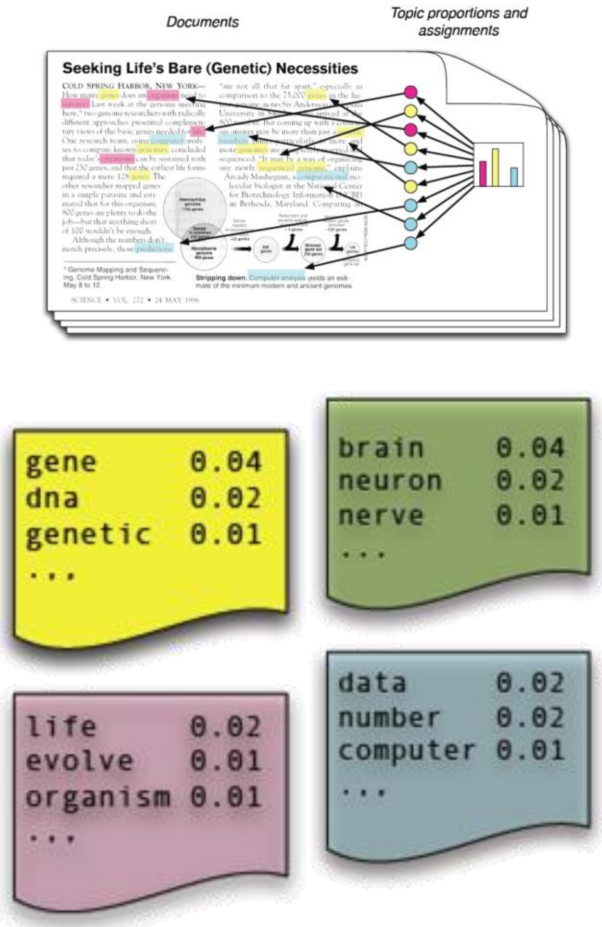
COMPSCI590.7 Algorithmic Aspects of Machine Learning

Summary

Outline

- What have we learned
  - Estimating parameters for unsupervised learning problems
  - Techniques: spectral/tensor/geometry/optimization
  - Main message
- What have we not covered
  - Why and where can you learn these

Unsupervised Learning



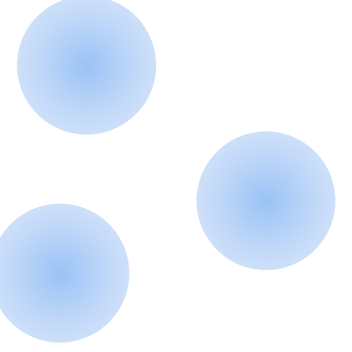
Given: Data

Assumption: Is generated from a prob. distribution that's described by small # of parameters. ("Model")

Learning ≈ Find good fit to these parameter values

Use spectral techniques

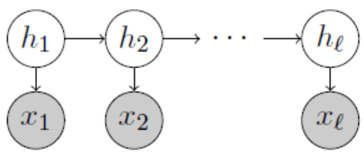
- Problems: Finding communities and mixture of Gaussians

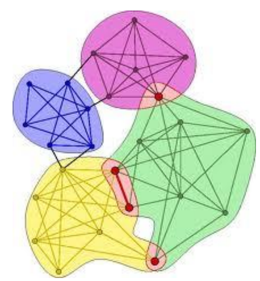


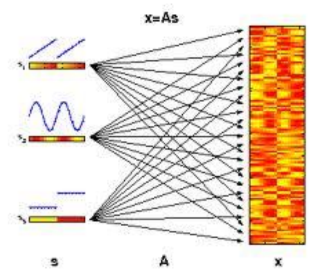
- Tools: Random matrix theory, Wedin’s Theorem
- Good for: problems where the signal is in a lower dimensional subspace.

Use tensor decomposition

- Problems: Hidden Markov Model, Overlapping Communities, Independent Component Analysis…



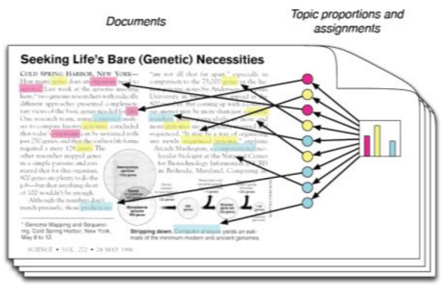


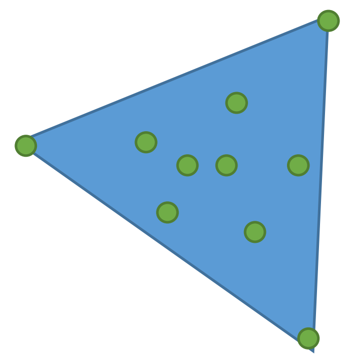


- Tools: Jenrich’s algorithm, low rank tensor
- Good for: problems with three independent views or problems with nice tensor structure.

Use Geometry

- Problem: Nonnegative Matrix Factorization, Topic models under separability assumption.

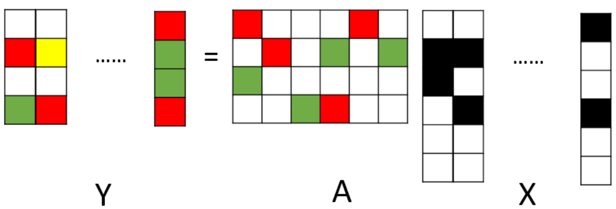


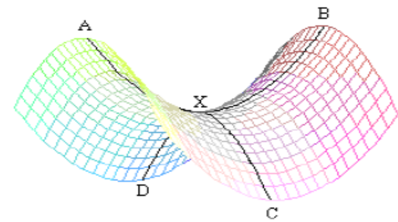


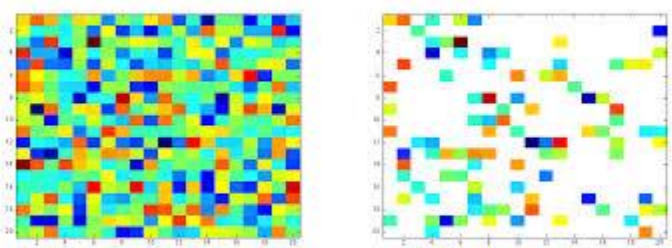
- Tools: Convex hull, geometric intuitions
- Good for: Problems with a geometric interpretation

Non-convex Optimization

- Problems: Dictionary learning, strict-saddle problems, matrix completion



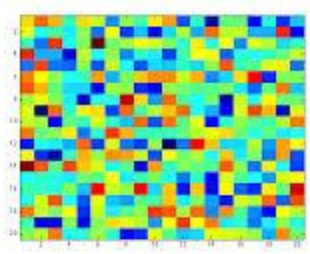


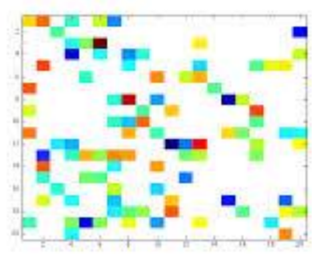


- Tools: Potential function, step-by-step analysis
- Widely used, but not easy to analyze

Use convex relaxations

- Problem: Matrix Completion





- Tools: convex programs, Rademacher complexity, duality
- Widely used, may be slow in practice.

Main Message

- We can design algorithms with provable guarantees for many unsupervised learning problems.

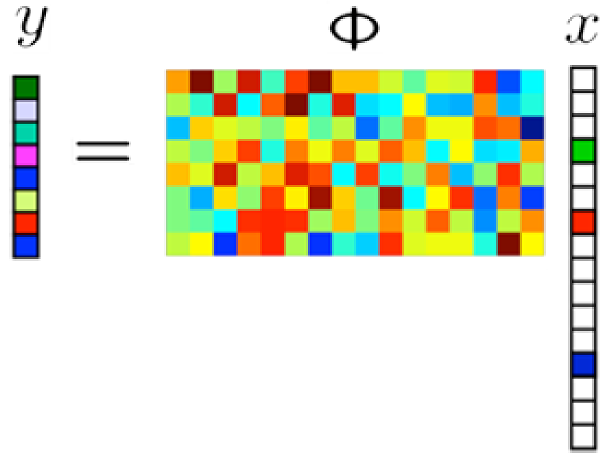
- When the problem seems hard in worst case, try to make assumptions.

- Follow the intuitions to design better algorithms.

Other related materials

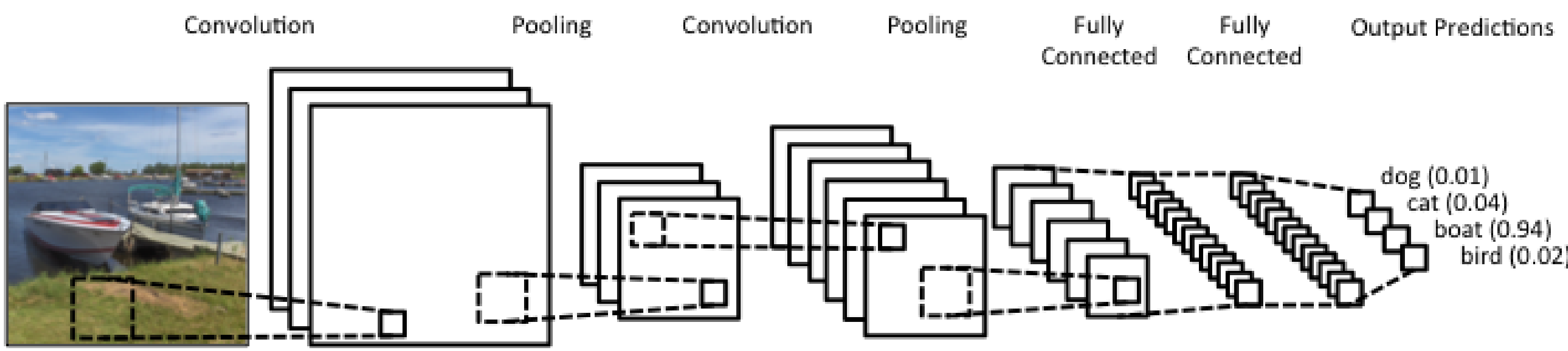
Many other unsupervised models

- Sparse PCA, Sparse recovery (compressed sensing) Super-resolution, Phase Retrieval, Subspace Clustering etc.



- Many provable algorithms in applied math literature.

Deep Learning



- Hot topic in machine learning now.
- Best empirical performance for many problems, especially those involving images and voice.
- Theoretical properties are major open problems.

[image from clarifai.com]

Deep Learning

- Some attempts to understand deep neural networks (from algorithmic perspective)
  - A provably efficient algorithm for training deep networks [Livni Shalev-Shwartz Shamir 2013]
  - Provable Bounds for Learning Some Deep Representations [Arora Bhaskara G Ma 2014]
  - Learning Polynomials with Neural Networks [Andoni Panigrahy, Valiant, Zhang 2014]
  - Beating the Perils of Non-Convexity: Guaranteed Training of Neural Networks using Tensor Methods [Janzamin Sedghi Anandkumar 2015]
  - A Generative Model View of Neural Networks with Rectifier Linear Gates [Arora Liang Ma and others, on arxiv soon?]

Sum-of-squares techniques

- The strongest convex relaxations we have.
- Can be used to get a “first poly time algorithm”.
- Can be used for showing a problem is hard.
- See survey by Boaz Barak and David Steurer

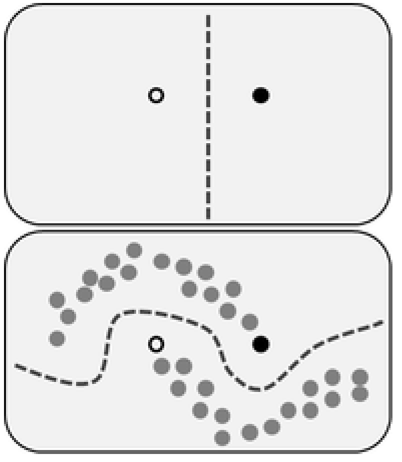
Computational Learning Theory

- PAC learning, VC dimensions, generalization bounds, SVM, boosting,…

- Can be a course by itself.
- Some parts are covered in STA561/COMPSCI571
- See also Avrim Blum’s course

Other learning models

- Active Learning, Semi-supservised Learning



- Many interesting open problems.

[image from Wikipedia]

Bayesian Inference

- Graphical models, MCMC, counting

- Some materials are covered in STA561/COMPSCI571.
- Some also appears in my course Spring 2016: COMPSCI 630 Randomized algorithms
- Also related: new course this Spring 2016: Information, Physics, and Computation

Online Learning

- Experts, Bandits, …

- Take Kamesh's course Spring 2016
  COMPSCI 590. Optimization and Decision-making
  Under Uncertainty

How to make algorithms faster

- By designing better algorithms on a single machine
  - Better optimization techniques
  - Dimension reduction
    (will be discussed in randomized algorithms course)

- By running the algorithm on multiple machines/GPUs.

Final Project

- Project report due Dec. 7th.

- Email me if you have any questions!

Thanks!