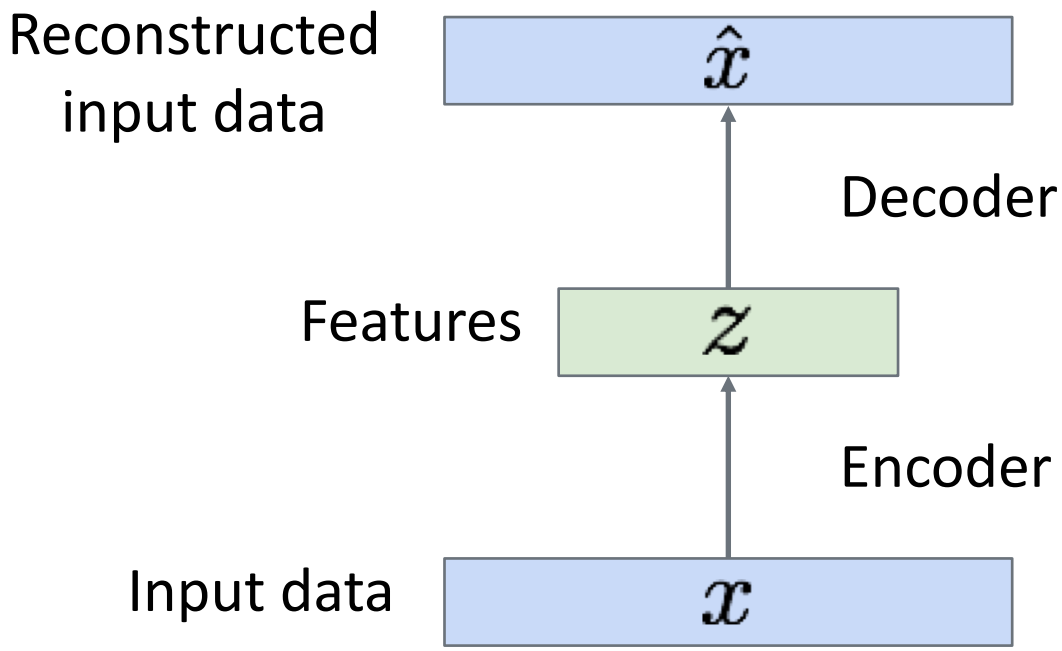
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

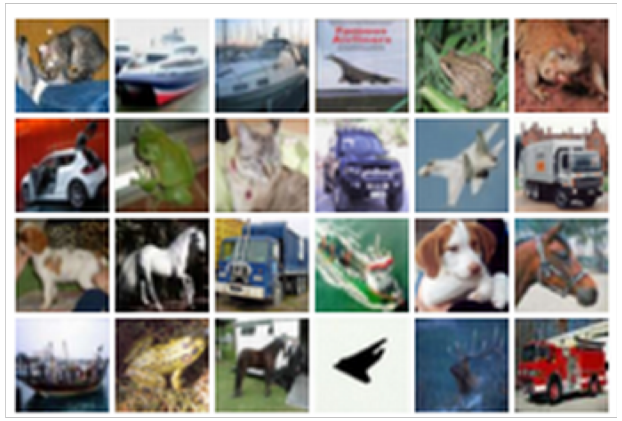
Some background first: Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data

“Autoencoding” - encoding itself





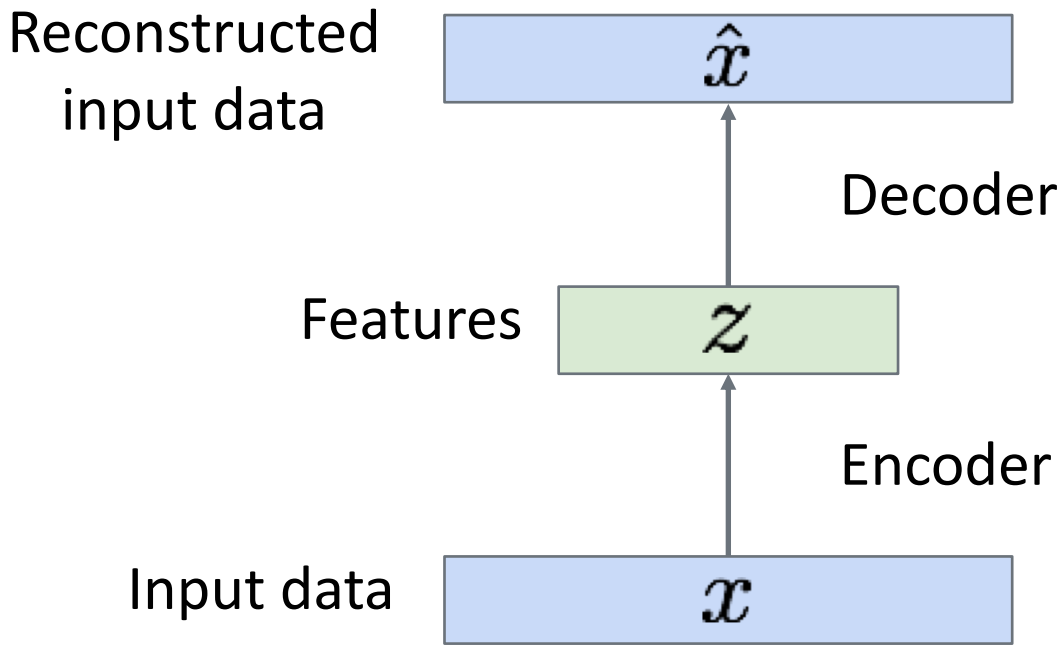
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data

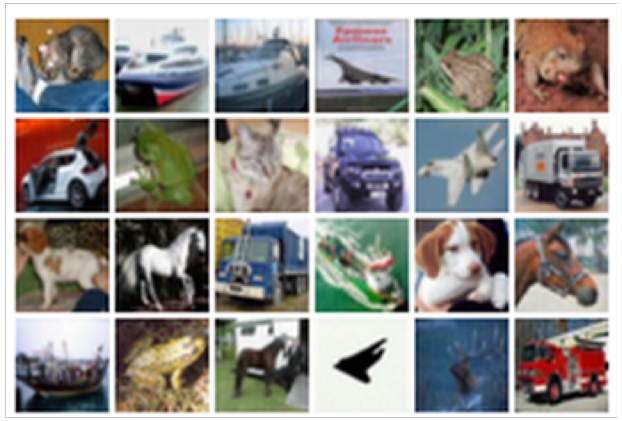
"Autoencoding" - encoding itself



Originally: Linear + nonlinearity (sigmoid)

Later: Deep, fully-connected

Later: ReLU CNN (upconv)



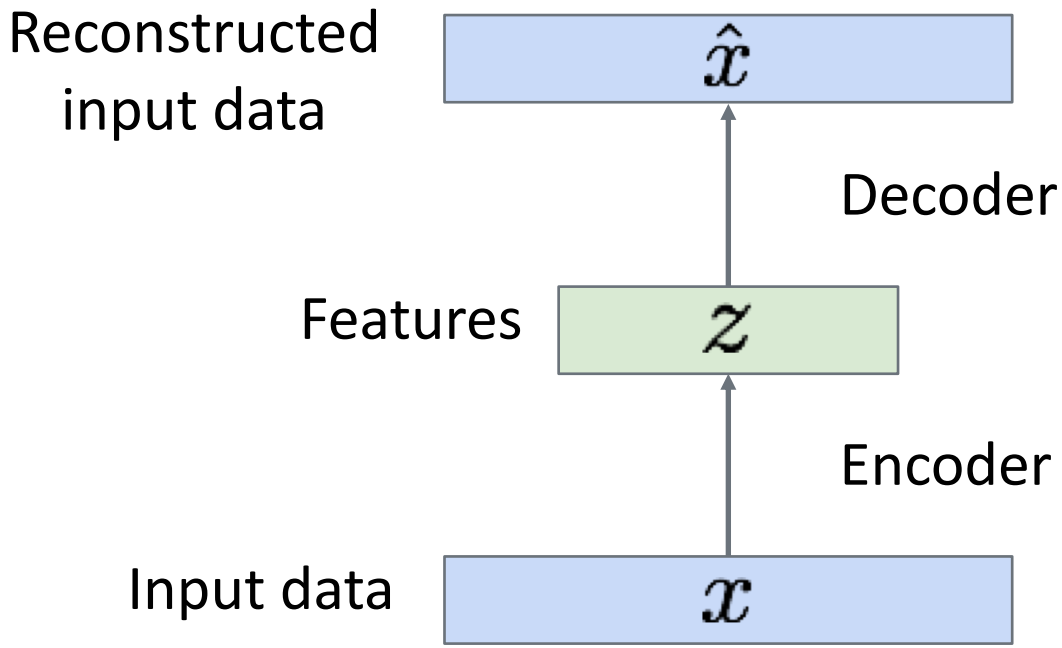
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

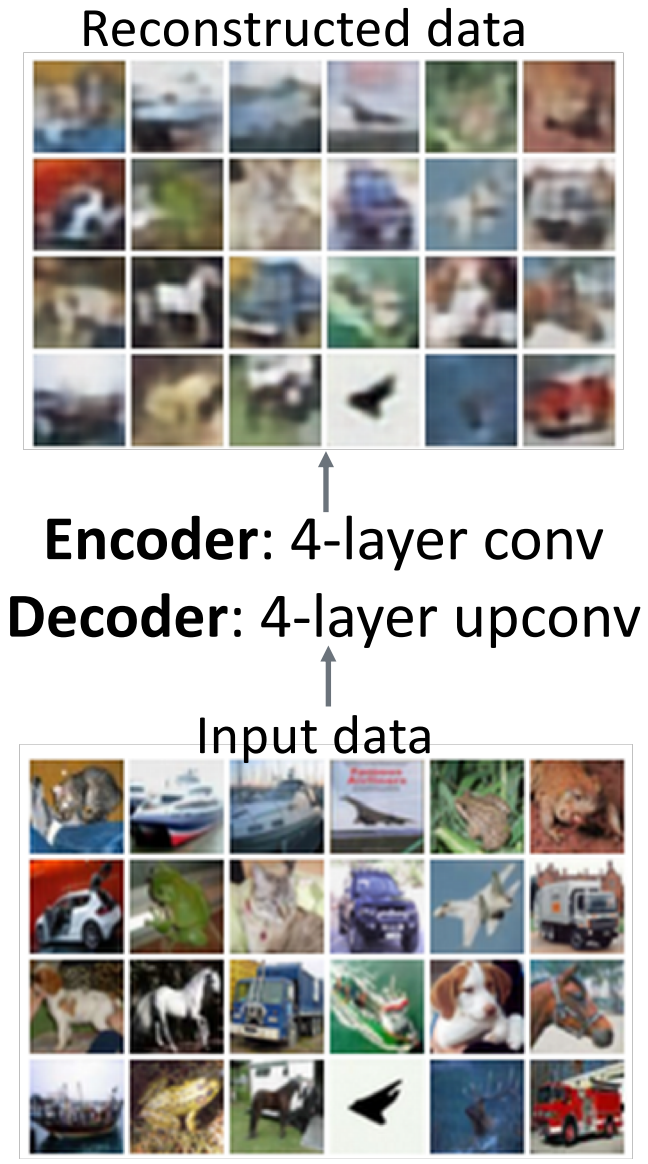
Some background first: Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data

"Autoencoding" - encoding itself

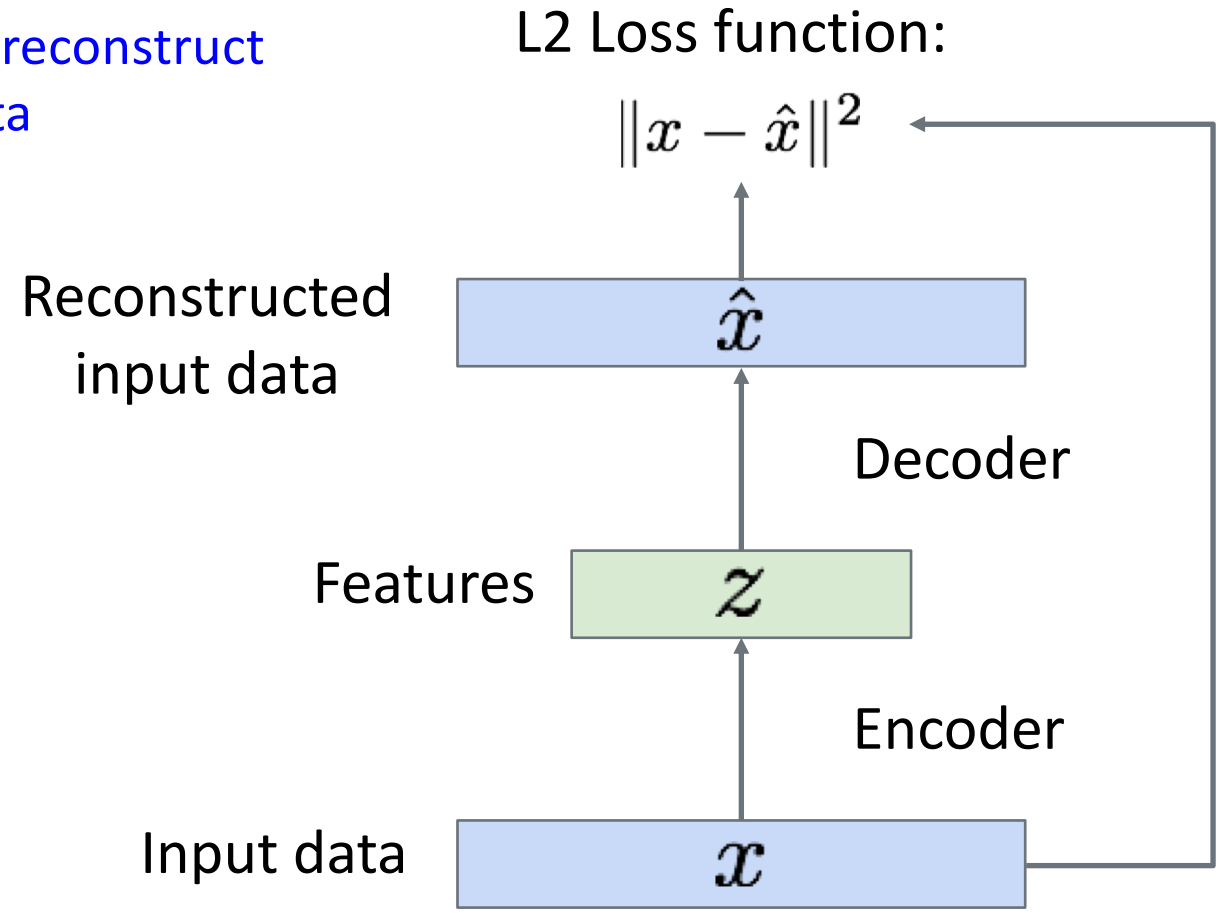




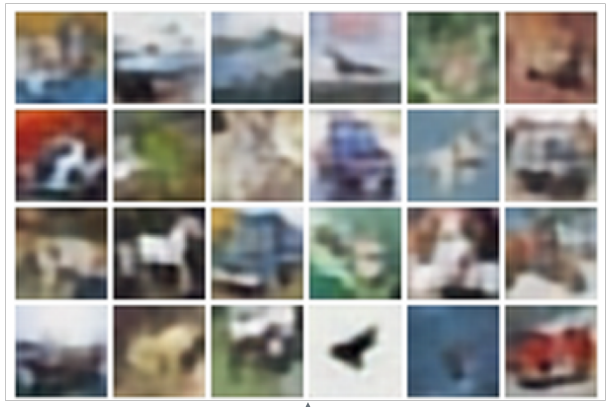
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

Train such that features can be used to reconstruct original data



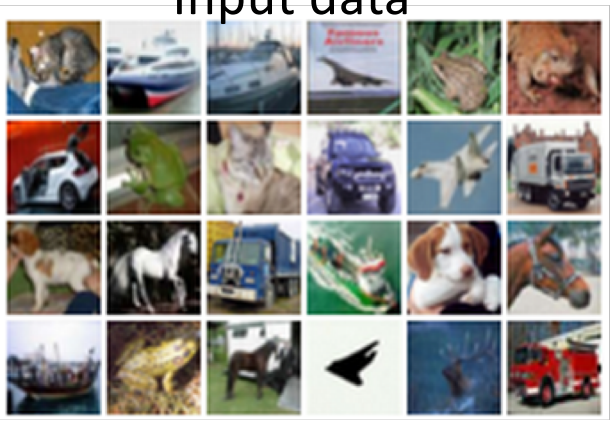
Reconstructed data



Encoder: 4-layer conv

Decoder: 4-layer upconv

Input data



Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

Train such that features can be used to reconstruct original data

Doesn't use labels!

L2 Loss function:

$$\|x - \hat{x}\|^2$$

Reconstructed input data

$\hat{x}$

Decoder

Features

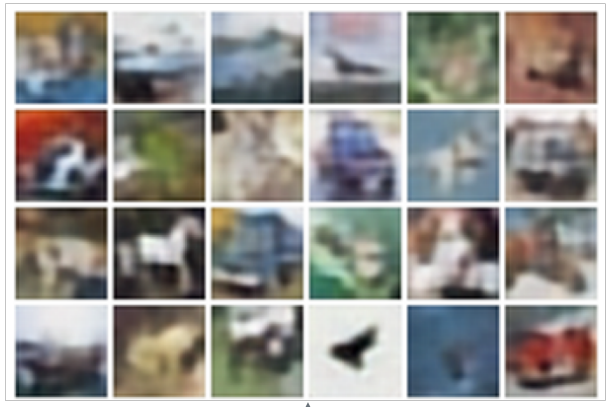
z

Encoder

Input data

x

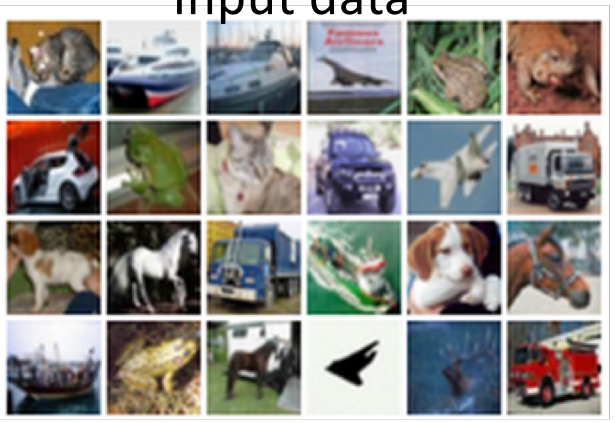
Reconstructed data



Encoder: 4-layer conv

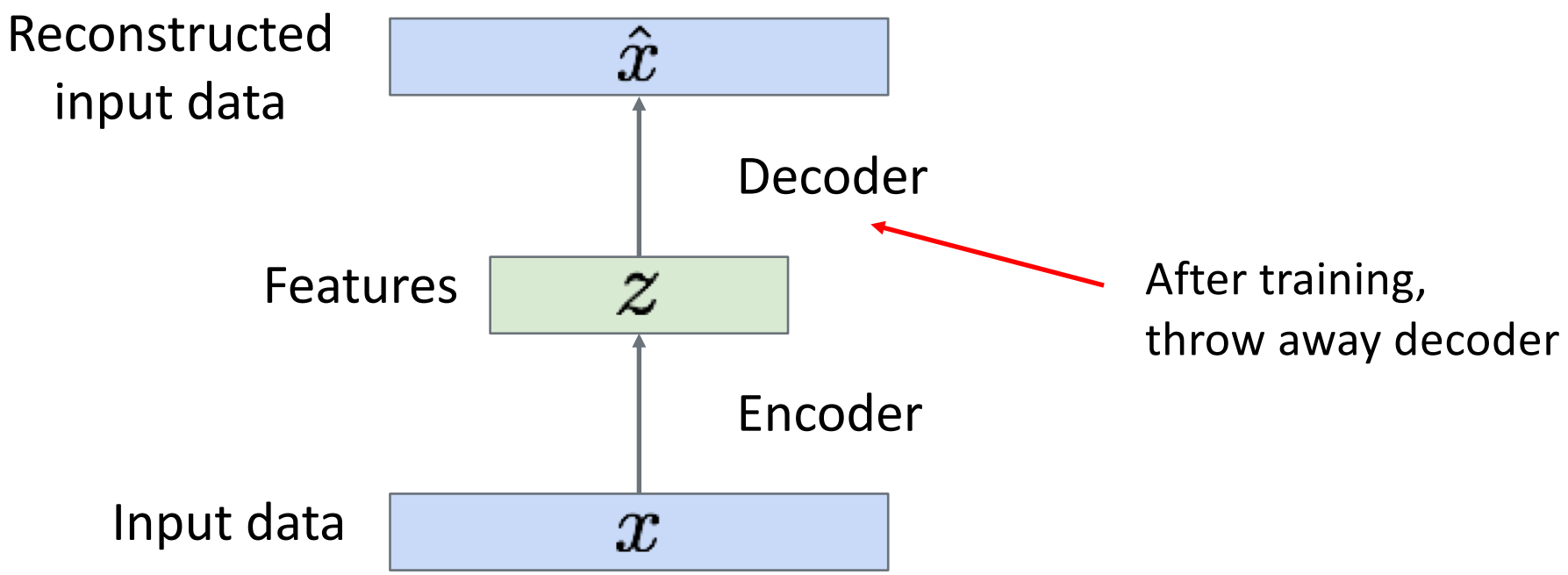
Decoder: 4-layer upconv

Input data



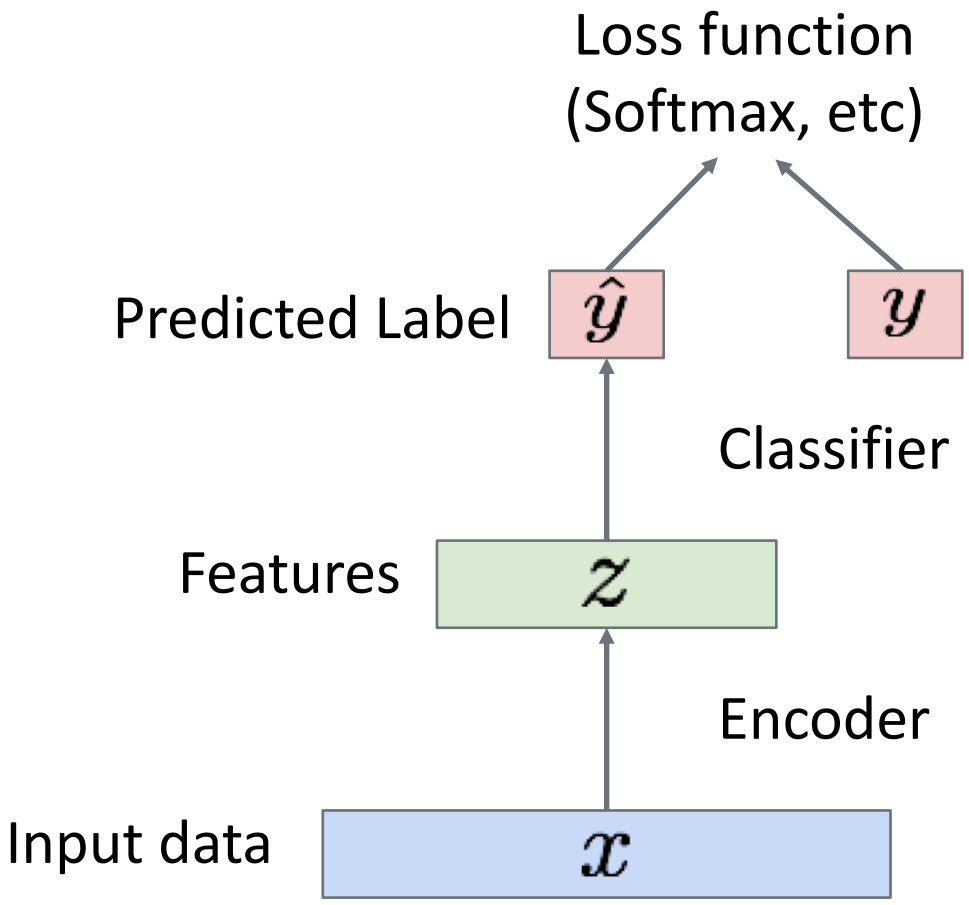
Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders



Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: Autoencoders

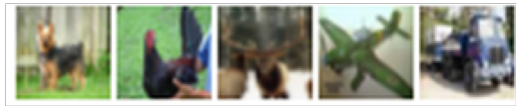


Encoder can be used to initialize a **supervised** model

Fine-tune encoder jointly with classifier

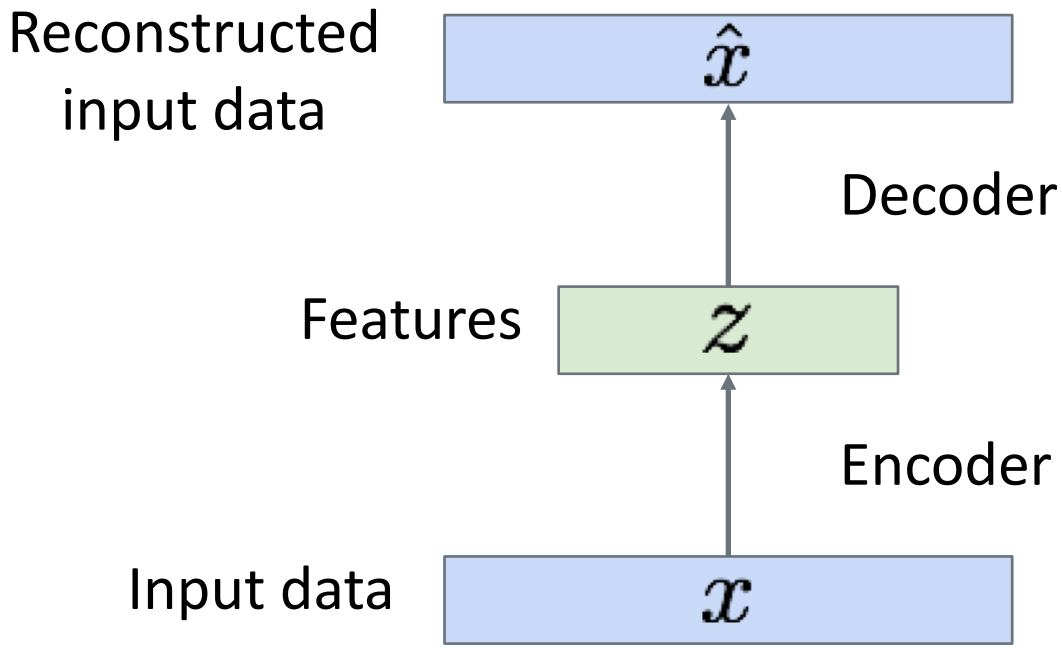
bird plan
dog deer truck

Train for final task (sometimes with small data)



Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung

Some background first: 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?

Slide credit: Fei-Fei Li, Justin Johnson, Serena Yeung