Learning from Labeled and Unlabeled Data using Markov Random Fields

Zoubin Ghahramani† and Xiaojin Zhu∗

† Gatsby Computational Neuroscience Unit
University College London

∗ School of Computer Science
Carnegie Mellon University

Aladdin Workshop, CMU, Jan 2003

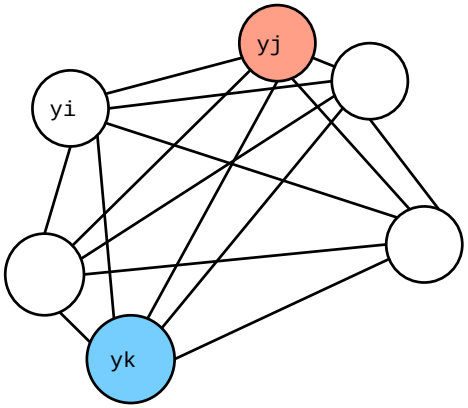
Problem Statement

Labeled Data: $(x_1, y_1) \ldots (x_l, y_l)$, where $Y_L = (y_1 \ldots y_l)$ and $y_i \in \{1 \ldots C\}$.
Unlabeled Data: $(x_{l+1}, y_{l+1}) \ldots (x_{l+u}, y_{l+u})$, where $Y_U = (y_{l+1} \ldots y_{l+u})$ are **unobserved**, usually $l \ll u$. Let $X = (x_1 \ldots x_{l+u})$ and $x_i \in \mathcal{R}^D$ (for simplicity).

Problem is to estimate Y_U from X and Y_L.

Simple Model: $$P(Y|X) = \frac{1}{Z} \exp \left[\sum_{i=1}^{l+u} \sum_{j<i} \delta(y_i, y_j) A_{ij} \right]$$

A_{ij} is the **affinity** between data point i and j, e.g.: $$A_{ij} = \exp\left(-\frac{d_{ij}^2}{2\sigma^2} \right)$$



Solutions to Label Inference Problem:

- compute $P(Y_U|X, Y_L, \sigma)$ or
- $y_i^{MAP} = \arg\max_{y_i} P(y_i|Y_L, X, \sigma)$, or
- $Y_U^{MAP} = \arg\max_{Y_U} P(Y_U|Y_L, X, \sigma)$
  (multiway cut problem / metric labeling)

This is a Markov Random Field, Potts Model, Boltzmann Machine, or Undirected Graphical Model.

Learning Problem

Simple Model: $$P(Y|X) = \frac{1}{Z}\exp\left[\sum_{i=1}^{l+u}\sum_{j<i}\delta(y_i, y_j)A_{ij}\right]$$

A_{ij} is the **affinity** between data point i and j, e.g.: $$A_{ij} = \exp\left(-\frac{d_{ij}^2}{2\sigma^2}\right)$$

Learning problem: how do we set σ?

"Obvious" Solution: Maximum Likelihood or MAP:

$$\begin{aligned}\sigma_{MAP} &= \arg\max_{\sigma} P(\sigma|X, Y_L) = \arg\max_{\sigma} P(Y_L|X, \sigma)P(\sigma) \\ &= \arg\max_{\sigma} \sum_{Y_U} P(Y_L, Y_U|X, \sigma)P(\sigma)\end{aligned}$$

$$\begin{aligned}\frac{\partial}{\partial\sigma}\log P(\sigma|X, Y_L) &= \frac{\partial}{\partial\sigma}\log P(Y_L|X, \sigma) + \frac{\partial}{\partial\sigma}\log P(\sigma) \\ \frac{\partial}{\partial\sigma}\log P(Y_L|X, \sigma) &= \left\langle\frac{\partial}{\partial\sigma}\log P(Y|X, \sigma)\right\rangle_c - \left\langle\frac{\partial}{\partial\sigma}\log P(Y|X, \sigma)\right\rangle_u\end{aligned}$$

Difference between clamped and unclamped expectations.

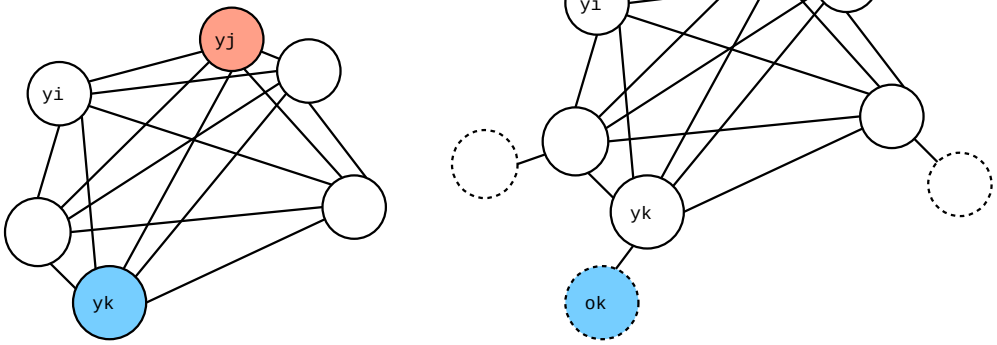
Computing Expectations

Note $\langle f(Y) \rangle_c = \sum_{Y_U} f(Y) P(Y_U | Y_L, X, \sigma)$ and $\langle f(Y) \rangle_u = \sum_Y f(Y) P(Y | X, \sigma)$.

Many methods exist for computing or approximating these expectations.

- Enumeration
- Gibbs Sampling
- Gibbs & Metropolis Sampling
- Swendsen Wang Sampling
- Mean Field
- Structured Variational Approximations
- Loopy Belief Propagation (Bethe/Kikuchi)
- Convex Combinations of Trees

Extended Model



We define a richer model $p(O, Y|X, \Theta)$ where,
Y are the true (hidden) labels;
O are the noisy observed labels;
Includes three terms: "separation", "assignment", and class priors:

$$\frac{1}{\mathcal{Z}} \exp\left(\sum_{i=1}^{l+u} \sum_{j<i} \delta(y_i, y_j) A_{ij} + \sum_{i=1}^{l+u} \sum_{c=1}^{C} \sum_{c'=1}^{C} \delta(o_i, c)\delta(y_i, c') e^{\gamma_{cc'}} + \sum_{i=1}^{l+u} \sum_{c=1}^{C} \delta(y_i, c)\beta_c \right)$$

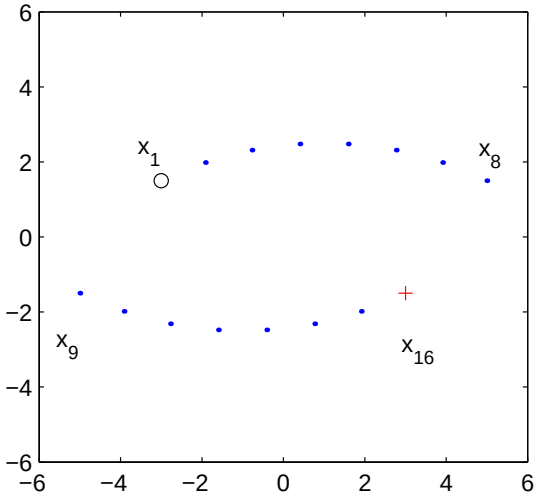
β_c defines class priors. $\gamma_{cc'}$ defines a cost matrix for mislabeling a class c' as c.

Affinity matrix is parameterized: $$A_{ij} = \exp\left\{ \alpha_0 - \sum_{d=1}^{D} \frac{(x_i^d - x_j^d)^2}{2\alpha_d{}^2} \right\}$$

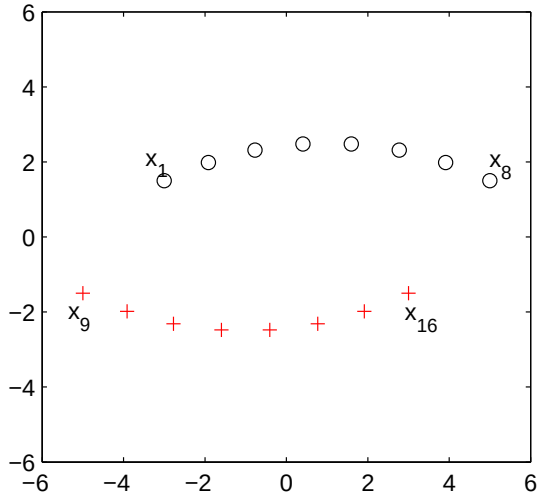
$\alpha = \{\alpha_0, \ldots, \alpha_D\}$ are analogous to the *length scales* in a Gaussian process.

The parameters of this MRF are $\Theta = \{\alpha, \beta, \gamma\}$.

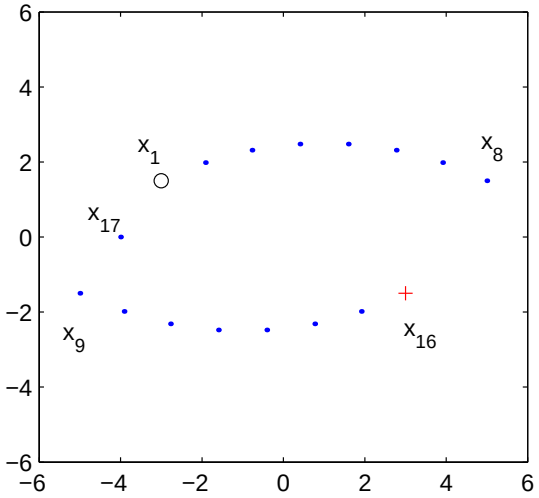
Double Arcs Datasets



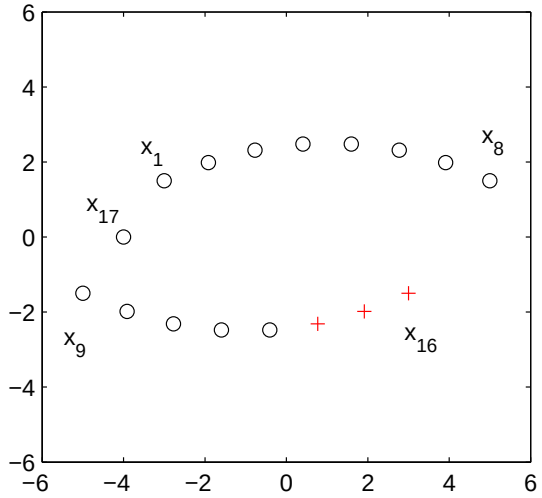
(a) Observed Labels o_i



(b) Computed Labels y_i



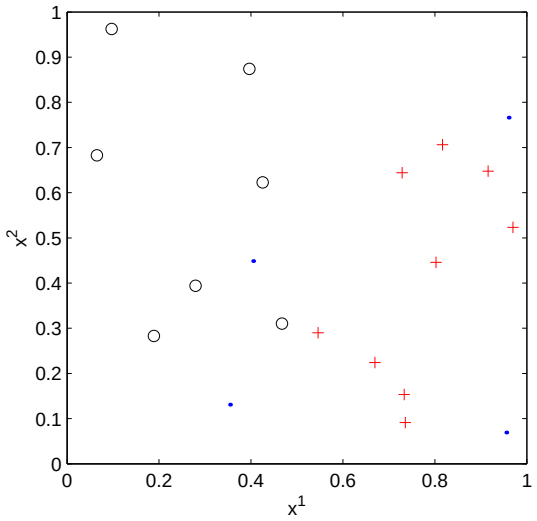
(a) Observed Labels o_i



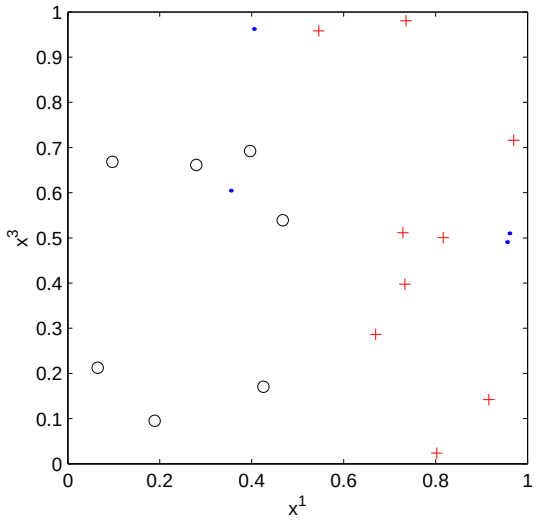
(b) Computed Labels y_i

Note: Max likelihood Θ disconnects all points if only one point per class.

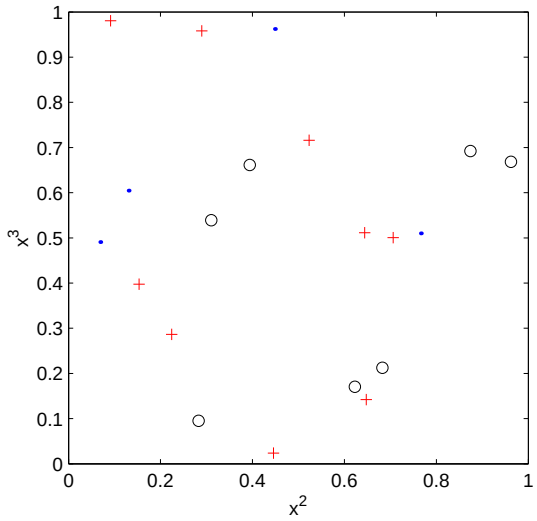
Cube with Irrelevant Dimensions



x^1 vs. x^2



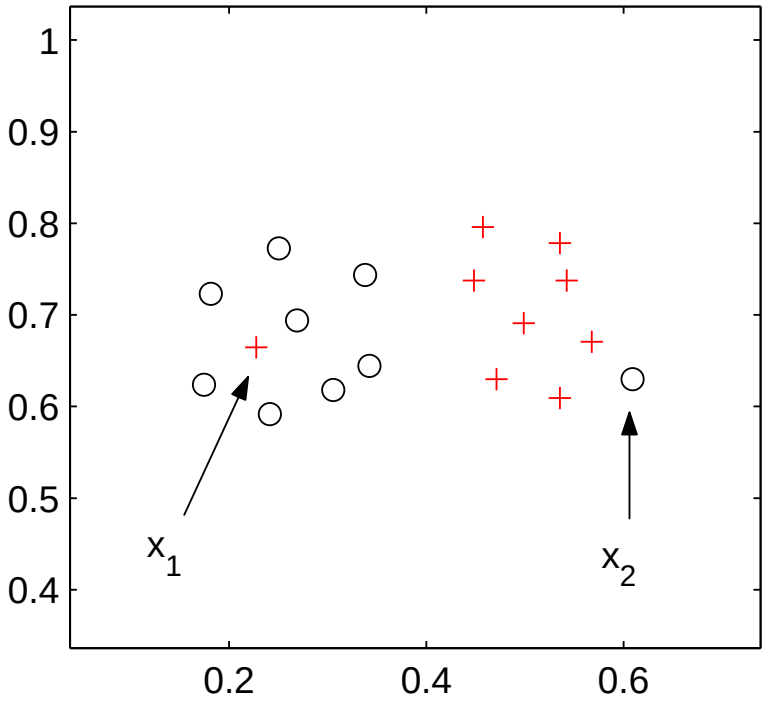
x^1 vs. x^3



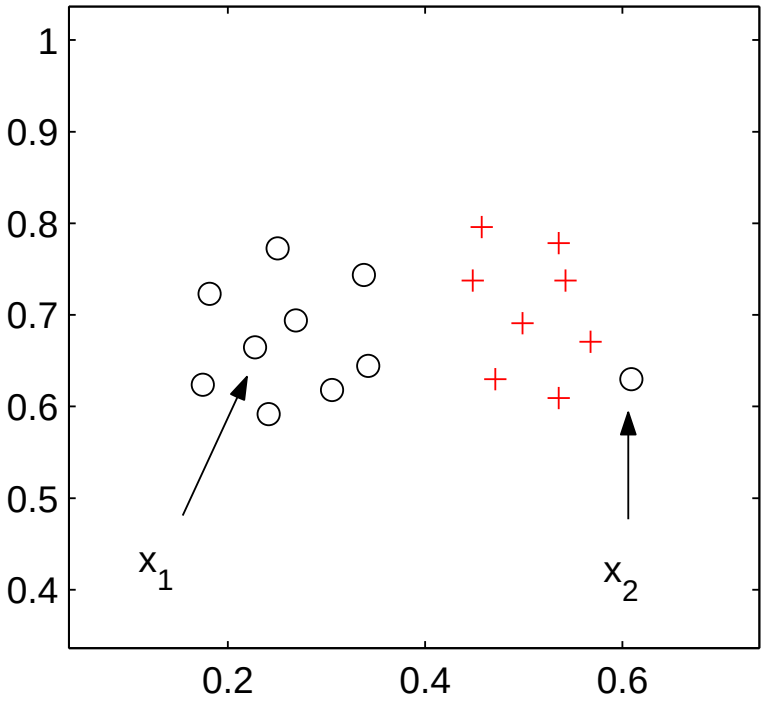
x^2 vs. x^3

Only the first dimension x^1 is relevant to classification.

Noisy Labels



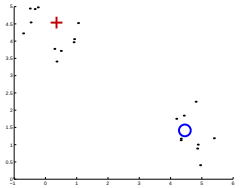
(a) Observed Labels o_i



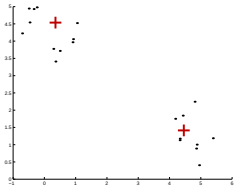
(b) Computed Labels y_i

The Noisy Labels Dataset. x_1 and x_2 are intentionally mislabeled in (a). The trained MRF (b) corrects x_1's, but not x_2's, label.

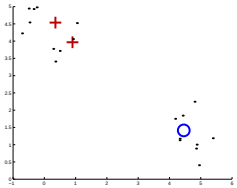
Several Synthetic Datasets



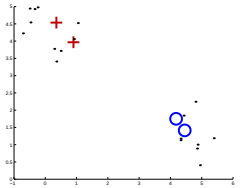
2c2l



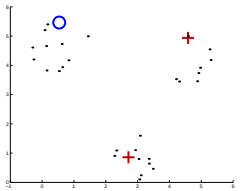
2c2ls



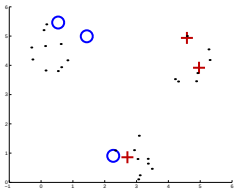
2c3l



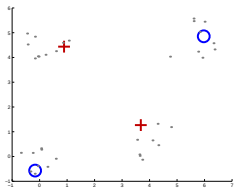
2c4l



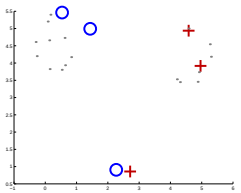
3c3l



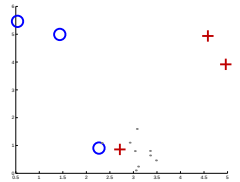
3c6l



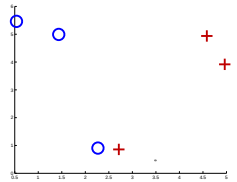
4c4l



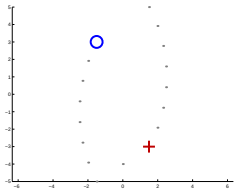
6lb



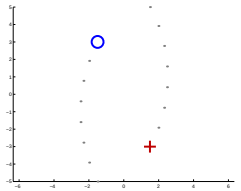
6lc



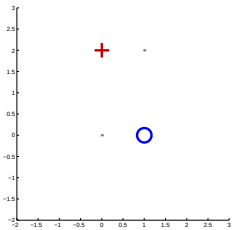
6ld



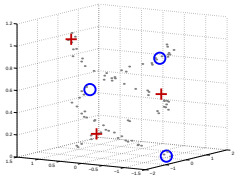
connected arcs



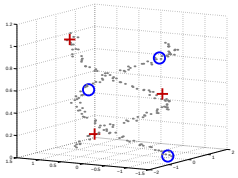
double arcs



rectangle



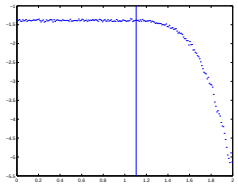
springs1



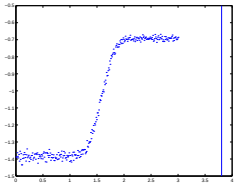
springs2

Several synthetic datasets. Large symbols are labeled data, dots are unlabeled.

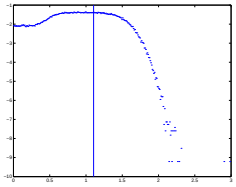
Log Likelihood for Synthetic Datasets



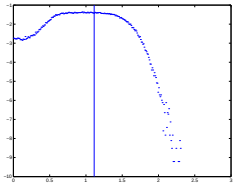
2c2l



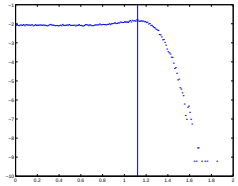
2c2ls



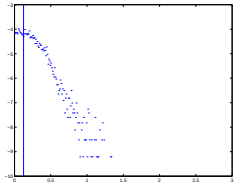
2c3l



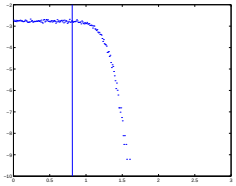
2c4l



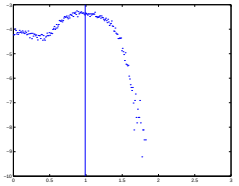
3c3l



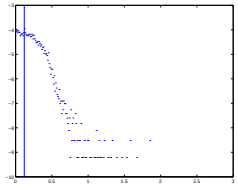
3c6l



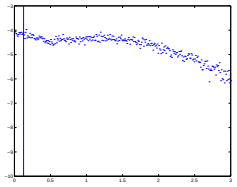
4c4l



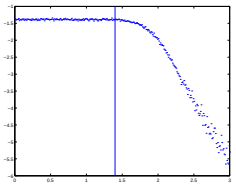
6lb



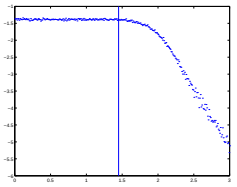
6lc



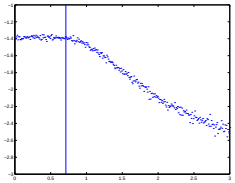
6ld



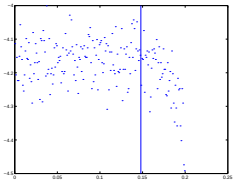
connected arcs



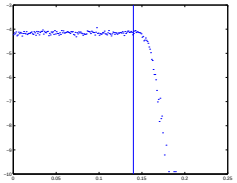
double arcs



rectangle



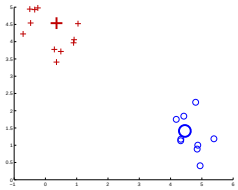
springs1



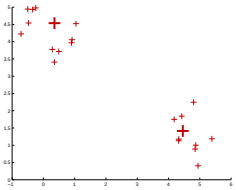
springs2

Log likelihood of labeled data (y-axis) vs. σ (x-axis) for the synthetic datasets. The vertical line marks the σ learned with gradient ascent after we add a prior.

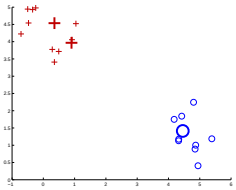
Results for Synthetic Datasets



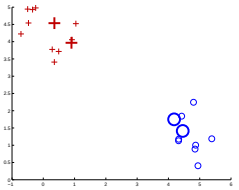
2c2l



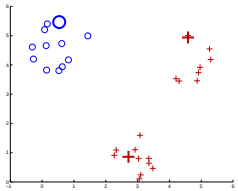
2c2ls



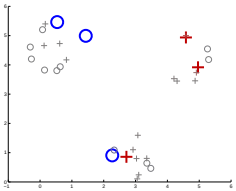
2c3l



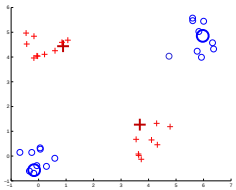
2c4l



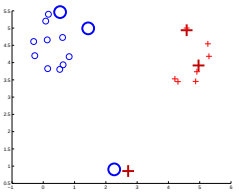
3c3l



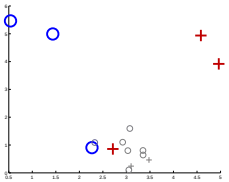
3c6l



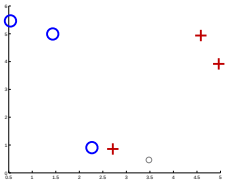
4c4l



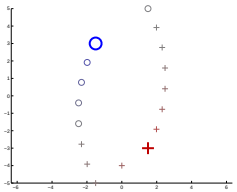
6lb



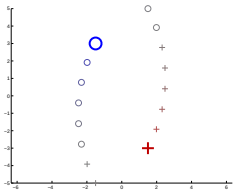
6lc



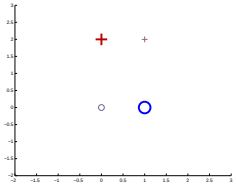
6ld



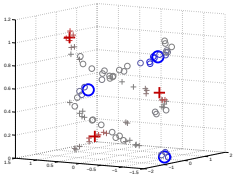
connected arcs



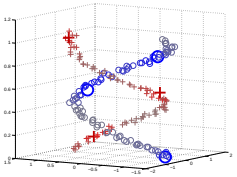
double arcs



rectangle



springs1



springs2

The labels of the synthetic datasets, under the learned σ with a prior.

Summary of MRF for labeled/unlabeled classification

Advantages:

- sound probabilistic framework
- interpretable hyperparameters

Disadvantages:

- The likelihood of the observed data is **not** a sensible objective function for classification with very few labeled points. We get unintuitive classifications.
- Exact and sampling methods are **slow**
- Sensitive to initial conditions; local minima.
- Inherently transductive: optimal Θ only optimal for some data set size.

More details in: *Zhu, X. and Ghahramani, Z. (2002) Towards Semi-supervised Classification with Markov Random Fields. Technical Report CMU-CALD-02-106. Carnegie Mellon.*

Part II: Bayesian Learning of Undirected Graphical Models

In an Undirected Graphical Model (or Markov Network), the joint probability over all variables can be written in a factored form:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_j g_j(\mathbf{x}_{C_j})$$

where $\mathbf{x} = [x_1, \ldots, x_K]$, and

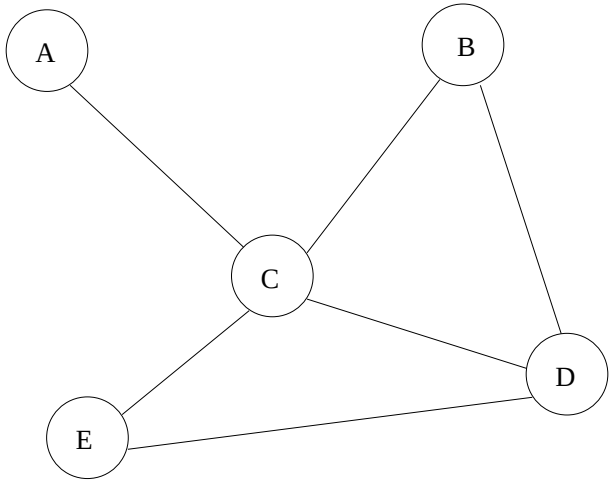
$$C_j \subseteq \{1, \ldots, K\}$$

are subsets of the set of all variables, and $\mathbf{x}_S \equiv [x_k : k \in S]$.

This type of probabilistic model can be represented graphically.

Graph Definition: Let each variable be a node. Connect nodes i and k if there exists a set C_j such that both $i \in C_j$ and $k \in C_j$. These sets form the *cliques* of the graph (fully connected subgraphs).

Undirected Graphical Models: An Example



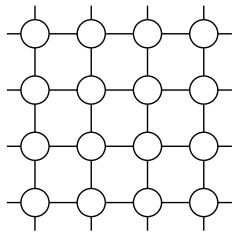
$$P(A, B, C, D, E) = \frac{1}{Z} g(A, C) g(B, C, D) g(C, D, E)$$

Semantics: Every node is conditionally independent from its non-neighbors given its neighbors.

Conditional Independence: $X \perp\!\!\!\perp Y | V \Leftrightarrow p(X|Y, V) = p(X|V)$ for $p(Y, V) > 0$ also $X \perp\!\!\!\perp Y | V \Leftrightarrow p(X, Y|V) = p(X|V)p(Y|V)$.

Examples of Undirected Graphical Models

- Markov Random Fields



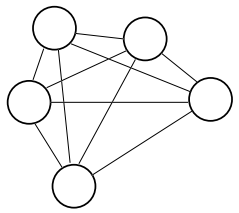
- Exponential Language Models

$$p(s) = \frac{1}{Z} p_0(s) \exp\left\{ \sum_i \lambda_i f_i(s) \right\}$$

- Products of Experts

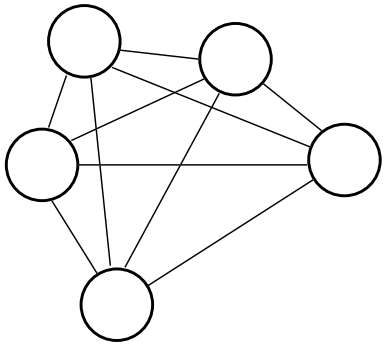
$$p(\mathbf{x}) = \frac{1}{Z} \prod_j p_j(\mathbf{x}|\theta_j)$$

⋆ Boltzmann Machines



Boltzmann Machines

Undirected graph over a vector of binary variables $s_i \in \{0, 1\}$. Variables can be hidden or visible (observed).



$$P(\mathbf{s}|W) = \frac{1}{Z} \exp\left\{ \sum_{j<i} W_{ij} s_i s_j \right\}$$

where Z is the *partition function* (normalizer)

Maximum Likelihood Learning Algorithm: a gradient version of EM

- **E step** involves computing averages w.r.t. $P(\mathbf{s}_H|\mathbf{s}_V, W)$ ("clamped phase"). This could be done via a propagation algorithm or (more usually) an approximate method such as Gibbs sampling.
- **M step** requires gradients w.r.t. Z, which can be computed by averages w.r.t. $P(\mathbf{s}|W)$ ("unclamped phase").

Hebbian and anti-Hebbian rule: $\Delta W_{ij} = \eta[\langle s_i s_j \rangle_c - \langle s_i s_j \rangle_u]$

Why Bayesian Learning?

- Useful prior knowledge can be included (e.g. sparsity)
- Avoids overfitting
- Error bars
- Model selection

A Simple Idea

Define the following **joint distribution** of weights W and matrix of binary variables S, organized into N rows (data vectors) and M columns (features, variables). Some variables on some data points may be hidden and some may be observed.

$$p(S, W) = \frac{1}{Z} \exp \left\{ -\frac{1}{2\sigma^2} \sum_{i,j=1}^{M} W_{ij}^2 + \sum_{n=1}^{N} \sum_{j<i}^{M} W_{ij} s_{ni} s_{nj} \right\}$$

Where $Z = \int dW \sum_S \exp\{\ldots\}$ is a nasty partition function.

Gibbs sampling in this model is very easy!

- Gibbs sample s_{ni} given all other $\mathbf{s}$ and W: Bernouilli, easy as usual.
- Gibbs sample W given s: diagonal multivariate Gaussian, easy as well.

What is wrong with this approach?

...a Strange Prior on W

$$p(S,W) = \frac{1}{Z}\exp\left\{-\frac{1}{2\sigma^2}\sum_{i,j=1}^{M} W_{ij}^2 + \sum_{n=1}^{N}\sum_{j<i}^{M} W_{ij}s_{ni}s_{nj}\right\}$$

This defines a Boltzmann machine for the data given W, but defines a **somewhat strange** and hard to compute "prior" on the weights.
What is the prior on W?

$$p(W) = \sum_S p(S,W) \propto N(0,\sigma^2 I)\sum_S \exp\left\{\sum_{n,j<i} W_{ij}s_{ni}s_{nj}\right\}$$

where the second factor is data-size dependent, so it's not a valid hierarchical Bayesian model of the kind $W \to S$. The second factor can be written as:

$$\sum_S \exp\left\{\sum_{n,j<i} W_{ij}s_{ni}s_{nj}\right\} = \left(\sum_s \exp\left\{\sum_{j<i} W_{ij}s_i s_j\right\}\right)^N = Z(W)^N$$

This will not work!

Estimating $Z(W)$

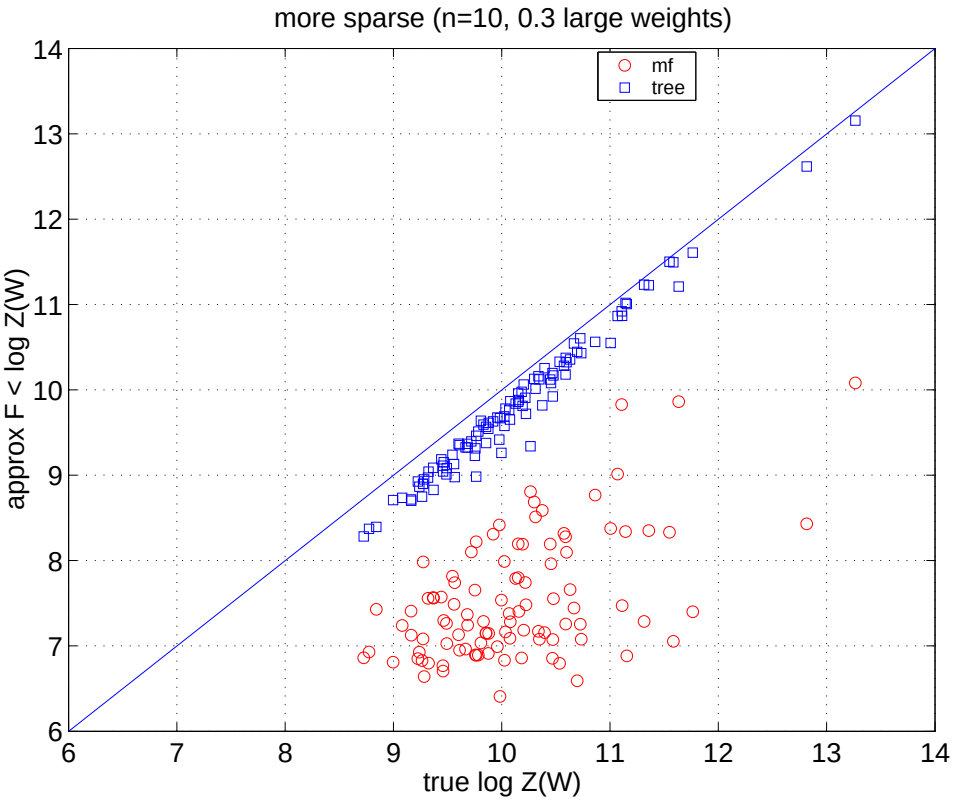
To define a valid prior we need to compute $Z(W) \equiv \sum_s \exp\left\{\sum_{j<i} W_{ij} s_i s_j\right\}$.

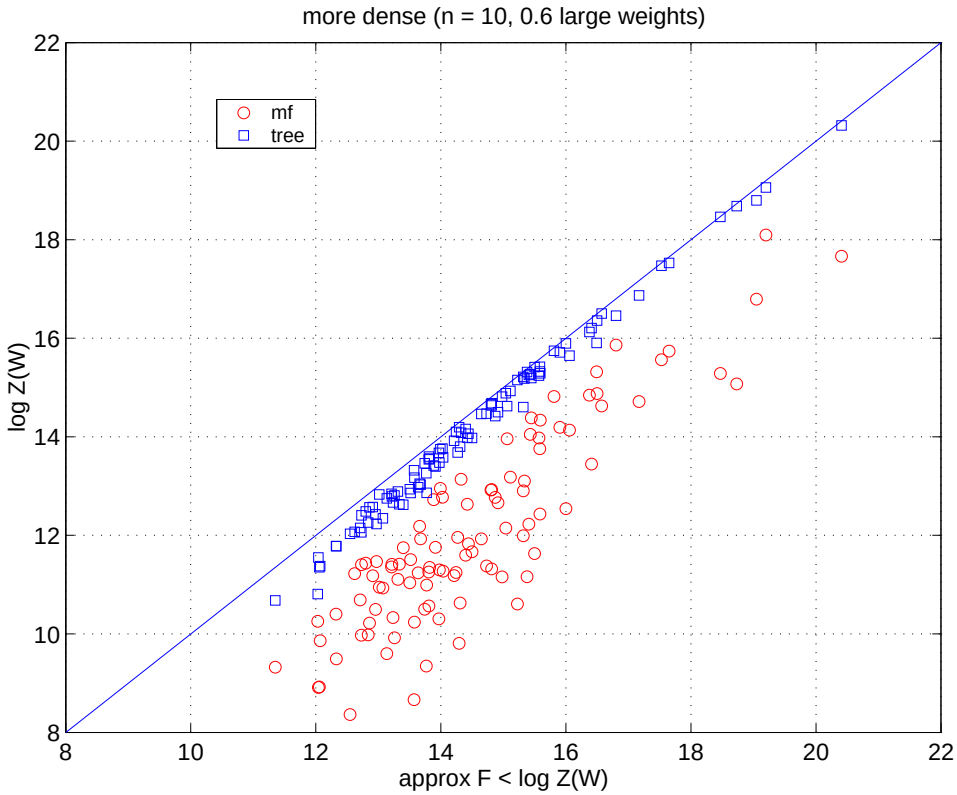
$$p(W,S) = p(W)P(S|W) = P(W)\left[\frac{1}{Z(W)^N}\exp\left\{\sum_{n=1}^{N}\sum_{j<i} W_{ij} s_{ni} s_{nj}\right\}\right]$$

- Mean-field $\ln Z(W) \geq \sum_s q(s) \sum_{j<i} W_{ij} s_i s_j + H(q)$

  where $q(s) = \prod_i m_i^{s_i}(1-m_i)^{(1-s_i)}$ and H is entropy.
- Tree-based variational approximation. Use a spanning tree for $q(s)$.
- Bethe free energy. Approximate $Z(W)$ by running loopy propagation and computing Bethe free energy.
- Annealed Importance Sampling: $\frac{Z(W)}{Z(0)} = \frac{Z(\alpha_1 W)}{Z(0)}\frac{Z(\alpha_2 W)}{Z(\alpha_1 W)} \cdots \frac{Z(\alpha_t W)}{Z(\alpha_{t-1} W)}$ where $0 \leq \alpha_1 \ldots \leq \alpha_t = 1$.
- Contrastive Sampling: Brief Gibbs sampling starting at data to compute $Z(W)/Z(W')$

Mean Field vs Tree Variational Approximation





The tree based approximation found an MST and then used Wiegerinck's (UAI, 2000) variational approximation.

Metropolis with Importance Sampling Inner Loop

- Sample S given W.
- Sample W given S:
  - Propose W' given W from some symmetric proposal distribution
  - Compute acceptance probability $a(W'|W) = \min(1, \frac{p(W')}{p(W)} \frac{p(S|W')}{p(S|W)})$ where the second term is:

$$\frac{p(S|W')}{p(S|W)} = \exp\left\{ \sum_{n,j<i} (W'_{ij} - W_{ij}) s_{ni} s_{nj} \right\} \left(\frac{Z(W)}{Z(W')} \right)^N$$

  and the last term is estimated as below.

$$\frac{Z(W)}{Z(W')} = \frac{\sum_s \exp\left\{ \sum_{j<i} W_{ij} s_i s_j \right\}}{\sum_s \exp\left\{ \sum_{j<i} W'_{ij} s_i s_j \right\}} = \left\langle \exp\left\{ \sum_{j<i} (W_{ij} - W'_{ij}) s_i s_j \right\} \right\rangle_{p(s|W')}$$

Contrastive Sampling to Estimate $Z(W)$

We wish to compute:

$$\frac{Z(W)}{Z(W')} = \frac{\sum_s \exp\left\{\sum_{j<i} W_{ij} s_i s_j\right\}}{\sum_s \exp\left\{\sum_{j<i} W'_{ij} s_i s_j\right\}}$$

- Start from data set
- Create a "corrupted" data set, for example, by Gibbs sampling briefly starting from data using W (or W', or many Ws)
- Define $q(s)$ to be the empirical distribution of this corrupted data
- Then use:

$$\frac{Z(W)}{Z(W')} \approx \frac{\sum_s q(s) \exp\left\{\sum_{j<i} W_{ij} s_i s_j\right\}}{\sum_s q(s) \exp\left\{\sum_{j<i} W'_{ij} s_i s_j\right\}}$$

* A pseudo-Bayesian posterior can be computed by fixing $q(s)$. This concentrates "posterior" mass on weights that are good at explaining the real data and bad at explaining the corrupted data.

* We get a nice bias/variance trade-off and a huge computational savings!

Part II: Summary and Future Directions

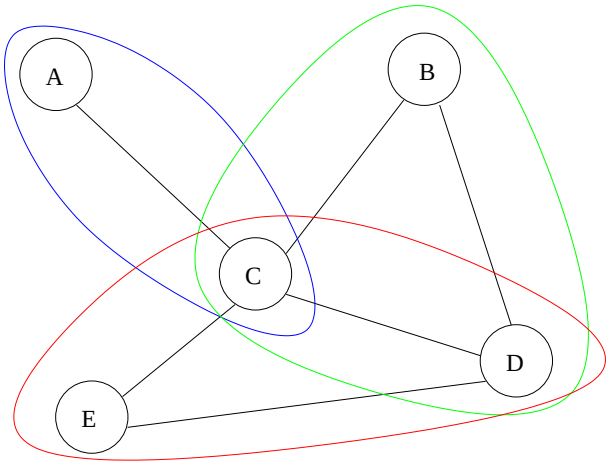
- Tree-based approximations seem to work well, but exponentiating amplifies any errors.
- We are comparing these methods (and hopefully others) on small BMs where the exact solution can be computed.
- There is much future work in this area!

`http://www.gatsby.ucl.ac.uk/~zoubin`
`http://www.variational-bayes.org/`

Appendix

Clique Potentials and Markov Networks

Definition: a *clique* is a fully connected subgraph (usually maximal).
C_i will denote the set of variables in the i^{th} clique.



1. Identify cliques of graph G
2. For each clique C_i assign a non-negative function $g_i(C_i)$ which measures "compatibility".
3. $p(X_1, \ldots, X_n) = \frac{1}{Z} \prod_i g_i(C_i)$ where $Z = \sum_{X_1 \cdots X_n} \prod_i g_i(C_i)$ is normalizer

The graph G embodies the conditional independencies in p (i.e. G is a Markov Field relative to p): If V lies in *all* paths between X and Y in G, then $X \perp\!\!\!\perp Y | V$.

Hammersley–Clifford Theorem (1971)

Theorem: A probability function p formed by a normalized product of positive functions on cliques of G is a Markov Field relative to G.

Definition: The graph G is a *Markov Field relative to* p if it does not imply any conditional independence relationships that are not true in p.
(We are usually interested in the minimal such graph.)

Proof: We need to show that the neighbors of X, $\mathrm{ne}(X)$ are a Markov Blanket for X:

$$\begin{aligned} p(X, Y, \ldots) =& \frac{1}{Z} \prod_i g_i(C_i) = \frac{1}{Z} \prod_{i: X \in C_i} g_i(C_i) \prod_{j: X \notin C_j} g_j(C_j) \\ =& \frac{1}{Z} f_1\big(X, \mathrm{ne}(X)\big)\ f_2\big(\mathrm{ne}(X), Y\big) = \frac{1}{Z'} p(X|\,\mathrm{ne}(X))\ p(Y|\,\mathrm{ne}(X)) \end{aligned}$$

This shows that: $p(X, Y|\,\mathrm{ne}(X)) = p(X|\,\mathrm{ne}(X))\ p(Y|\,\mathrm{ne}(X)) \Leftrightarrow X \perp\!\!\!\perp Y|\,\mathrm{ne}(X)$.