CS 4803 / 7643: Deep Learning

Topics:

- Unsupervised Learning
- Generative Models

Michael Cogswell

(subbing for Dhruv Batra)

Georgia Tech

Overview

- Unsupervised Learning
  - Comparison to Supervised and Reinforcement Learning
  - Review of K-Means

- e.g., Generative Models
  - Varieties
  - PixelRNN and PixelCNN

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

Supervised Learning

Given: (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, CS 231n

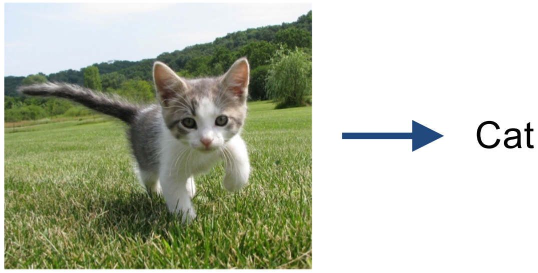
Supervised vs Reinforcement vs Unsupervised Learning

Supervised Learning

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

Goal: Learn a *function* to map x → y

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



Classification

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

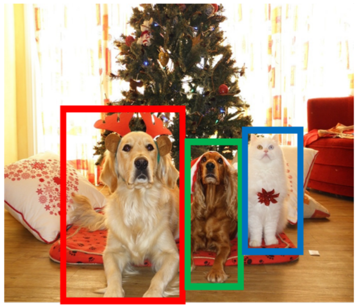
Supervised vs Reinforcement vs Unsupervised Learning

Supervised Learning

Given: (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

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

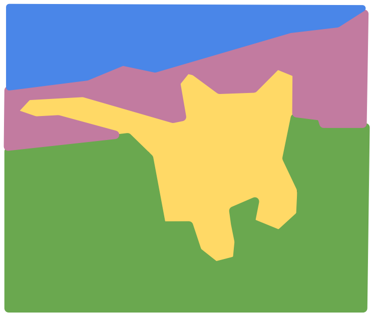
Supervised Learning

Given: (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

Supervised vs Reinforcement vs Unsupervised Learning

Supervised Learning

Given: (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, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

Reinforcement Learning

Given: (e, r)

Environment e, Reward function r (evaluative feedback)

Goal: Maximize expected reward

Examples: Robotic control, video games, board games, etc.

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

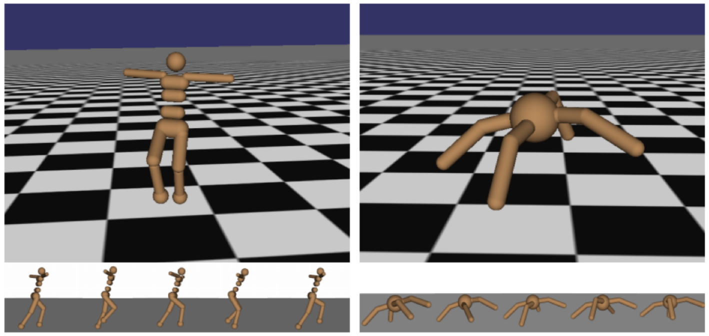
Reinforcement Learning

Given: (e, r)

Environment e, Reward function r (evaluative feedback)

Goal: Maximize expected reward

Examples: Robotic control, video games, board games, etc.



Robotic Locomotion

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

Reinforcement Learning

Given: (e, r)

Environment e, Reward function r (evaluative feedback)

Goal: Maximize expected reward

Examples: Robotic control, video games, board games, etc.

Atari Games

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

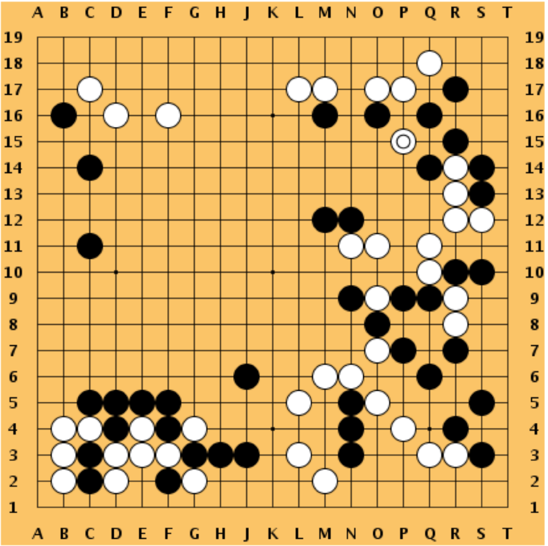
Reinforcement Learning

Given: (e, r)

Environment e, Reward function r (evaluative feedback)

Goal: Maximize expected reward

Examples: Robotic control, video games, board games, etc.



Go

Supervised vs Reinforcement vs Unsupervised Learning

Unsupervised Learning

Given: 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, CS 231n

Supervised vs Reinforcement vs Unsupervised Learning

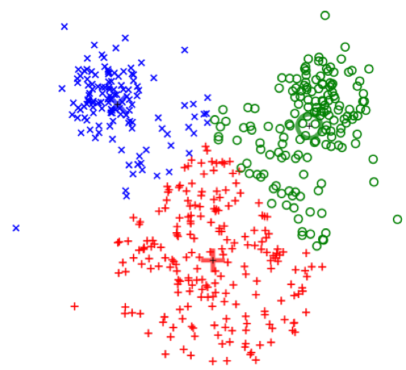
Unsupervised Learning

Given: 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

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

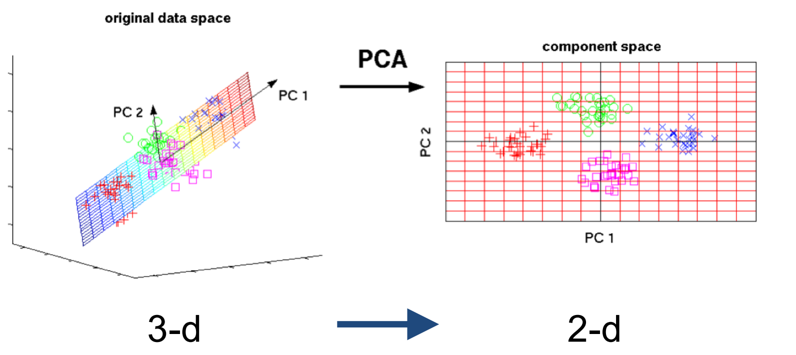
Supervised vs Reinforcement vs Unsupervised Learning

Unsupervised Learning

Given: 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)

Supervised vs Reinforcement vs Unsupervised Learning

Unsupervised Learning

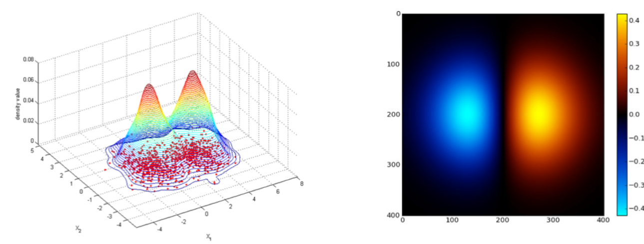
Given: Data x

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

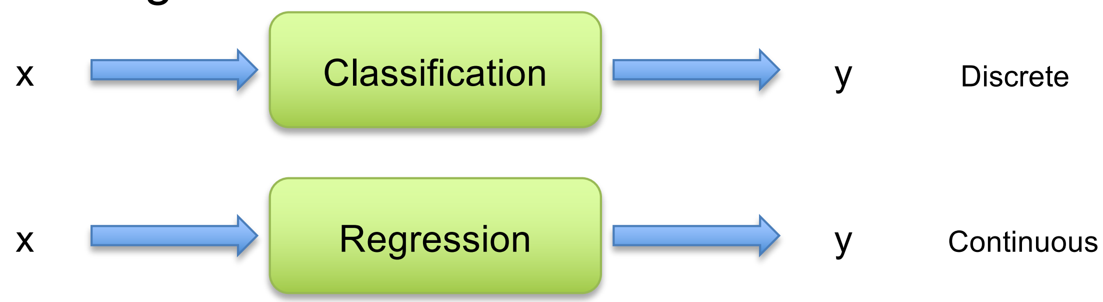


2-d density estimation

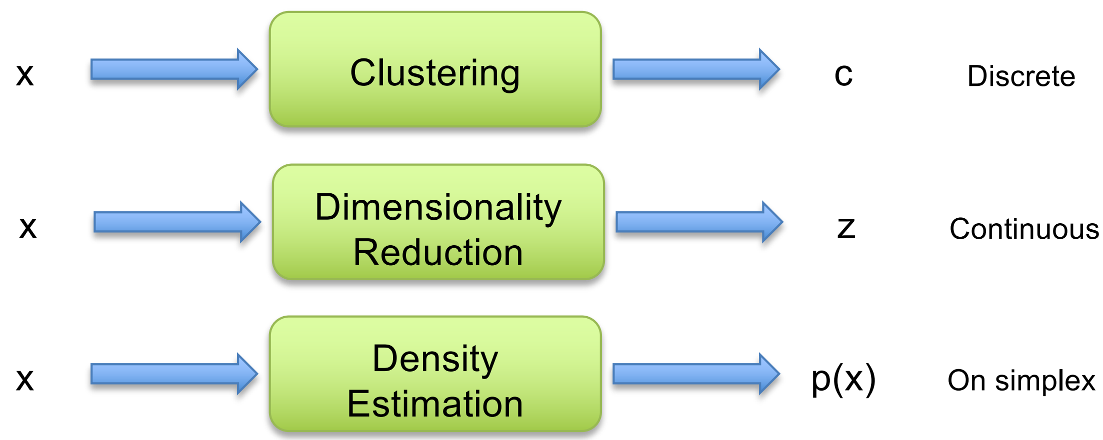
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Tasks

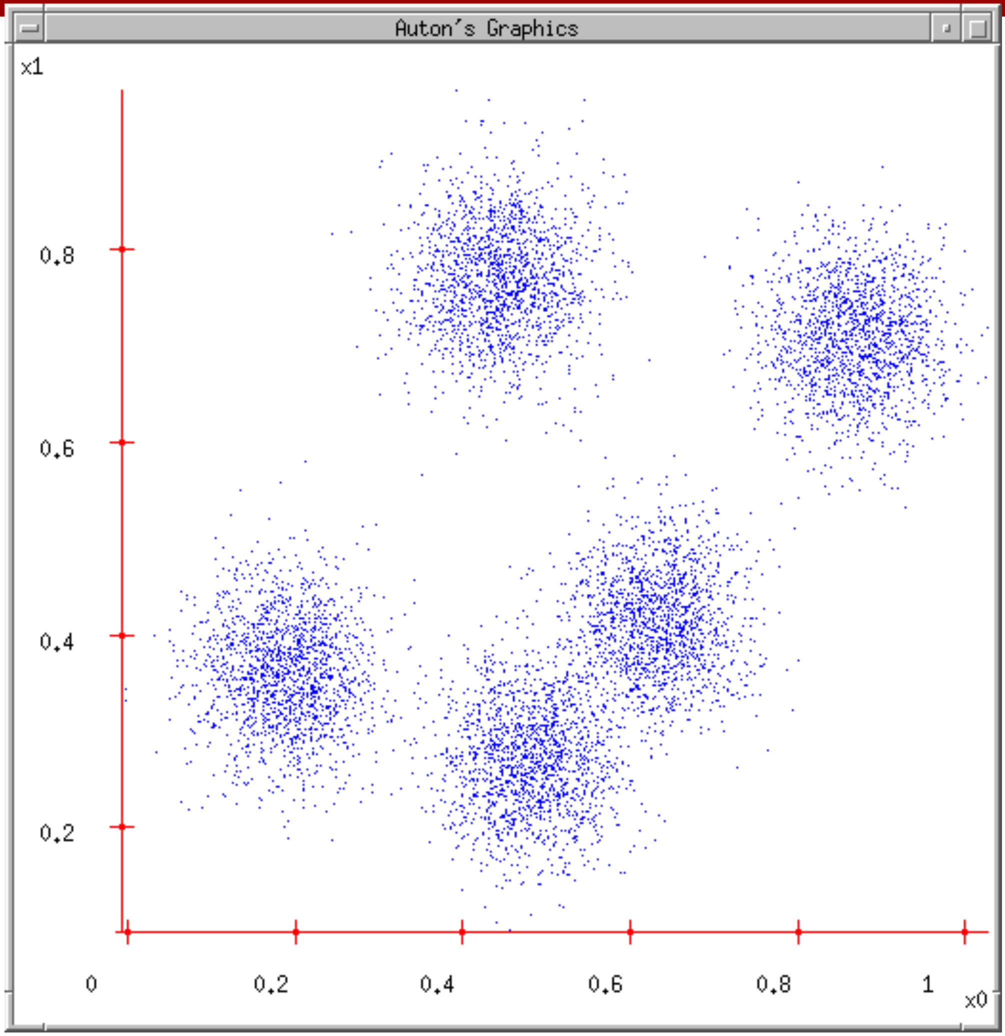
Supervised Learning



Unsupervised Learning

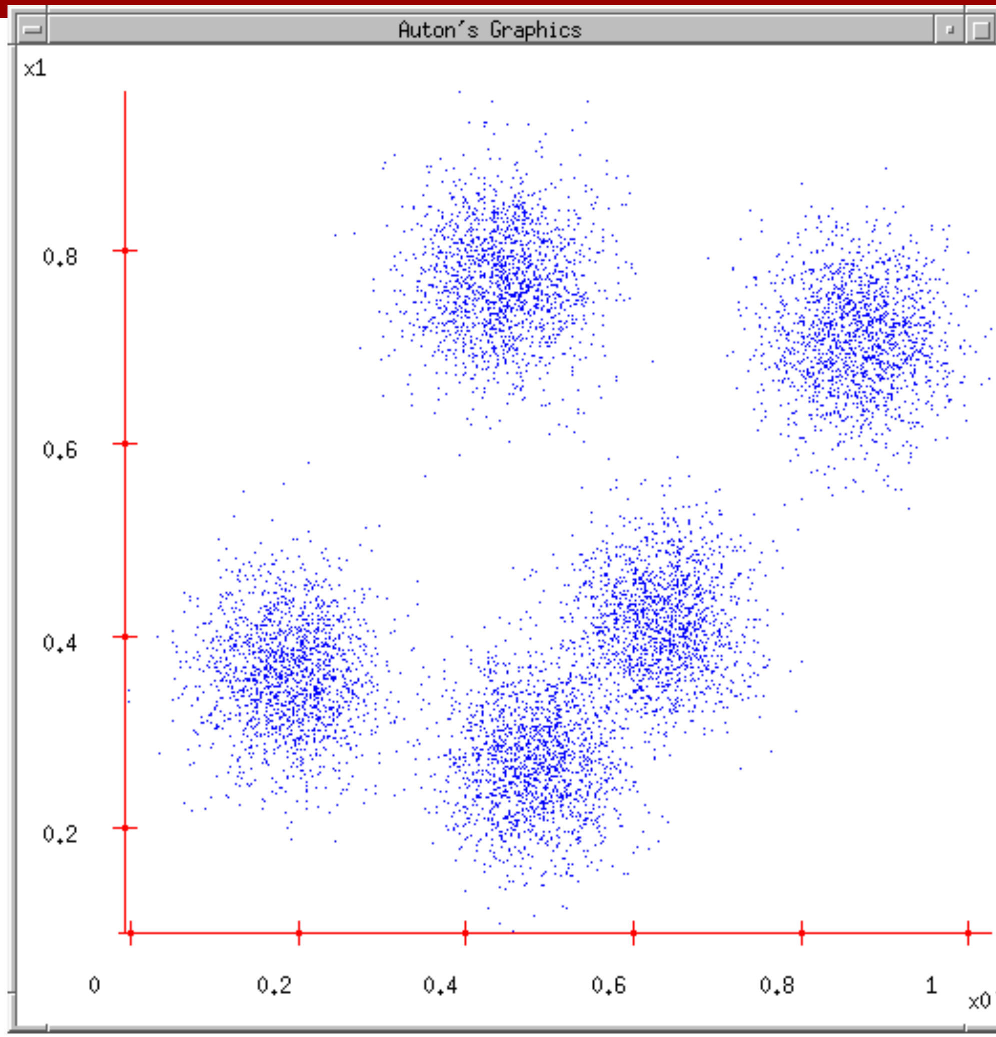


Some Data



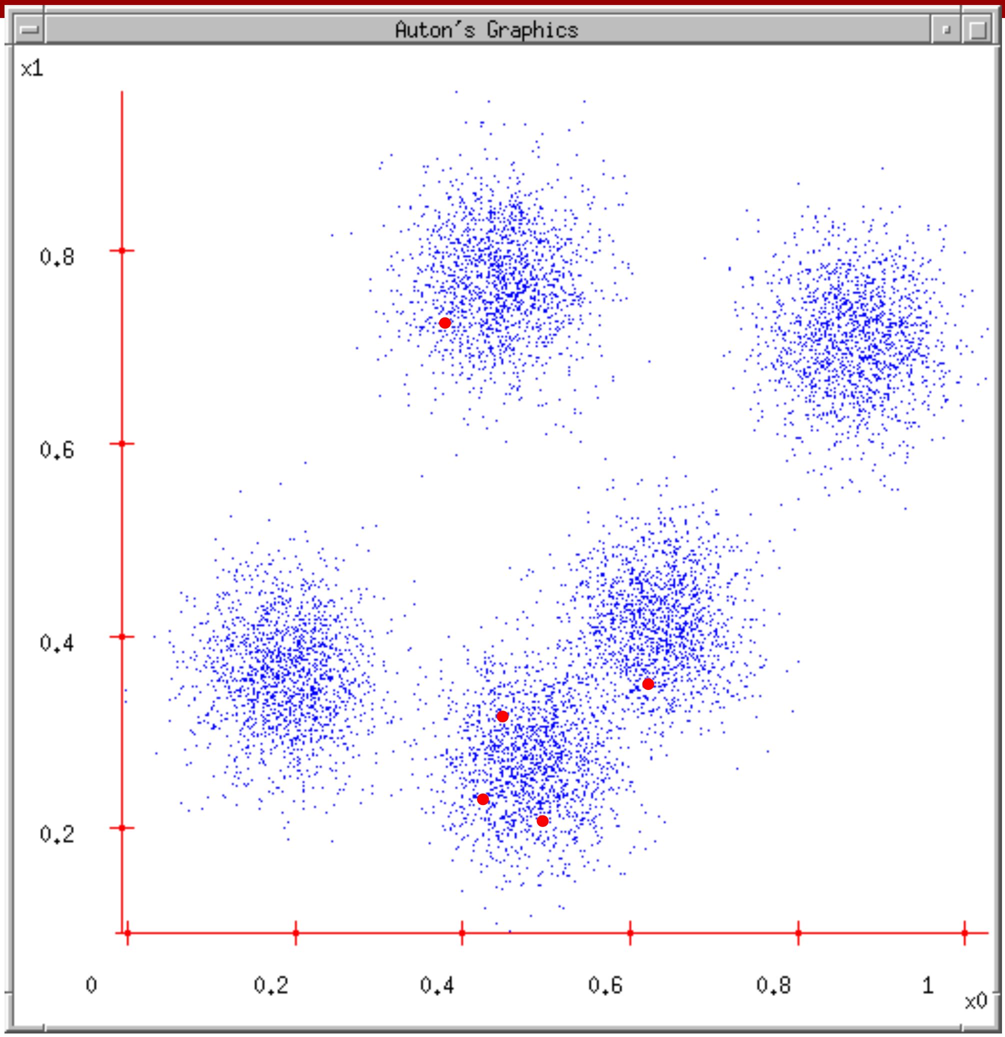
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



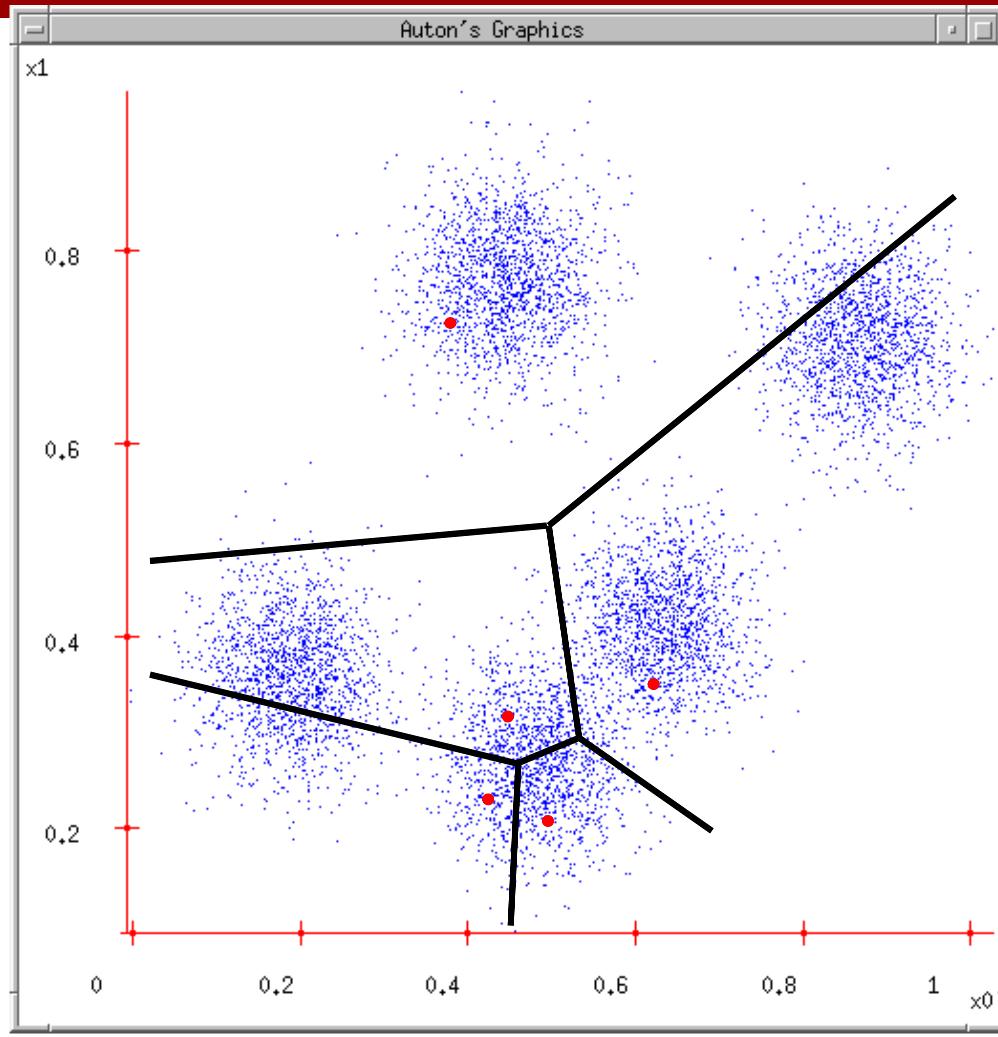
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



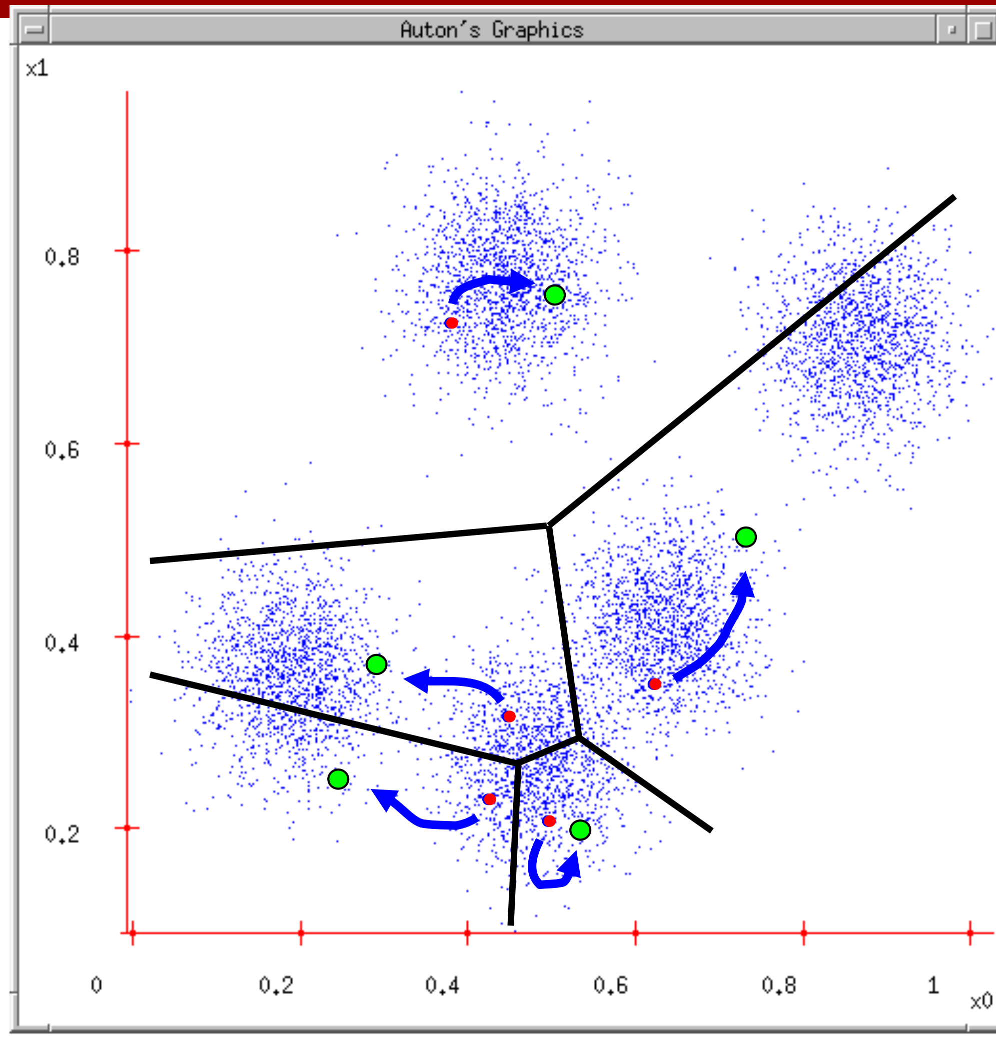
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.



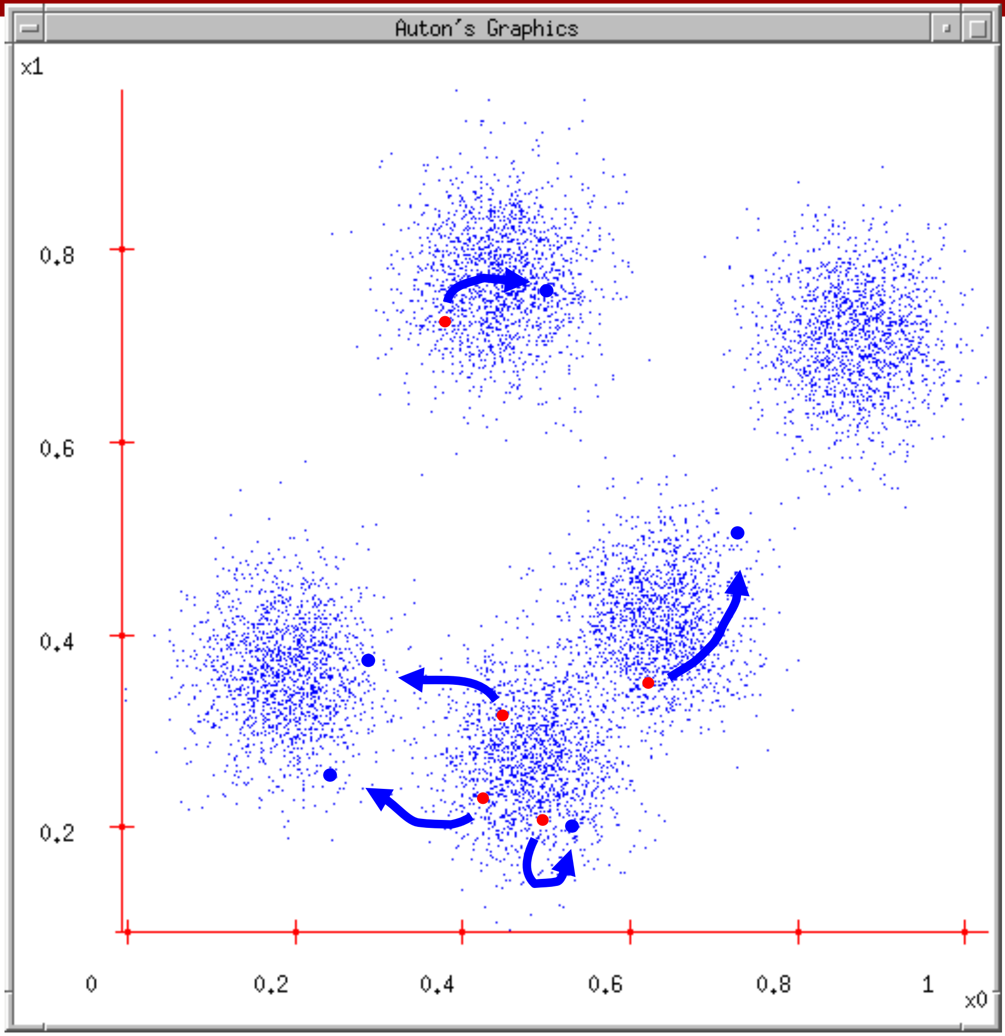
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



K-means

- Demo
  - http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html

K-means

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

- **Assign**:
  - Assign each point $i \in \{1, \ldots n\}$ to nearest center:
  - $$C(i) \longleftarrow \underset{j}{\operatorname{argmin}} \, ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

- **Recenter**:
  - μ_j becomes centroid of its points

What is K-means optimizing?

- Objective F(μ,C): function of centers μ and point allocations C:
  –
  $$F(\boldsymbol{\mu}, C) = \sum_{i=1}^{N} ||\mathbf{x}_i - \boldsymbol{\mu}_{C(i)}||^2$$
  – 1-of-k encoding
  $$F(\boldsymbol{\mu}, \boldsymbol{a}) = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

- Optimal K-means:
  – $\min_{\mu} \min_{a} F(\mu, a)$

Coordinate descent algorithms

- Want: $\min_a \min_b F(a,b)$
- Coordinate descent:
  - fix a, minimize b
  - fix b, minimize a
  - repeat
- Converges!!!
  - if F is bounded
  - to a (often good) local optimum
- K-means is a coordinate descent algorithm!

K-means as Co-ordinate Descent

- Optimize objective function:

$$\min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \min_{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N} F(\boldsymbol{\mu}, \boldsymbol{a}) = \min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \min_{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N} \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

- Alternate between
  - Fix μ, optimize a (i.e. C)
  - Fix a (i.e. C), optimize μ

Supervised vs Unsupervised Learning

Supervised Learning

Given: (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.

Reinforcement Learning

Given: (e, r)
Environment e, Reward function r (evaluative feedback)

Goal: Maximize expected reward

Examples: Robotic control, video games, board games, etc.

Unsupervised Learning

Given: 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, CS 231n

Supervised vs Unsupervised Learning

Supervised Learning

Given: (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.

Reinforcement Learning

Given: (e, r)
Environment e, Reward function r (evaluative feedback)

Goal: Maximize expected reward

Examples: Robotic control, video games, board games, etc.

Unsupervised Learning

Training data is cheap

Given: Data x
Just data, no labels!

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

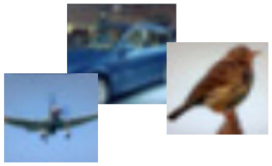
Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

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

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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, CS 231n

Generative Classification vs Discriminative Classification vs Density Estimation

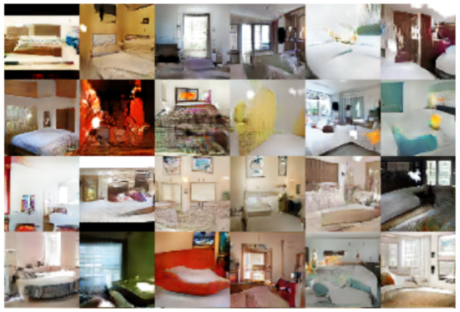
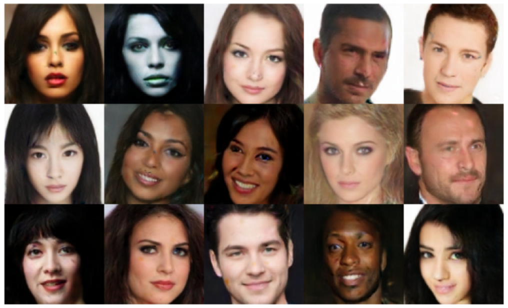
- Generative Classification
  - Model p(x, y); estimate p(x|y) and p(y)
  - Use Bayes Rule to predict y
  - E.g Naïve Bayes

- Discriminative Classification (not a Generative Model)
  - Estimate p(y|x) directly
  - E.g. Logistic Regression

- Density Estimation
  - Model p(x)
  - E.g. VAEs

Why Generative Models?

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

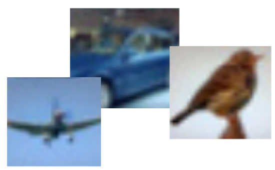




- Generative models of time-series data can be used for simulation and planning (model based reinforcement learning!)

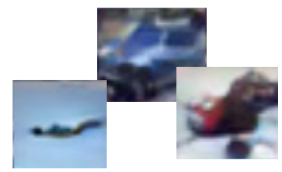
- Training generative models can also enable inference of latent representations that can be useful as general features

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

Taxonomy of Generative Models

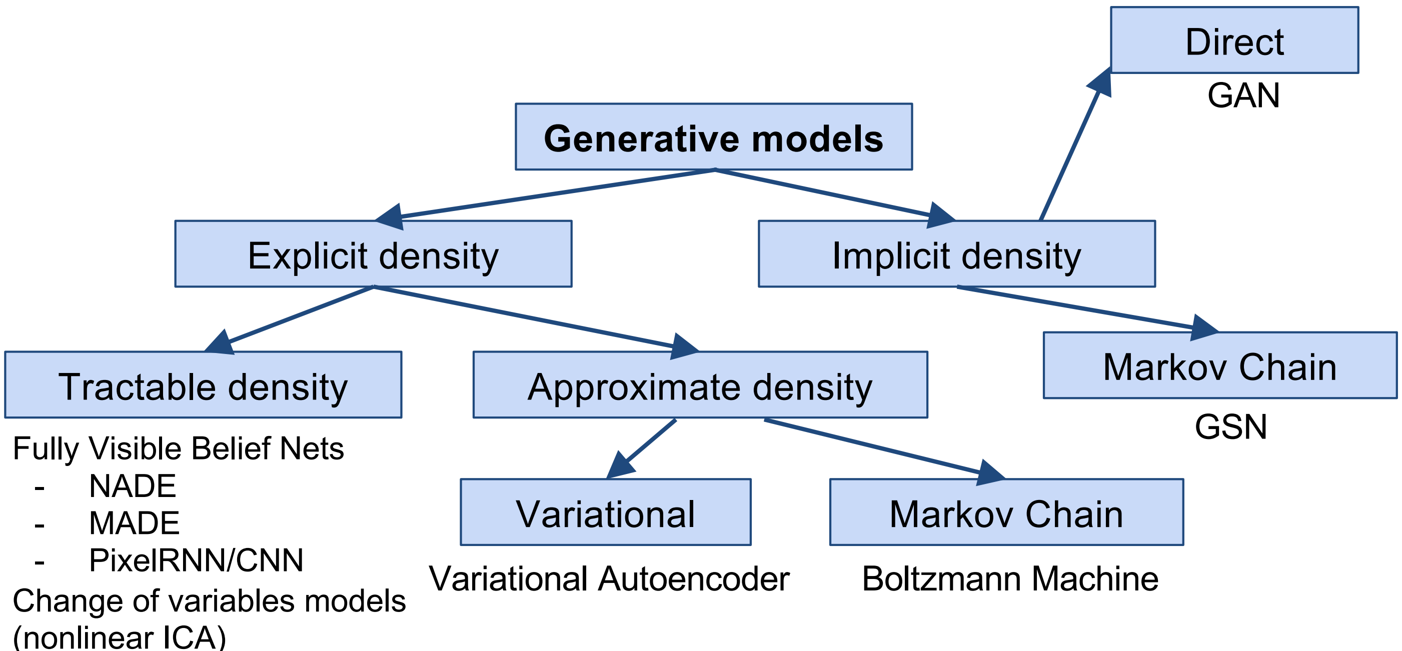


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Taxonomy of Generative Models

We will discuss 3 most popular types of generative models

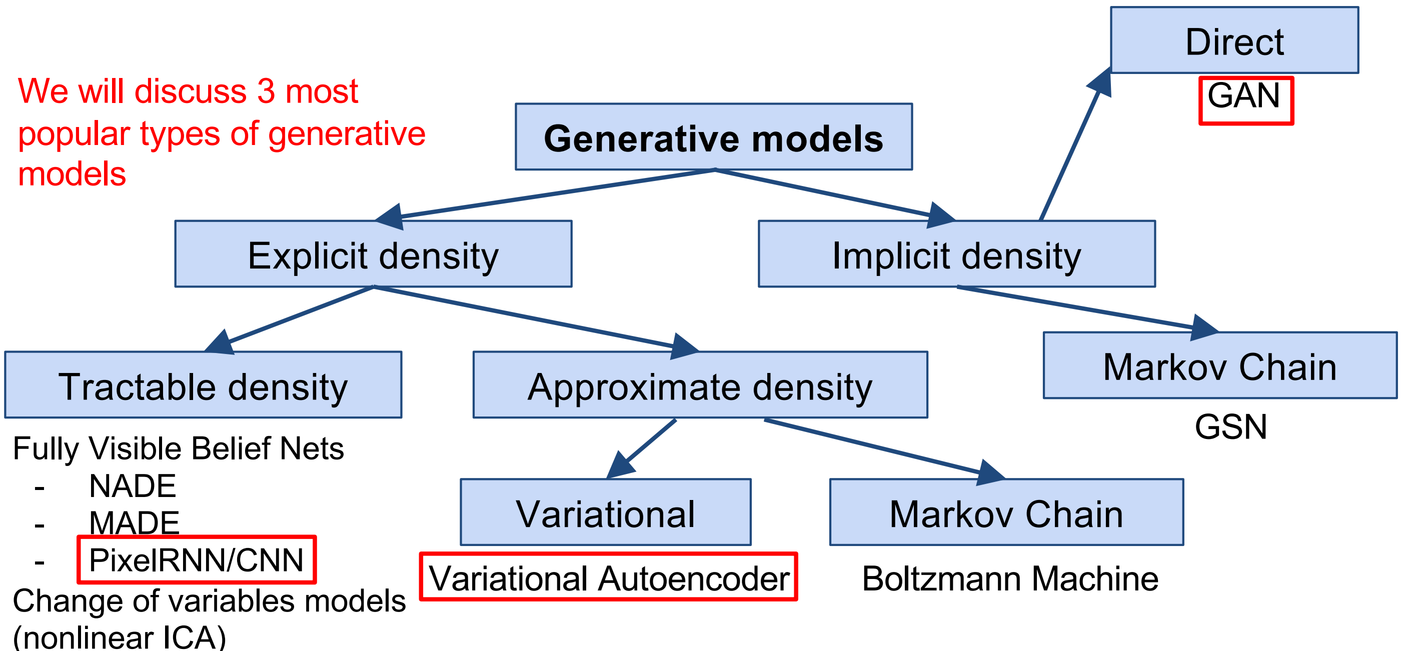


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

PixelRNN and PixelCNN

Fully Visible Belief Network

Explicit density model

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

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

Likelihood of image x

Probability of i'th pixel value given all previous pixels

Then maximize likelihood of training data

Fully Visible Belief Network

Explicit density model

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

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

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

Fully Visible Belief Network

Explicit density model

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

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

Likelihood of image x

Probability of i'th pixel value given all previous pixels

Will need to define ordering of "previous pixels"

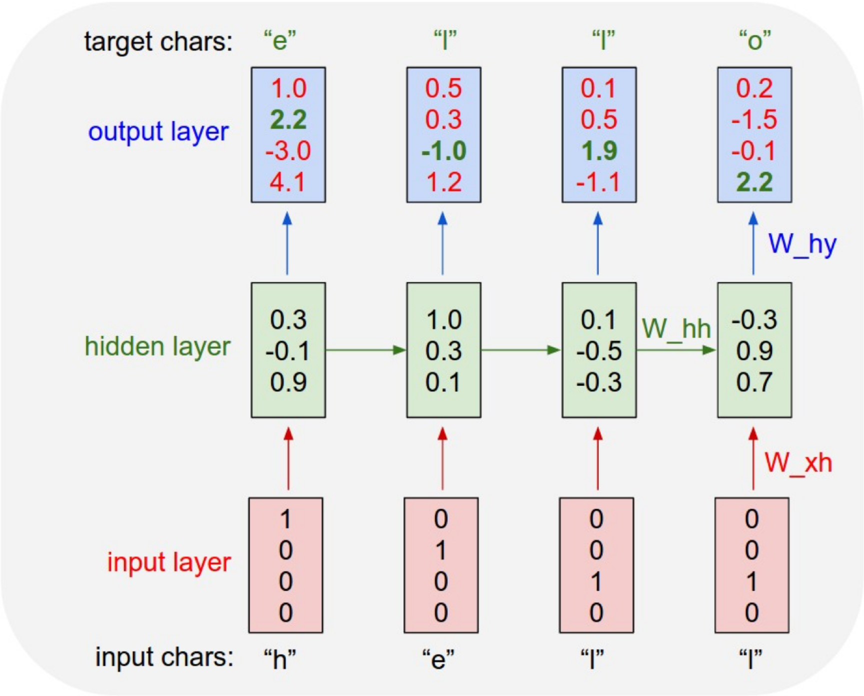
Complex distribution over pixel values
=> Express using a neural network!

Then maximize likelihood of training data

Example: Character-level Language Model

Vocabulary: [h,e,l,o]

Example training sequence: **“hello”**

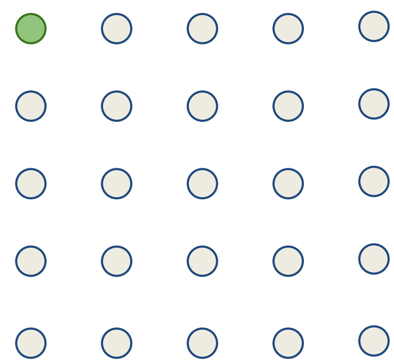


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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

Generate image pixels starting from corner

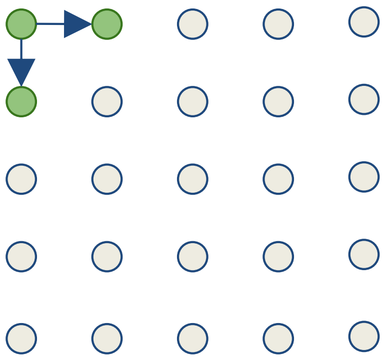
Dependency on previous pixels modeled using an RNN (LSTM)



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

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)

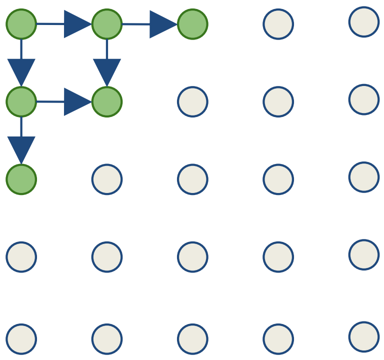


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)



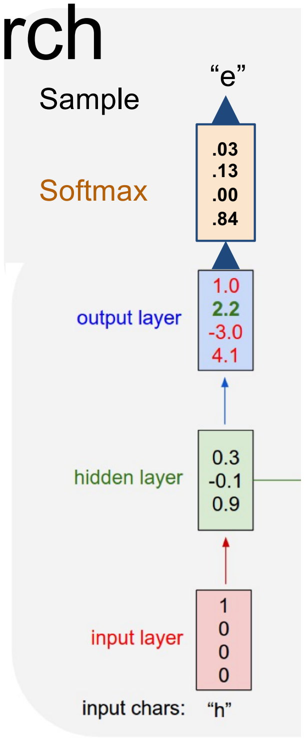
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Test Time: Sample / Argmax / Beam Search

Example: Character-level Language Model Sampling

Vocabulary: [h,e,l,o]

At test-time sample characters one at a time, feed back to model

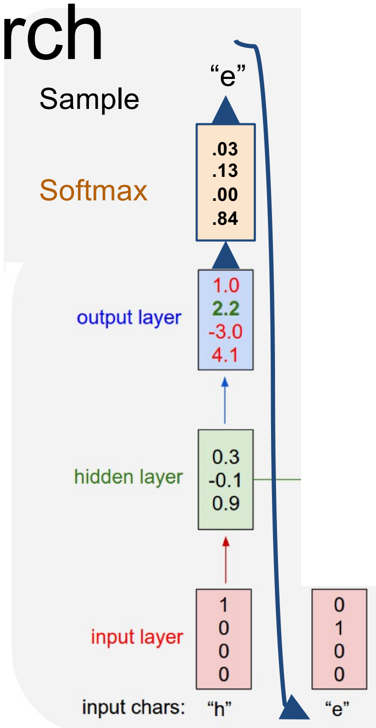


Test Time: Sample / Argmax / Beam Search

Example:
Character-level
Language Model
Sampling

Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model



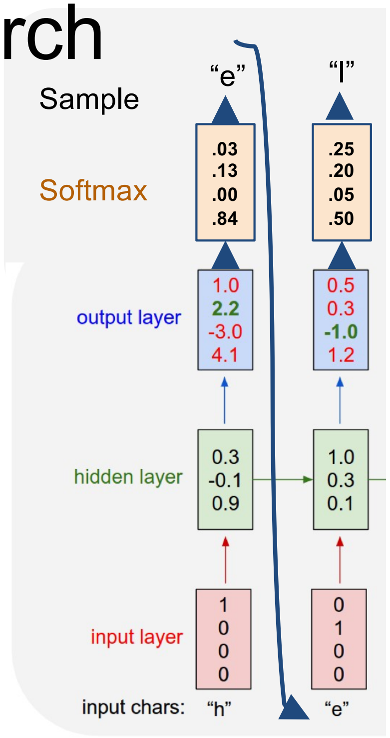
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Test Time: Sample / Argmax / Beam Search

Example: Character-level Language Model Sampling

Vocabulary: [h,e,l,o]

At test-time sample characters one at a time, feed back to model

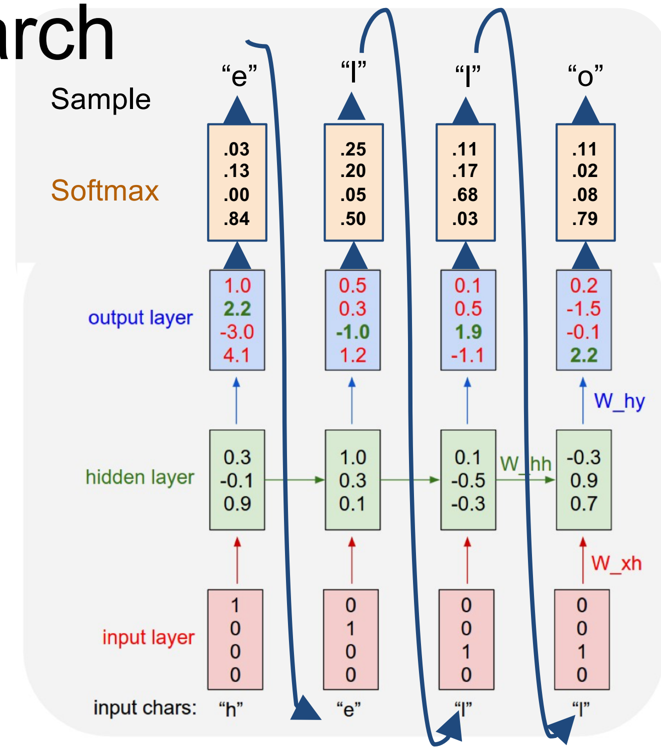


Test Time: Sample / Argmax / Beam Search

Example: Character-level Language Model Sampling

Vocabulary: [h,e,l,o]

At test-time sample characters one at a time, feed back to model



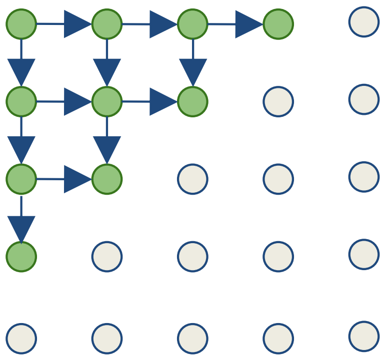
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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!

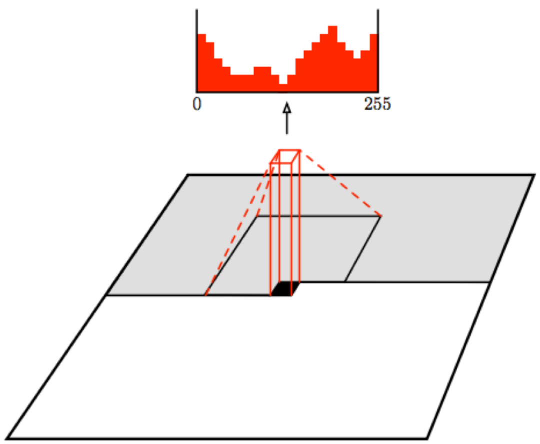


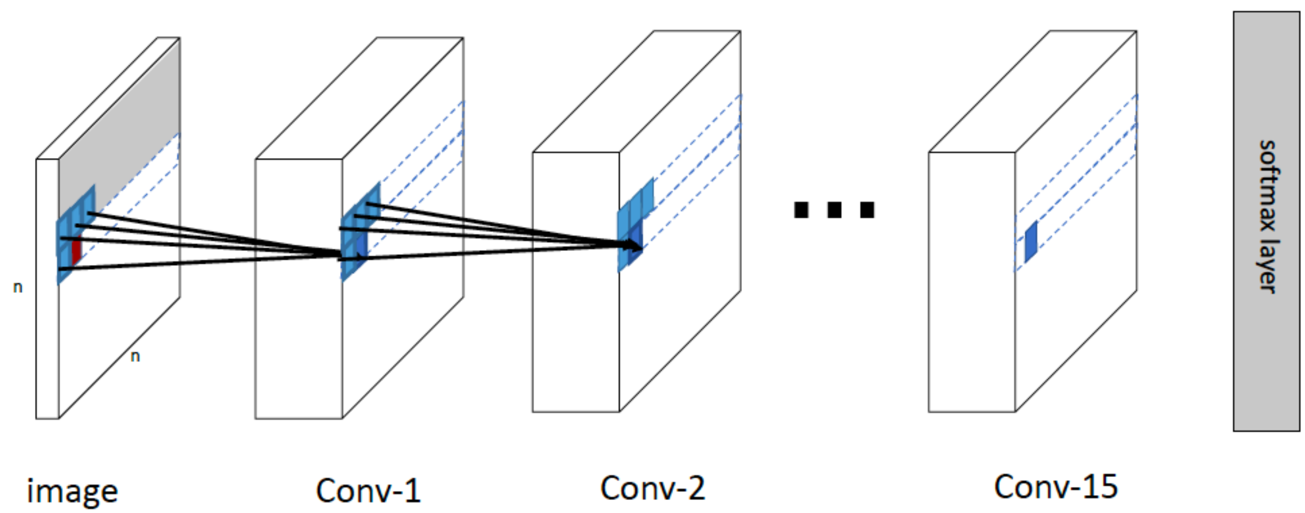
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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

Still generate image pixels starting from corner

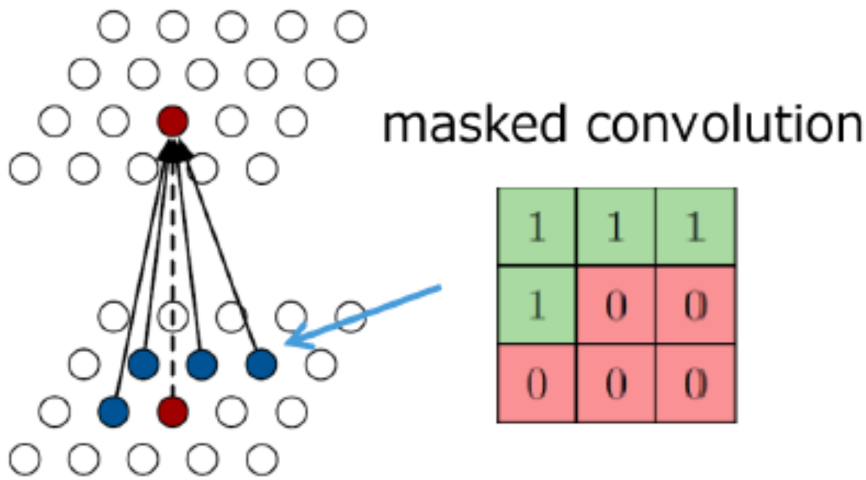
Dependency on previous pixels now modeled using a CNN over context region




n
n
softmax layer
image
Conv-1
Conv-2
Conv-15

Masked Convolutions

- Apply masks so that a pixel does not see "future" pixels



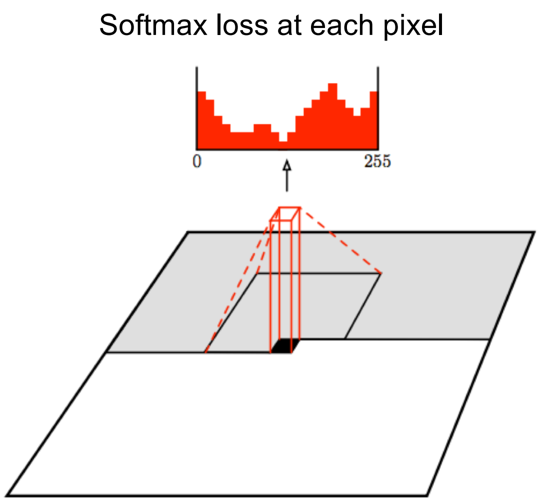
PixelCNN *[van der Oord et al. 2016]*

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

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



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

Still generate image pixels starting from corner

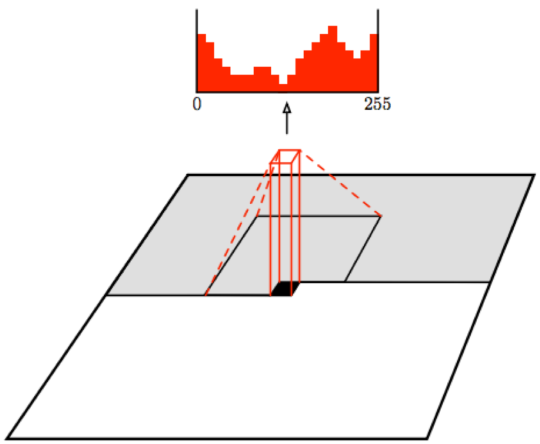
Dependency on previous pixels now modeled using a CNN over context region

Training: maximize likelihood of training images

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

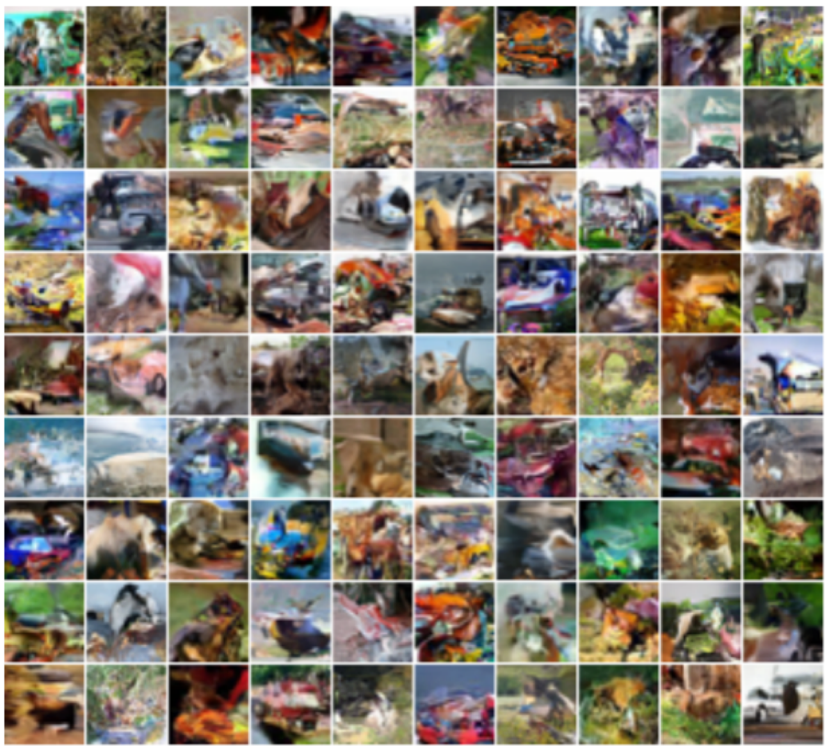
Generation must still proceed sequentially
=> still slow

Softmax loss at each pixel

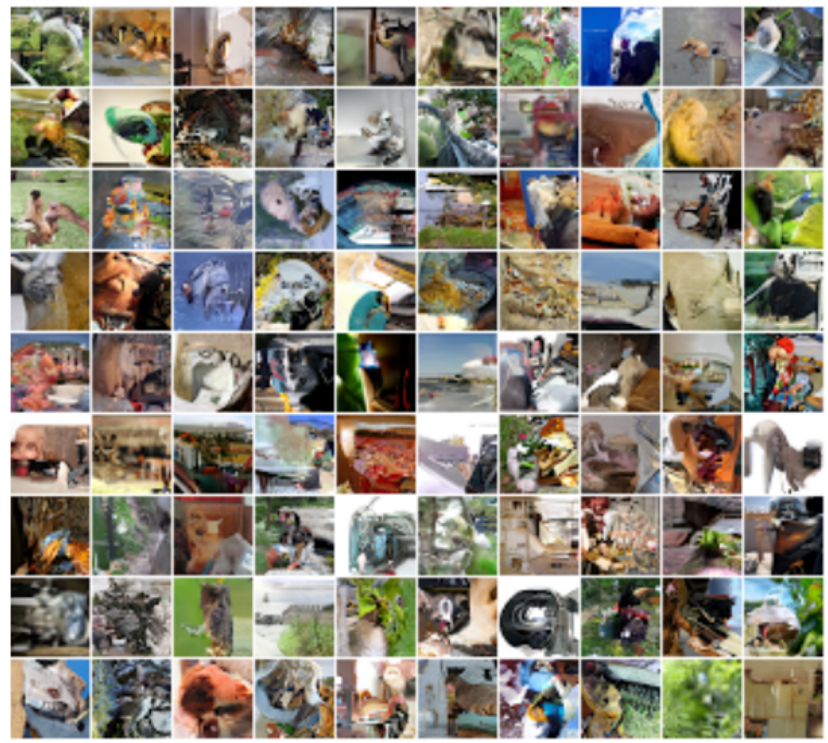


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Generation Samples (PixelRNN)



32x32 CIFAR-10



32x32 ImageNet

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Image Completion

Figure 1. Image completions sampled from a PixelRNN.

Results from generating sounds

- https://deepmind.com/blog/wavenet-generative-model-raw-audio/

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++)

Conclusion

- Unsupervised Learning
  - Comparison to Supervised and Reinforcement Learning
  - Review of K-Means

- e.g., Generative Models
  - Varieties
  - PixelRNN and PixelCNN

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

The End