CS 236: Deep Generative Models

Stefano Ermon and Aditya Grover

Stanford University

URL: deepgenerativemodels.github.io

Introduction

Challenge: understand complex, unstructured inputs

Computer Vision

Computational Speech

Natural Language Processing

Robotics

Introduction

Richard Feynman: “*What I cannot create, I do not understand*”

Generative modeling: “*What I understand, I can **create***”

Generative Modeling: Computer Graphics

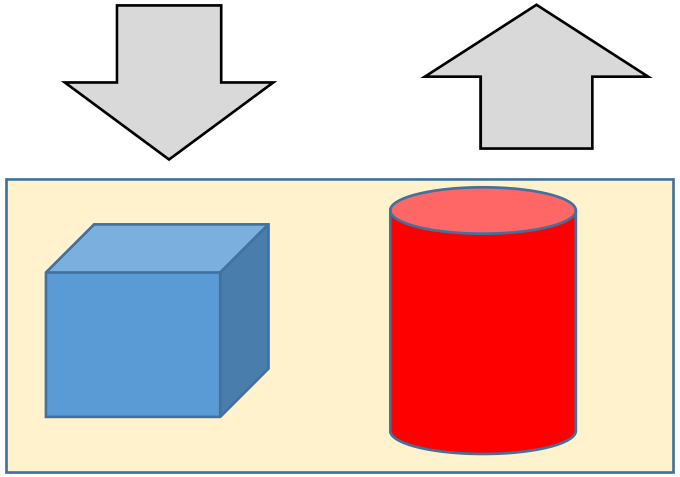
How to generate natural images with a computer?

High level description

Cube(color=blue, position=, size=, …)
Cylinder(color=red, position=, size=,..)

Generation (graphics)

Inference (vision)



Raw sensory outputs

Our models will have **similar structure (generation + inference)**

Statistical Generative Models

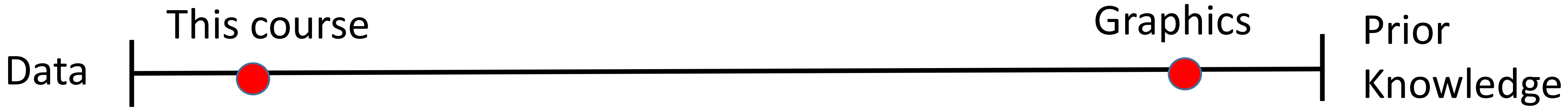
Statistical generative models are **learned from data**

Data
(e.g., images of bedrooms)

\+

Prior Knowledge
(e.g., physics, materials, ..)

Priors are always necessary, but there is a spectrum



Statistical Generative Models

A statistical generative model is a **probability distribution** p(x)

- **Data:** samples (e.g., images of bedrooms)
- **Prior knowledge:** parametric form (e.g., Gaussian?), loss function (e.g., maximum likelihood?), optimization algorithm, etc.

Image x  → A probability distribution p(x) → probability p(x)

It is generative because **sampling from p(x) generates new images**

Discriminative vs. generative

Discriminative: classify bedroom vs. dining room

The input image X is always given. **Goal**: a good decision boundary, via **conditional distribution**

P(Y = Bedroom | X=  1

Ex: logistic regression, convolutional net, etc.

Generative: generate X

The input X is **not** given. Requires a model of the **joint distribution**

P(Y = Bedroom , X= )02

Discriminative vs. generative

Joint and conditional are related via **Bayes Rule**:

P(Y = Bedroom | X= 

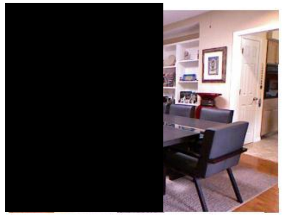
P(Y = Bedroom, X= 

P(X= 

Discriminative: X is always given, does not need to model

P(X= 

Therefore it cannot handle missing data

P(Y = Bedroom | X= 

Conditional Generative Models

Class **conditional generative models** are also possible:

P(X=  = Bedroom)

It's often useful to condition on rich side information Y

P(X=  ption = "A black table with 6 chairs")

A discriminative model is a very simple conditional generative model of Y:

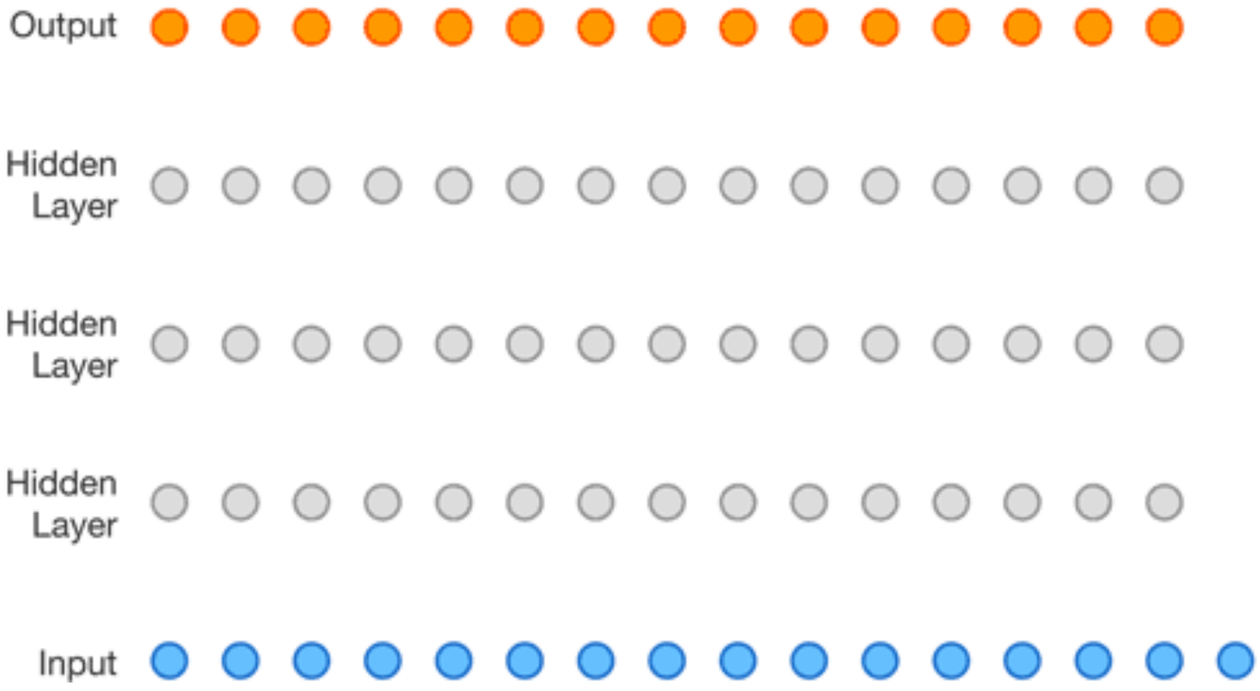
P(Y = Bedroom | X= 

Progressive Growing of GANs

Karras et al., 2018

WaveNet

Generative model of speech signals



Text to Speech

Parametric

Concatenative

WaveNet

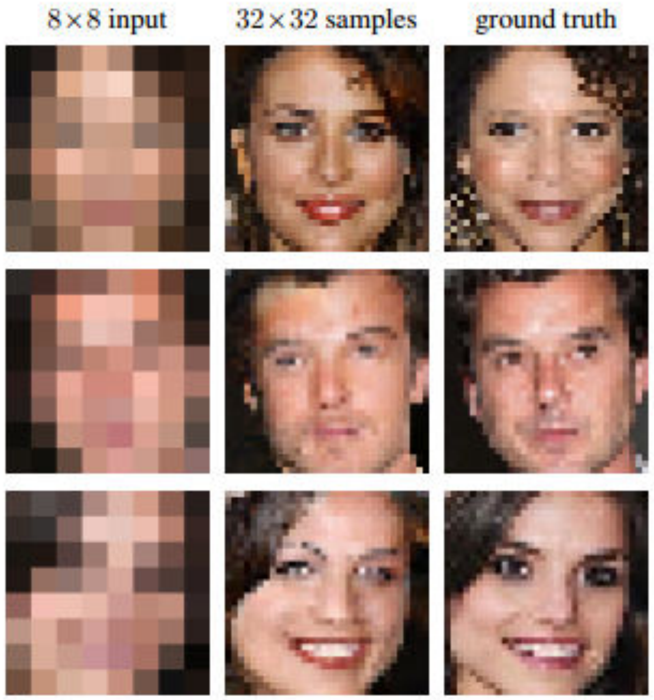
Unconditional

Music

van den Oord et al, 2016c

Image Super Resolution

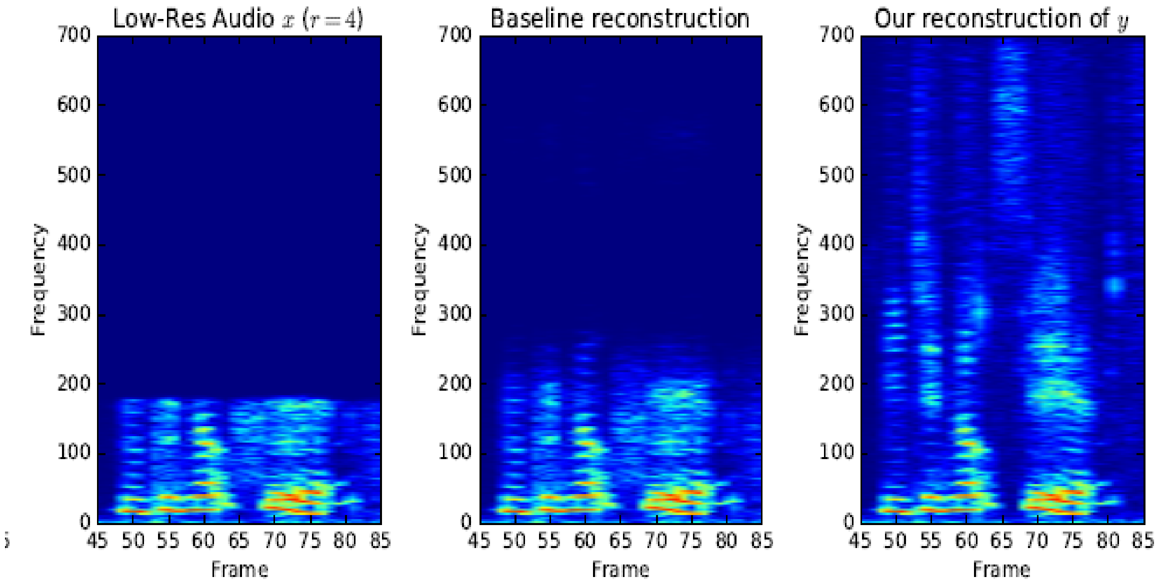
Conditional generative model P(high res image | low res image)



Ledig et al., 2017

Audio Super Resolution

Conditional generative model P(high-res signal | low-res audio signal)



Low res signal

High res audio signal

Kuleshov et al., 2017

Machine Translation

Conditional generative model P(English text| Chinese text)

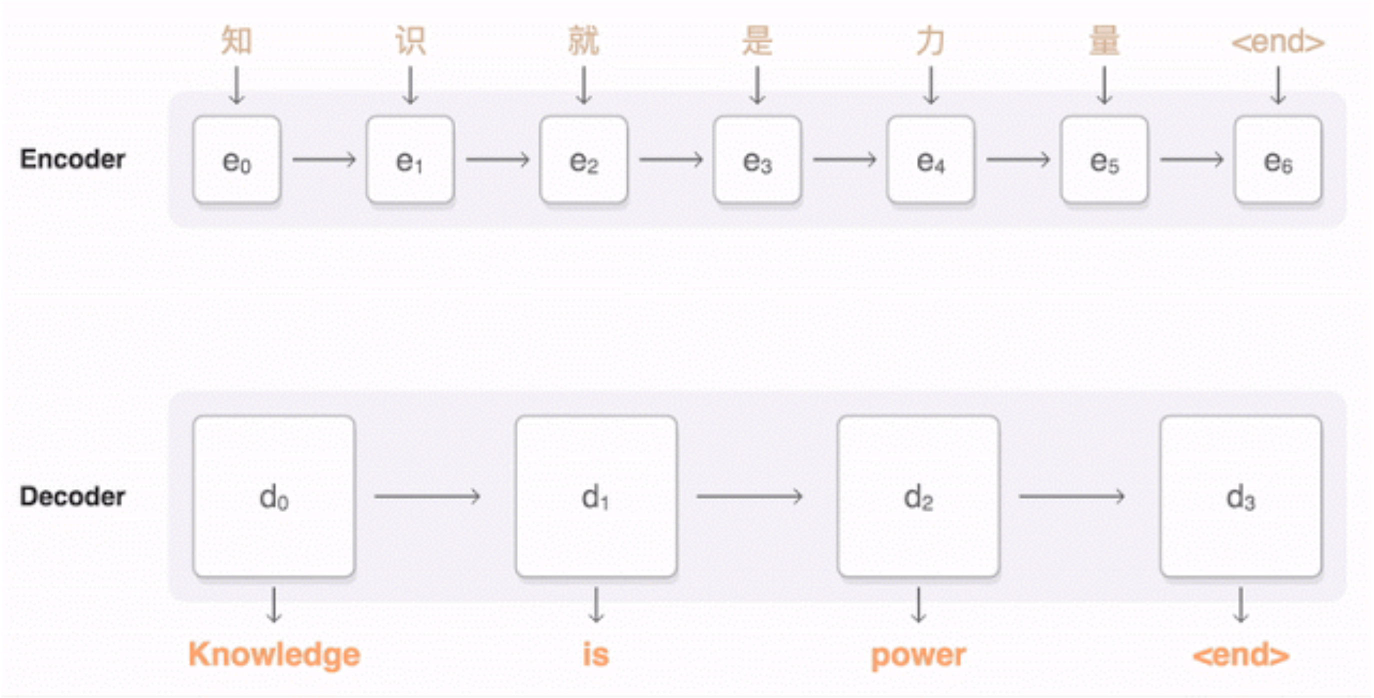


Figure from Google AI research blog.

Image Translation

Conditional generative model P(zebra images| horse images)

Zhu et al., 2017

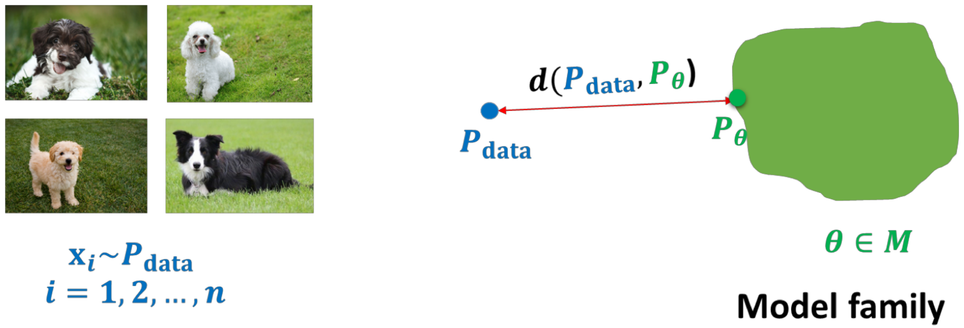
Imitation Learning

Conditional generative model P(actions | past observations)

Li et al., 2017

Roadmap and Key Challenges

- **Representation**: how do we model the joint distribution of many random variables?
  - Need compact representation
- **Learning**: what is the right way to compare probability distributions?



- **Inference**: how do we invert the generation process (e.g., vision as inverse graphics)?
  - Unsupervised learning: recover high-level descriptions (features) from raw data

Syllabus

- Fully observed likelihood-based models
  - Autoregressive
  - Flow-based models
- Latent variable models
  - Variational learning
  - Inference amortization
  - Variational autoencoder
- Implicit generative models
  - Two sample tests, embeddings, F-divergences
  - Generative Adversarial Networks
- Learn about algorithms, theory & applications

Prerequisites

- Basic knowledge about machine learning from at least one of CS 221, 228, 229 or 230.
- Basic knowledge of probabilities and calculus:
  - Gradients, gradient-descent optimization, backpropagation
  - Random variables, independence, conditional independence
  - Bayes rule, chain rule, change of variables formulas
- Proficiency in some programming language, preferably Python, required.

Logistics

- Class webpage: https://deepgenerativemodels.github.io/
- http://piazza.com/stanford/fall2018/cs236
- There is no required textbook. Reading materials and course notes will be provided.
- Suggested Reading: *Deep Learning* by Ian Goodfellow, Yoshua Bengio, Aaron Courville. Online version available free here.
- Lecture notes are under construction
- Teaching Assistants: Yang Song, Jiaming Song, Rui Shu, Casey Chu, Nish Khandwala
- Office hours: See calendar on class website

Logistics – Grading policies

- Grading Policy:
  - Three homeworks (15% each): mix of conceptual and programming based questions
  - Midterm: 15%
  - Course Project: 40%
    - Proposal: 5%
    - Progress Report: 10%
    - Poster Presentation: 10%
    - Final Report: 15%

Projects

- Course projects will be done in groups of up to 3 students and can fall into one or more of the following categories:
  - Application of deep generative models on a novel task/dataset
  - Algorithmic improvements into the evaluation, learning and/or inference of deep generative models
  - Theoretical analysis of any aspect of existing deep generative models
- Teaching staff will suggest possible projects
- We will provide Google Cloud coupons