# Learning with Dependencies between Several Response Variables:

## From Hierarchical Bayes and Multitask Learning to Structured Output Prediction and Relational Learning

Volker Tresp

Siemens

Corporate Technology

Kai Yu

NEC Labs America

## Multiple Outputs

“Multiple outputs do not affect each others least squares estimates”

*Hastie, Tibshirani, Friedman (2001)*

*We will study cases, where this statement is not applicable!*

## Overview

Hierarchical Bayes, inductive transfer learning, multi-label prediction, multitask learning, random-effects models, random parameter models, mixed models, mixed effect models, nested models, multilevel models, hierarchical linear models, generalized mixed models, collaborative filtering, canonical correlation analysis, maximal covariance regression, partial least squares, multivariate regression, structured output prediction (and probably many more things I am not even aware of)

- An attempt to provide a view
- With a Bayesian flavor but not strictly Bayes

I. Hierarchical Bayes - Mixed Models
  - A: Problem Settings and Simple Solutions
  - B: Hierarchical Bayes - Mixed Models
  - C: Nonparametric Hierarchical Bayes

II. Projection Methods

III. Multivariate Models and Structured Outputs

IV. Link Prediction / Relationship Prediction

# A Classical Generic Supervised Learning Task

Data matrix

$$\begin{pmatrix} \mathbf{x}_1 & y_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \mathbf{x}_N & y_N \end{pmatrix}$$

$$y_i = f_{\mathbf{w}}(\mathbf{x}_i) + \varepsilon_i$$

$$P(y_i = 1) = \sigma\left( f_{\mathbf{w}}(\mathbf{x}_i)\right)$$

- Rows: data points
- Columns:
  - Input vector ***x***
  - Output scalar *y*

## A New Generic Supervised Learning Problem?

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & & y_{1,M} \\ \cdot & & & \\ \cdot & & & \\ \mathbf{x}_N & y_{N,1} & & y_{N,M} \end{pmatrix}$$

- Rows: data points
- Columns:
  - Input vector ***x***
  - Output **vector** ***y***

Perspective of the presentation:

- This is the data, what should one do?

# I: Hierarchical Bayes - Mixed models
# II. Projection approaches

$$\begin{pmatrix} \boxed{\begin{matrix} \mathbf{x}_1 \\ \cdot \\ \cdot \\ \mathbf{x}_N \end{matrix}} & \boxed{\begin{matrix} y_{1,1} & & y_{1,M} \\ & & \\ & & \\ y_{N,1} & & y_{N,M} \end{matrix}} \end{pmatrix}$$

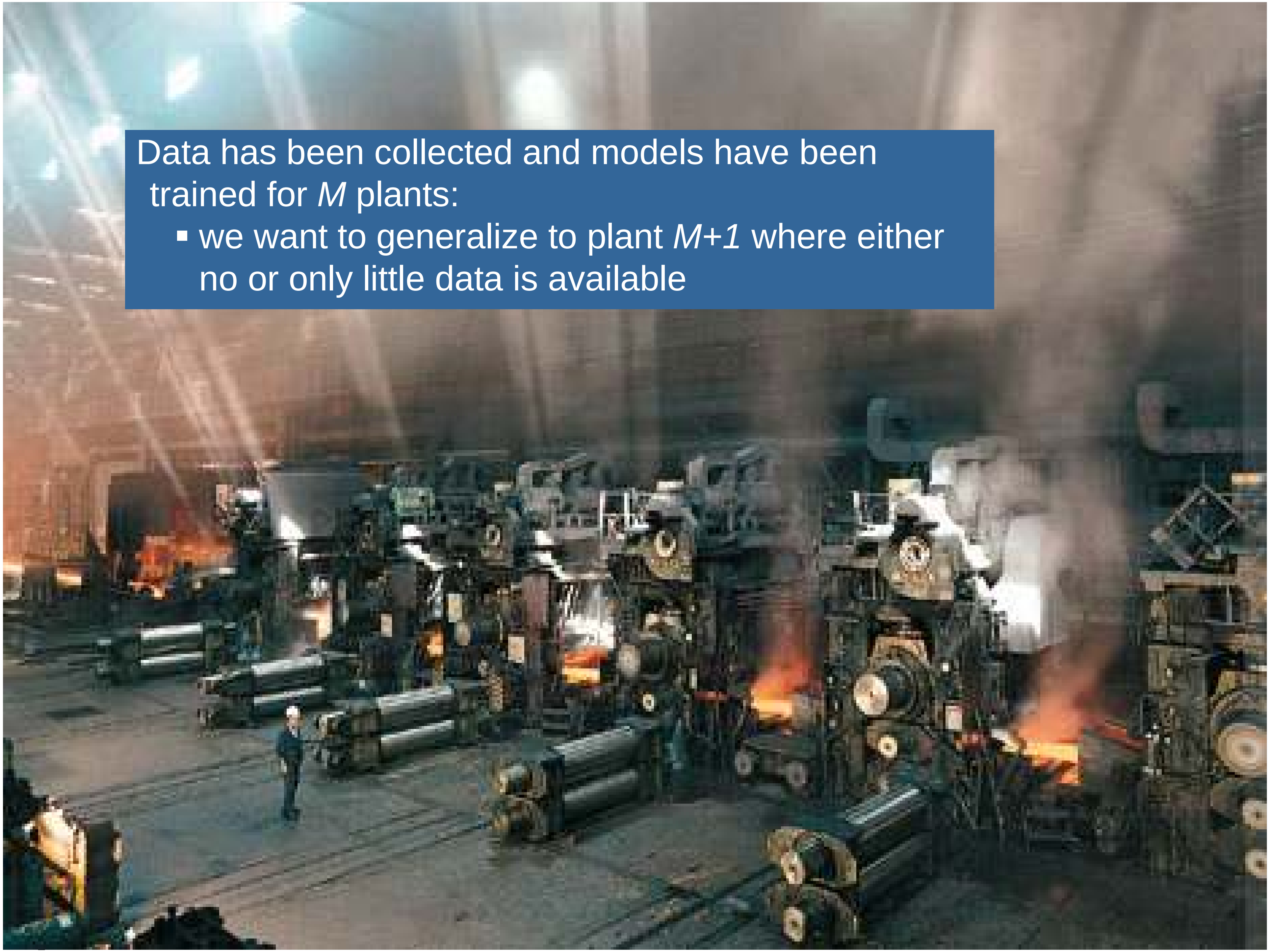
$$P(y_{i,j} = 1) = \sigma(f_{\mathbf{w}_j}(\mathbf{x}_i))$$

$\mathbf{w}_j$ implicitly depends on all $\{y_{i,j}\}$

- Each output dimension (situation) is trained *independently* given a transformation (resp. given a prior distribution) that is found using *all data*
- *Hierarchical Bayes: statistical strength between multiple outputs is shared by common parameters in the prior distribution*
- *Projection Methods: The input is mapped to a lower-dimensional space that was found using all data*
- Applicable when one can assume that the functional dependencies for all outputs (situations) come from the same (simple) family of functions

Data has been collected and models have been trained for *M* plants:

- we want to generalize to plant *M+1* where either no or only little data is available



- Data for length of stay prediction has been collected for patients in *M* hospitals:
  - can we generalize to patients in hospital *M+1*

# Length of Stay Prediction

Properties of patient 1

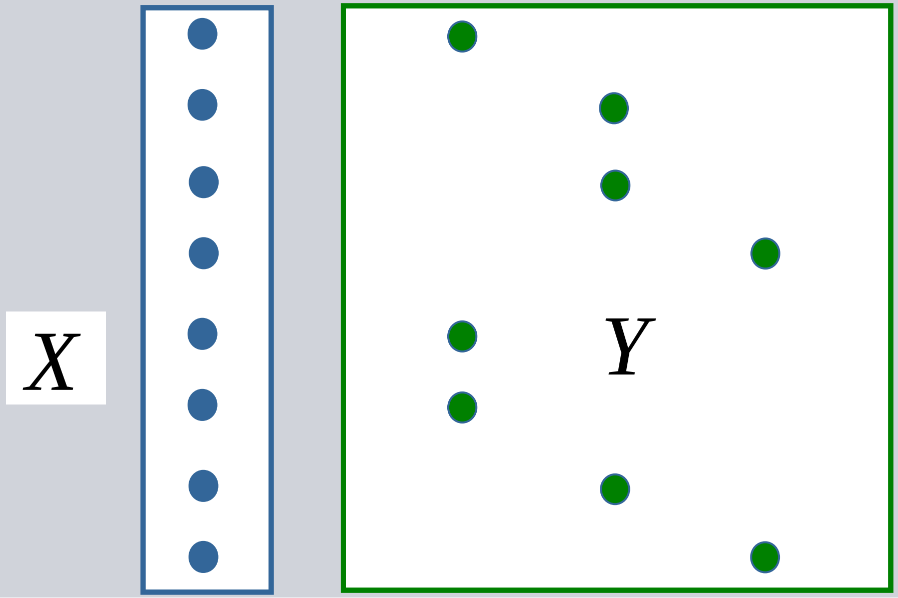
Length of stay prediction for hospital 1

Length of stay prediction for hospital *M*

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & & y_{1,M} \\ \cdot & & & \\ \cdot & & & \\ \mathbf{x}_N & y_{N,1} & & y_{N,M} \end{pmatrix}$$

Properties of patient *N*

- Naturally, a patient has typically been only in one hospital: technically, for each input only one output might be available
- Thus in many Hierarchical Bayesian setting, the algorithms need to be able to deal with (an extreme case of) missing output information



# III. Multivariate Prediction

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & & y_{1,M} \\ \cdot & & & \\ \cdot & & & \\ \mathbf{x}_N & y_{N,1} & & y_{N,M} \end{pmatrix}$$

$$P(y_{i,1},\ldots,y_{i,M} \mid \mathbf{x}_i, \mathbf{w})$$

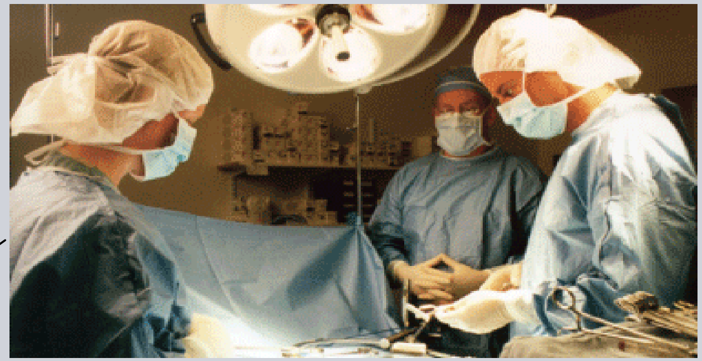
After training

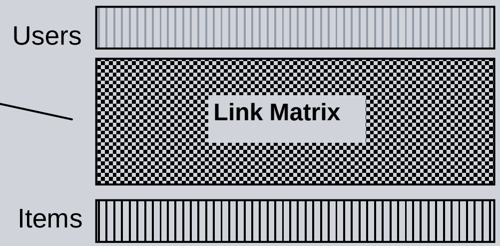
- we obtain one global model
- dependencies between outputs are modeled
- *statistical strength between multiple outputs is shared since parameters are sensitive to all outputs*

# Examples: Multivariate Prediction

For a given object, several output variables / labels are measured: is it easier to predict *M* things than one?

- <u>Decision Support:</u> For a given patient many procedures are possible
- <u>Recommendation:</u> For a given user, many items might be of interest
- <u>Semantic Web:</u> For a given text, many annotations are possible



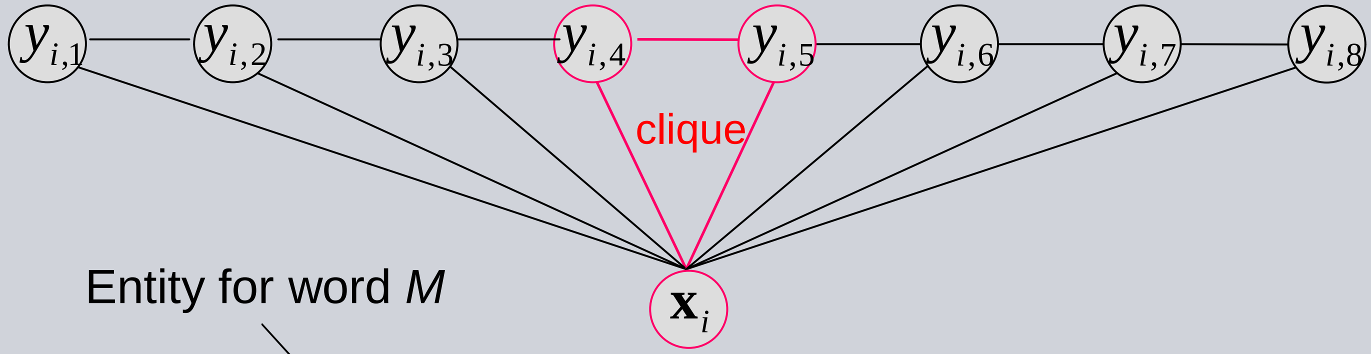


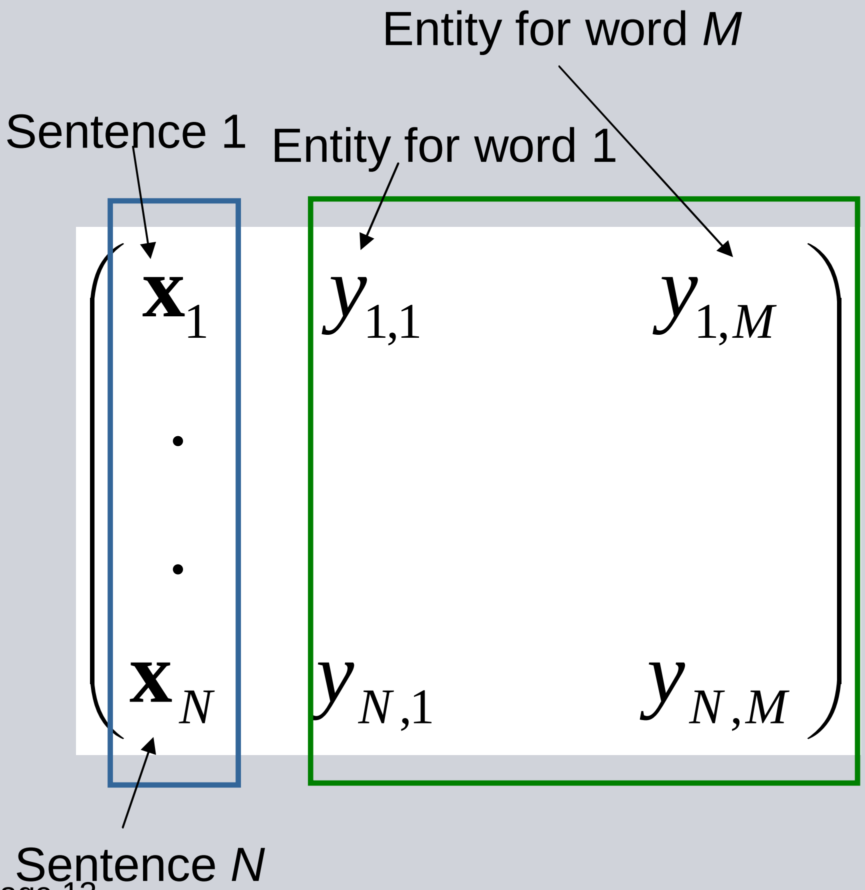

# Structured Output Prediction

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & y_{1,M} \\ \cdot & & \\ \cdot & & \\ \mathbf{x}_N & y_{N,1} & y_{N,M} \end{pmatrix}$$

- Dependency structure between the outputs is known and simplifies model
- Parameter sharing (invariance assumption): data efficiency
- Applicable when structural dependencies between outputs are known

# Example: Named Entity Recognition with Conditional Random Fields

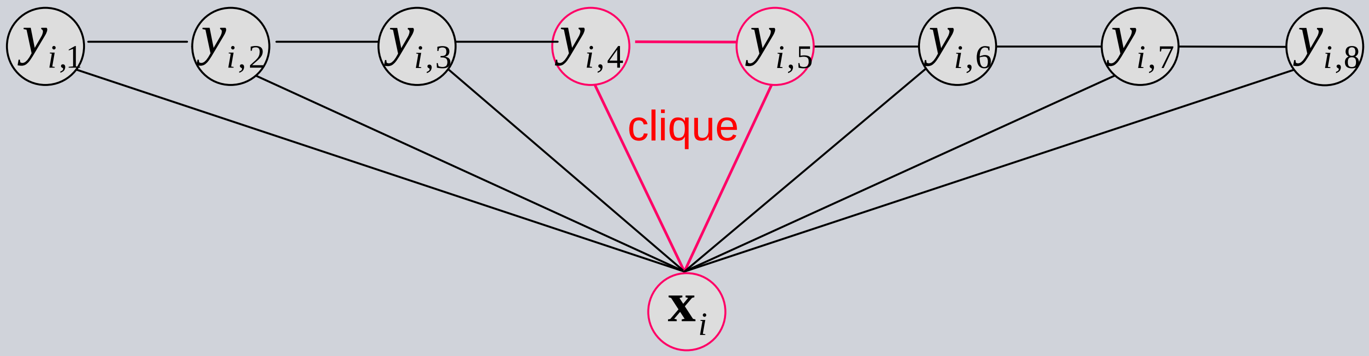



$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_k^{(c)} f_k^{(c)}(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

- Neighborhood structure between outputs
- Note that after training we obtain a probabilistic score for a joint configuration of *x* and *y* so we don't really predict *y* from *x*
- Clique: $x_{i,j}^{(c)}, y_{i,j}, y_{i,j-1}$

## Parameters Sharing




- Often one assumes some invariance, e.g.,

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

- Each clique uses the same feature functions
  - Data efficiency
  - Can handle sequences with varying lengths

# Example: Image Restoration

Image 1

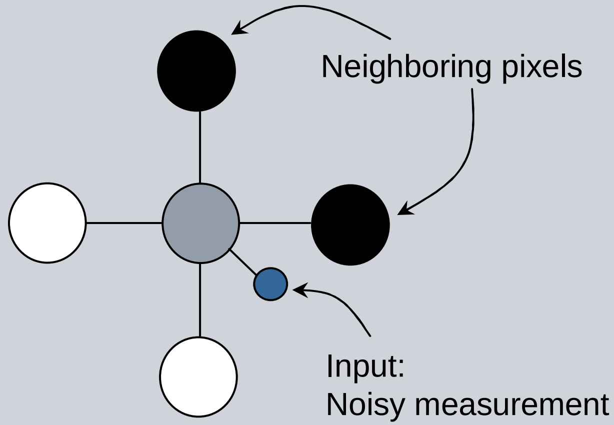
Reconstructed Pixel 1

Reconstructed Pixel M

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & & y_{1,M} \\ \cdot & & & \\ \cdot & & & \\ \mathbf{x}_N & y_{N,1} & & y_{N,M} \end{pmatrix}$$

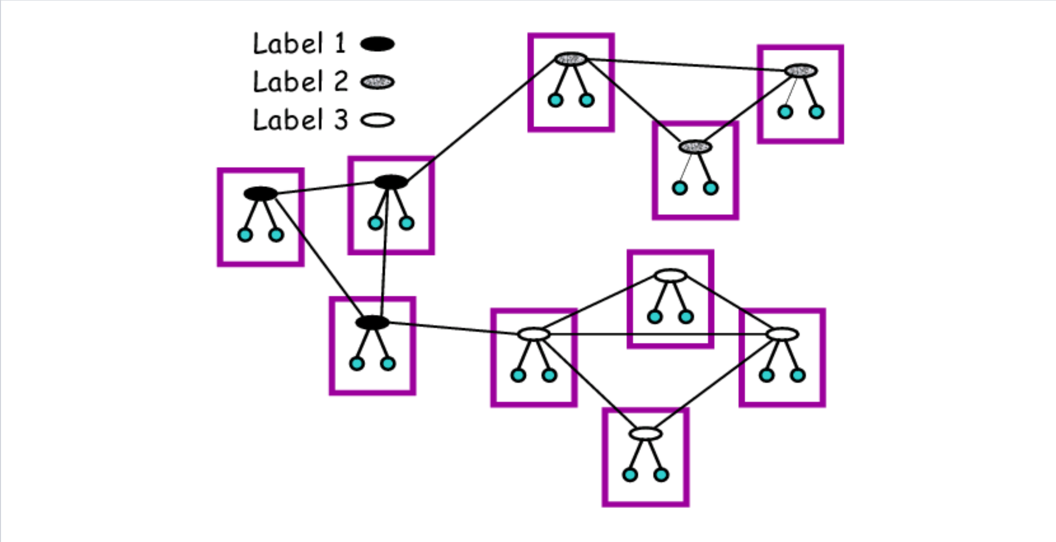
Image *N*

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$



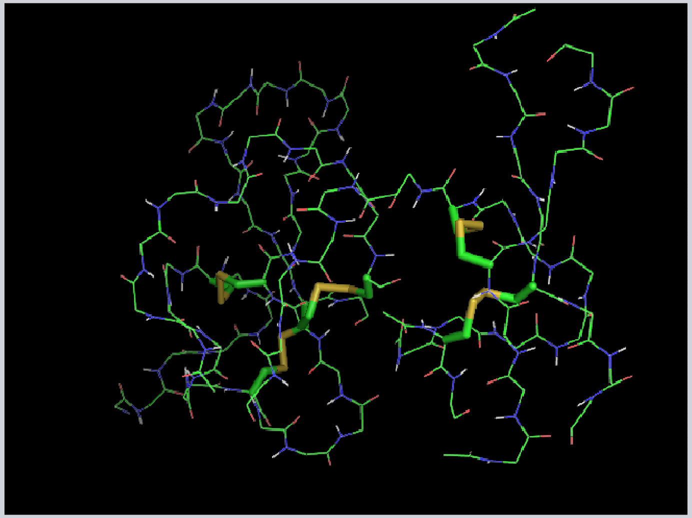
# Example: Social Network Analysis

- Nodes are actors
- Typical task: classification of actors based on actor attributes and based on the class labels of neighboring actors (collective classification)
- New:
  - Often there is only one social network available: then the social network corresponds to only **one row (the network is one data point)** and learning relies on parameter sharing
  - A node has a varying number of neighbors: aggregation



# Protein Structure Prediction

AVITGACERDLQCG
KGTCCAVSLWIKSV
RVCTPVGTSGEDCH
PASHKIPFSGQRMH
HTCPCAPNLACVQT
SPKKFKCLSK



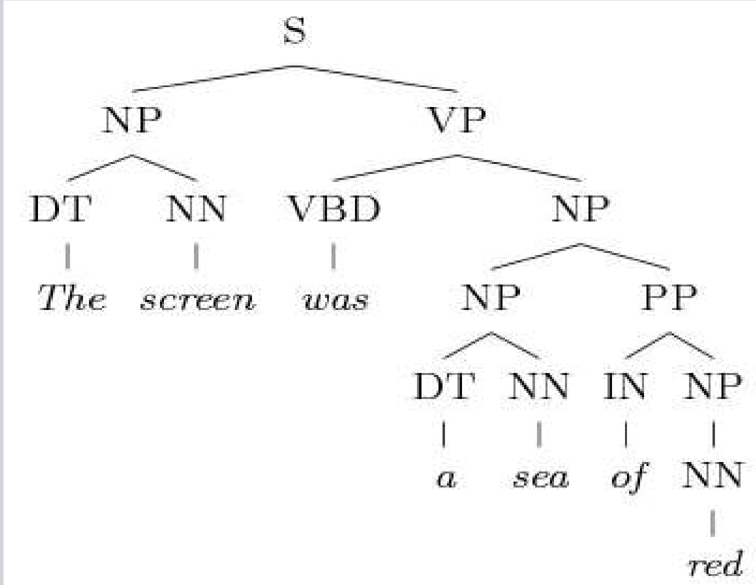
$$\underbrace{\mathbf{x}_i \qquad\qquad y_{i,*}}_{\text{score}}$$

Taskar, Chatalbashev, Koller and Guestrin (2006)

# Natural Language Parsing

- Mapping of sentence to a parse tree
- Features count how many times a weighted grammar rule occurs on valid parse trees

The screen was
a sea of red




$$\underbrace{\mathbf{x}_i \qquad\qquad\qquad\qquad y_{i,*}}_{\text{score}}$$

Taskar, Chatalbashev, Koller and Guestrin (2006)

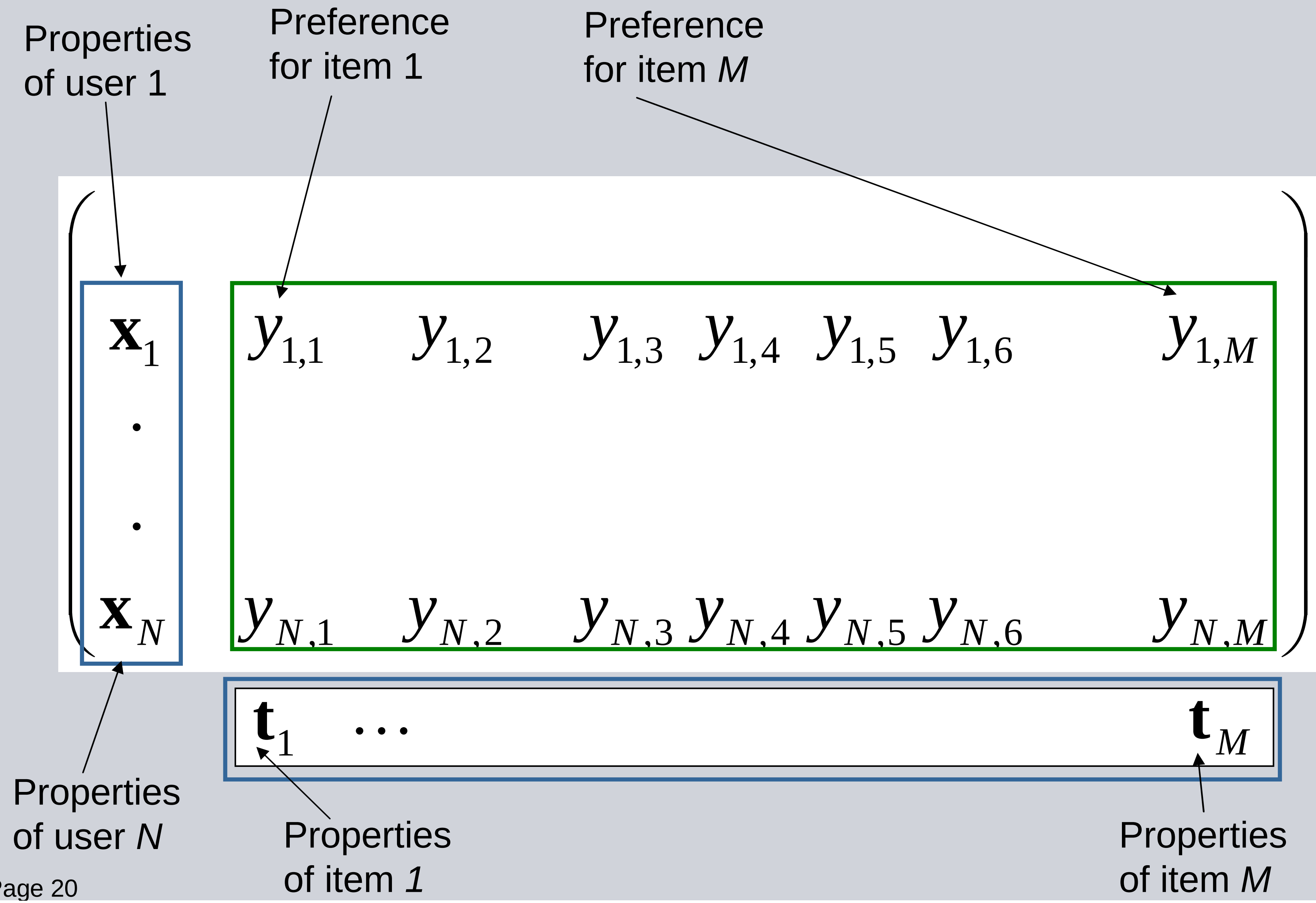
## IV. Link Prediction / Relationship Prediction

- $y_{i,j}$ describes the link (relationship) between row entity *i* and column entity *j*
- Row objects and column objects might of same type or of different types
- Row and column objects both might have attributes

$$
\begin{pmatrix}
\mathbf{x}_1 & y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} & y_{1,5} & y_{1,6} & y_{1,M} \\
\cdot & & & & & & & \\
\cdot & & & & & & & \\
\mathbf{x}_N & y_{N,1} & y_{N,2} & y_{N,3} & y_{N,4} & y_{N,5} & y_{N,6} & y_{N,M}
\end{pmatrix}
$$

$$\mathbf{t}_1 \quad \ldots \quad \mathbf{t}_M$$

# Recommendation System (Bipartite)



## Key Message of the Presentation

- Prediction accuracy is improved in models with several response variables if some or all model parameters are sensitive to all outputs
  - Then, in learning, some or all parameter estimates benefit from the multiple outputs

# I. Hierarchical Bayes

- Predicting the same thing (patient’s length of stay) but in different situations (different hospitals)

# I.A.
# Problem Settings and Simple Solutions

## Problem Setting

- Data is collected for *M* different situations (entities/sites/tasks) and the goal is to learn predictive models $f_j(\mathbf{x}), \quad j = 1, \cdots, M$
- Can data from other situations help to improve the prediction of both $f_j(\mathbf{x})$ and for a new situation $f_{new}(\mathbf{x})$ ?
- For simplicity, we consider models linear in the parameters of the form

$$f_j(\mathbf{x}) = \sum_{l=1}^{L} w_{j,l} \phi_l(\mathbf{x})$$

Typically we only have access to $y_j(\mathbf{x}) = f_j(\mathbf{x}) + \varepsilon_j(\mathbf{x})$

## Simple Solution: One Global Model

$$f(\mathbf{x}) = \sum_{l=1}^{L} w_l \phi_l(\mathbf{x})$$

- We learn one model with all data: Fruits, not apple and oranges
- Data efficient solution

- Problems: ignores differences in different situations

# Simple Solution: Separate Models

- A model for each situation is trained solely on its own data

$$f_j(\mathbf{x}) = \sum_{l=1}^{L} w_{j,l}\phi_l(\mathbf{x})$$

- Problem: no sharing of statistical strength
  (but sometimes the correct solution)
  - Only one output dimension contributes to parameter estimates

## Simple Solution: Situation as Input

- The situation is just another set of inputs to the model, e.g., in form of indicator variables

$$f(\mathbf{x}, \mathbf{u}_j)$$

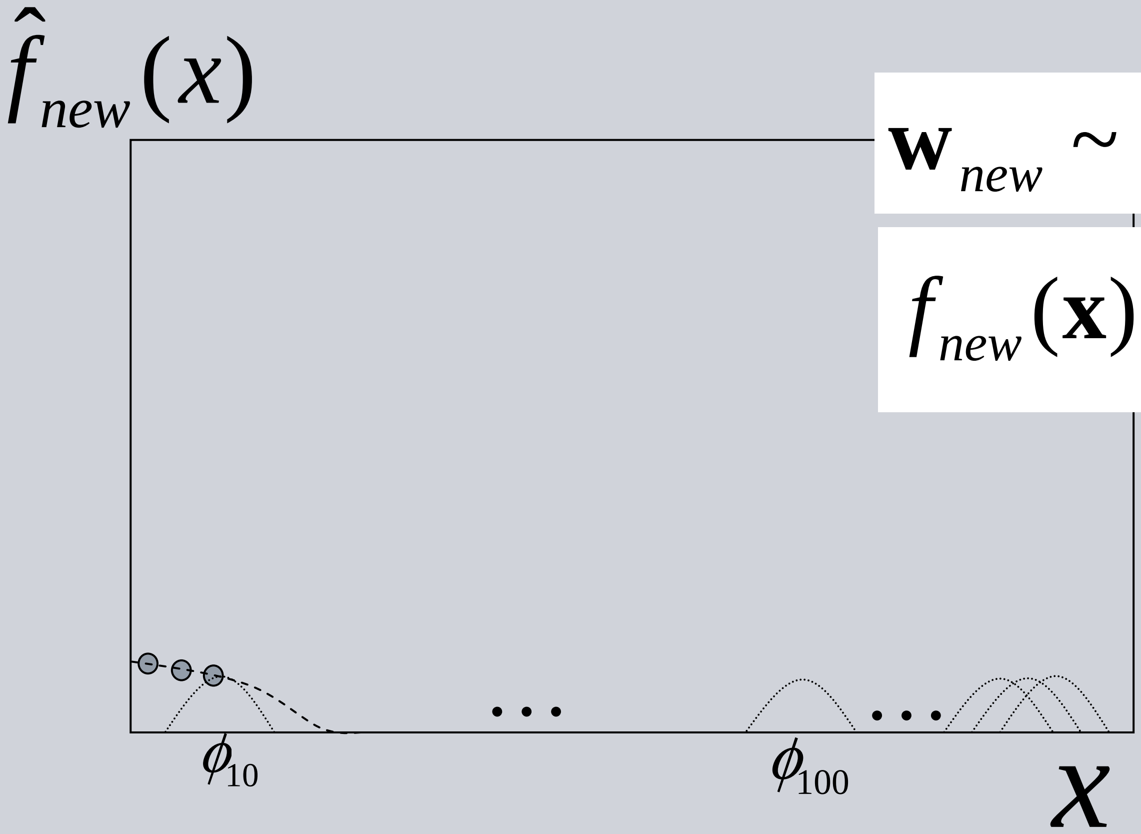
$$\mathbf{u}_j = (0,0,\cdots,u_{j,j} = 1,\cdots 0,0,0)^T$$

- Data efficient
- Problem: sometimes suitable but the influence of the situation might be quite complex

# I.B.
# Hierarchical Bayes / Mixed Models

## New Situation with Few Data Points

- Assume a few data points local in input space

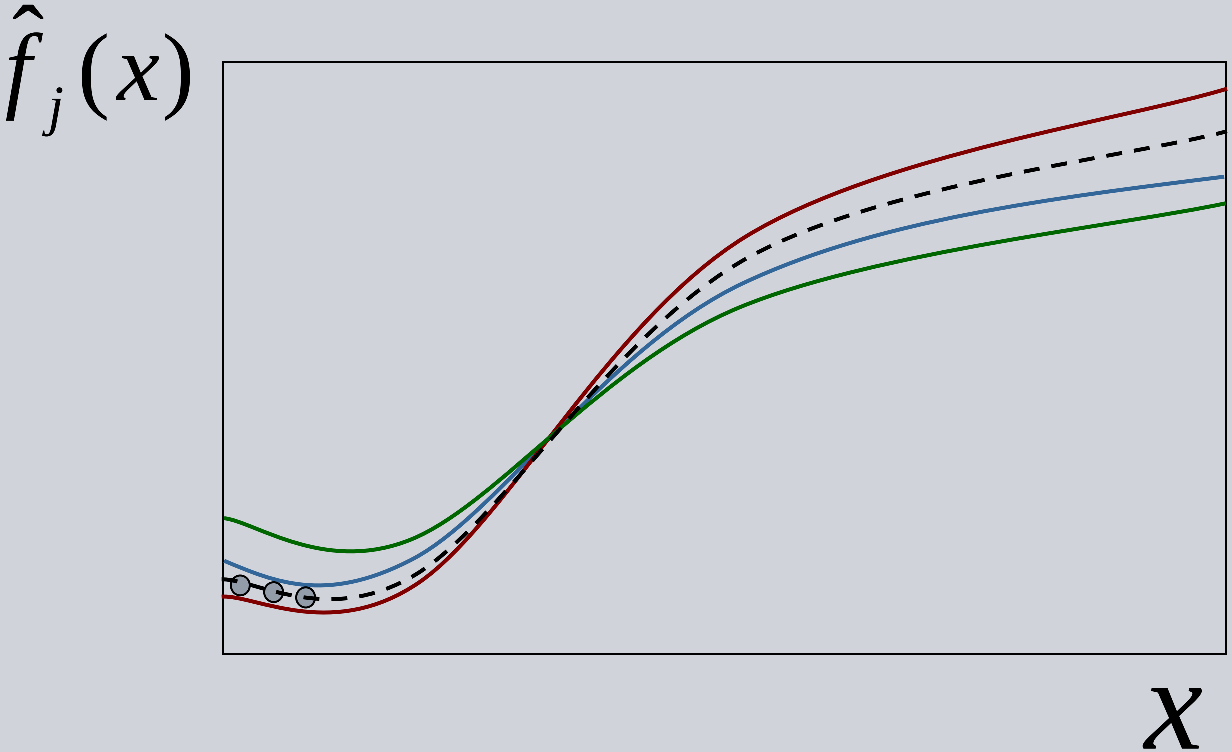



$$\mathbf{w}_{new} \sim N(0, \alpha^2 I)$$

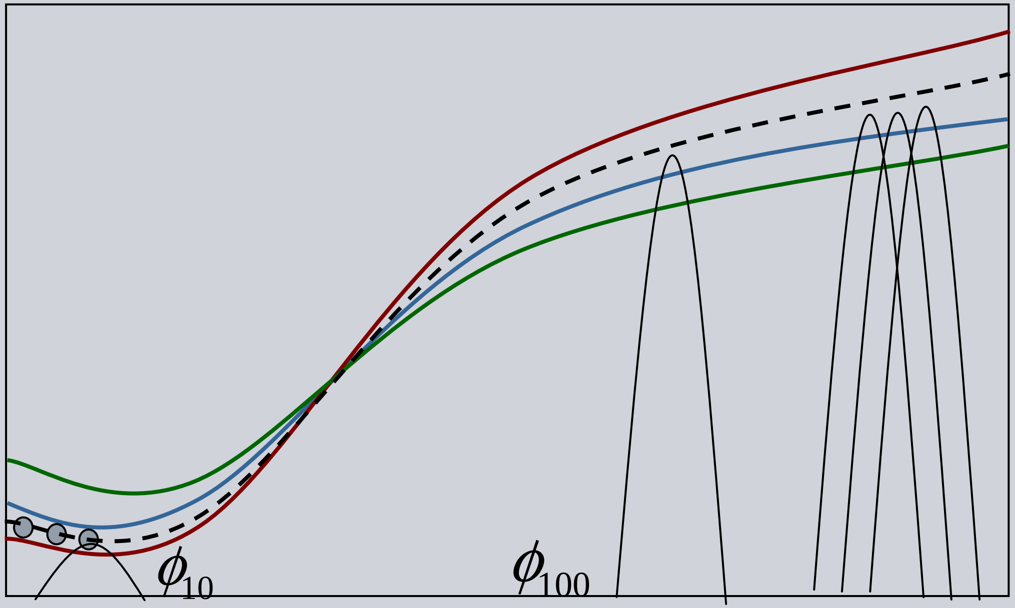
$$f_{new}(\mathbf{x}) = \sum_{l=1}^{L} w_{new,l} \phi_l(\mathbf{x})$$

## Motivation for Hierarchical Bayes

- Looking at other models $\hat{f}_j(x)$ another solution becomes more likely

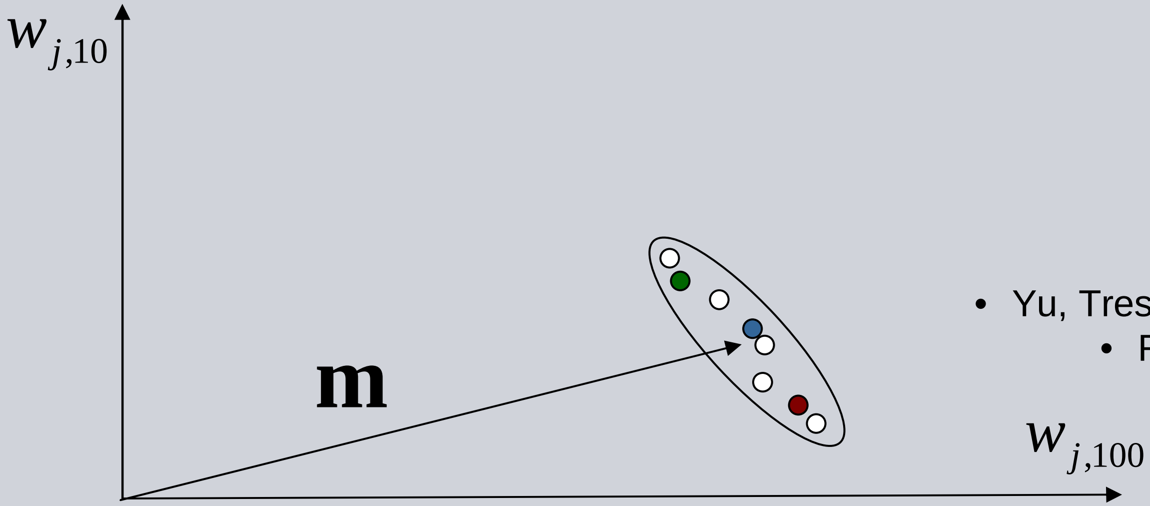


## Parameter Distributions




- The parameters for the different models might form again a Gaussian distribution

$$\mathbf{w}_{new} \mid \mathcal{D} \sim N(\mathbf{m}, \Sigma)$$




- Yu, Tresp, Schwaighofer (2005)
- Raina, Ng, Koller (2006)

## Learned Prior

- A new model sees the “learned” prior

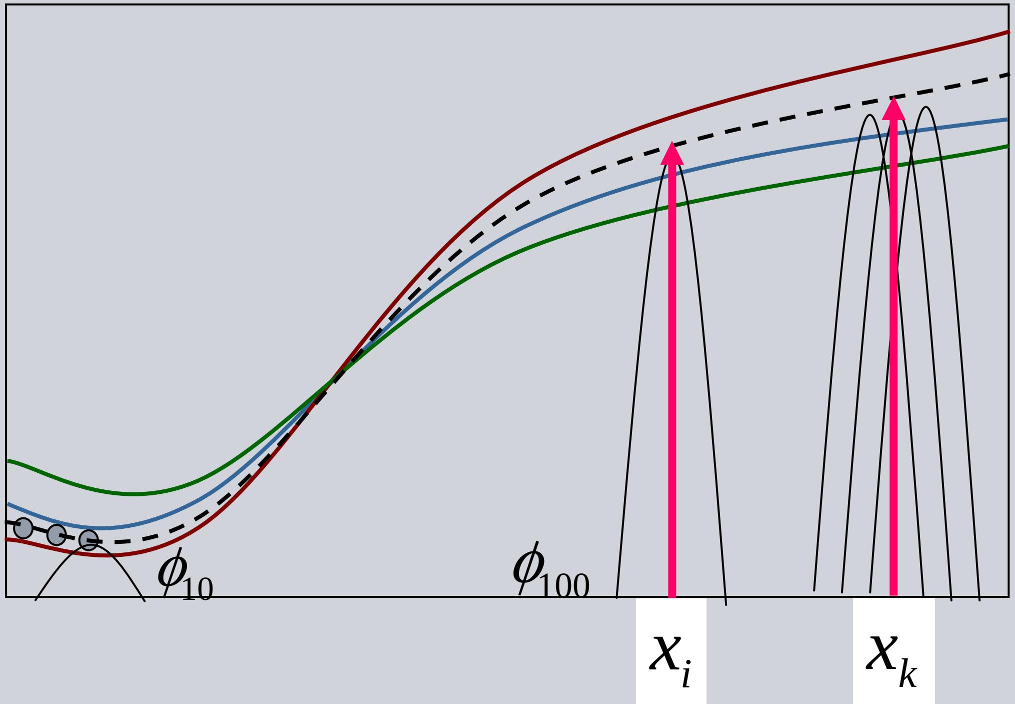
$$\mathbf{w}_{new} \mid \mathcal{D} \sim N(\mathbf{m}, \Sigma)$$

- With a Gaussian (learned) prior we obtain a Gaussian process with mean function and covariance kernel given by

$$E(f_{new}(\mathbf{x}) \mid \mathcal{D}) = \sum_{l=1}^{L} m_l \phi_l(\mathbf{x})$$

$$\mathrm{cov}(f_{new}(\mathbf{x}_i), f_{new}(\mathbf{x}_k) \mid \mathcal{D}) = \phi^T(\mathbf{x}_i)\Sigma\phi(\mathbf{x}_k)$$

# Learned Prior in Function Space




$$E(f_{new}(\mathbf{x}) \mid \mathcal{D}) = \sum_{l=1}^{L} m_l \phi_l(\mathbf{x})$$

$$\mathrm{cov}(f_{new}(\mathbf{x}_i), f_{new}(\mathbf{x}_k) \mid \mathcal{D}) = \phi^T(\mathbf{x}_i)\Sigma\phi(\mathbf{x}_k)$$

- So we got what we wanted: the new function is guided by the previously learned functions

## Covariance and Basis Functions

- We can decompose
  Singular Value Decomposition (SVD):

$$\Sigma = VDD^T V^T$$

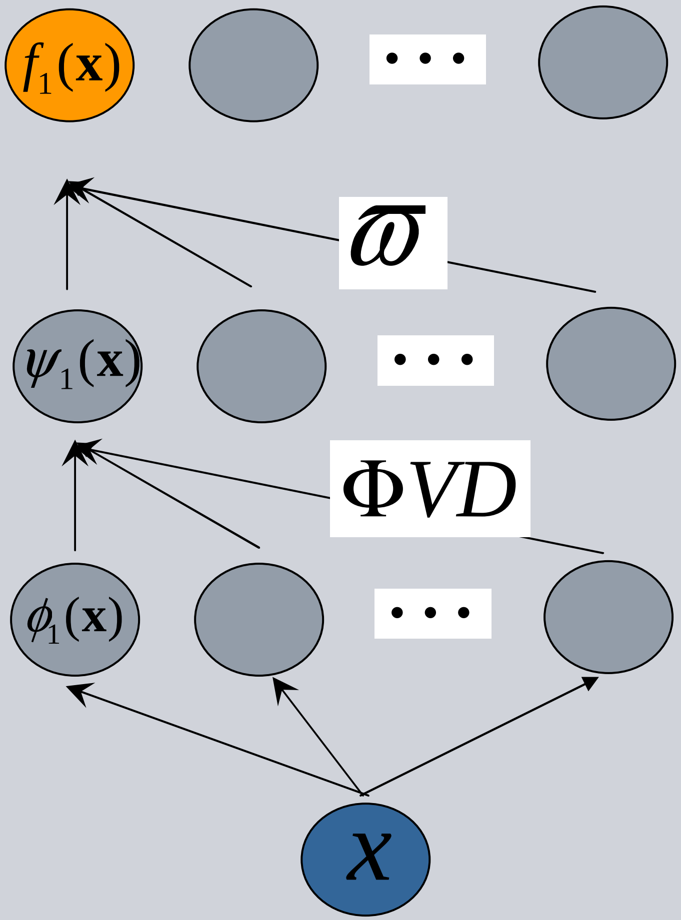
- And obtain:

$$\phi^T(\mathbf{x}_i)\Sigma\phi(\mathbf{x}_j) = \left(\phi^T(\mathbf{x}_i)VD\right)\left(\phi(\mathbf{x}_j)VD\right)^T$$

- From this view point the new model has a Gaussian parameter distribution with identity covariance matrix and with *new learned basis functions formed as linear combinations of the original basis functions:*

$$\psi_k(\mathbf{x}) = d_{k,k}\sum_l v_{l,k}\phi_l(\mathbf{x})$$

## Architecture: Hierarchical Bayesian Modeling




$$f_j(\mathbf{x}) = \sum_{k=1}^{K} \varpi_{k,j} \psi_k(\mathbf{x})$$

$$\psi_k(\mathbf{x}) = d_{k,k} \sum_{l} v_{l,k} \phi_l(\mathbf{x})$$

Bottleneck!

$$\phi_l(\mathbf{x})$$

Compare: Neural network with hidden layers and multiple outputs

## Technical Details: EM Updates

- In typical applications noisy measurements for the different situations are available. The design matrix for situation *j*: $\Phi_j$ inverse Wishart: $\mathcal{IW}$
- Complete data likelihood

$$N(\mathbf{m} \mid \mu, \eta^{-1}\Sigma)\mathcal{IW}(\Sigma \mid \alpha, \kappa)\prod_{j=1}^{M} N(y_{*,j} \mid \Phi_j \mathbf{w}_j, \sigma^2 I) N(\mathbf{w}_j \mid \mathbf{m}, \Sigma)$$

- E-step $P(\mathbf{w}_j \mid y_{*,j}, \mathbf{m}, \Sigma) = N(\mathbf{w}_j \mid \mathbf{r}_j, V_j)$

$$V_j = (\Sigma^{-1} + \frac{1}{\sigma^2}\Phi_j{}^T\Phi_j)^{-1} \text{ and } \quad \mathbf{r}_j = V_j(\frac{1}{\sigma^2}\Phi_j{}^T y_{*,j} + \Sigma^{-1}\mathbf{m})$$

- M-Step

$$\mathbf{m} = \frac{1}{M+\eta}(\sum_{j=1}^{M}\mathbf{r}_j + \eta\mu)$$

$$\Sigma = \frac{1}{M+L+\eta+\alpha+1}(\eta(\mathbf{m}-\mu)^T(\mathbf{m}-\mu) + \sum_{j=1}^{M}(\mathbf{m}-\mathbf{r}_j)^T(\mathbf{m}-\mathbf{r}_j) + \sum_{j=1}^{M}V_j + 2\kappa)$$

## Definition of Inverse Wishart

$$\mathcal{IW}(\Sigma \mid \alpha, \kappa) \propto \left(\det\Sigma\right)^{-(\delta+2L)/2} \exp[-\frac{1}{2}\mathrm{tr}(\kappa\Sigma^{-1})]$$

$$\delta = \alpha - L + 1$$

Marginalization consistent

A. P. Dawid. Some matrix-variate distribution theory: Notational considerations and a Bayesian application. Biometrika, 68(1), 1981

## Learned Basis Functions

- The key benefit in Hierarchical Bayesian modeling for linear systems is that common basis functions are learned that are used for all outputs
- Let’s briefly look at two solutions that also lead to a set of shared basis functions
  - Empirical basis functions
  - Basis functions derived from a SVD

## Empirical Basis Functions

- After training
  (regression with little noise)

$$k_{i,k} = \mathrm{cov}(f(x_i), f(x_k) \mid \mathcal{D}) = \psi^T(\mathbf{x}_i)\psi(\mathbf{x}_k)$$

$$K \approx \frac{1}{M} YY^T \qquad k_{i,k} \approx \frac{1}{M} y_{i,*} y_{k,*}^T$$

- Thus why not set $\psi_k(\mathbf{x}_i) = y_{i,k}$ $\qquad \varpi_{k,j} = \delta_{k,j}$
  - The learned basis functions are given by the output data
  - Disadvantage: different situations do not benefit from one another
  - Still, we can make predictions based on the estimated *K*

## Learned Basis Functions Based on SVD

- With SVD $Y = UDV^T$
- We can decompose: $\frac{1}{M} YY^T = \frac{1}{M} UD^2U \approx \frac{1}{M} U\left(D^{(rr)}\right)^2 U$
- In $D^{(rr)}$ we have $d_{k,k}^{(rr)} = 0$ for $k > r$
- Thus another sensible set of learned basis functions might be defined by

$$\psi_k(\mathbf{x}_i) = d_{k,k} u_{i,k} = \sum_j v_{j,k} y_{i,j} \qquad \varpi_{k,j} = v_{j,k}$$

- Since here, the singular vectors (columns of *U*) are calculated based on all data, statistical strength is shared
- This might explain the great success of matrix decomposition methods in collaborative filtering (e.g., in the Netflix competition)

## Comments

- Advantages of Hierarchical Bayes:
  - Inclusion of prior knowledge by defining the basis functions
  - Generalization to new inputs
  - No problems with missing outputs

- Alternatively: in Hierarchical Bayes inference is often performed via Gibbs sampling or other approximate methods such as variational learning (see, e.g., Latent Dirichlet Allocation, LDA)

  (Blei, Ng, Jordan, 2003)

- Naturally Hierarchical Bayes is also applicable beyond linear models

- Gelman, Carlin, Stern and Rubin (2003) provide a thorough discussion of Hierarchical Bayesian models

## Three Phases in HB modeling

- **First Phase:** With no data yet available the model for a new situation follows the prior (the mean function)
- **Second Phase:** With some data available for a new situation, a model follows more closely a previous model that fits those data well
- **Finally:** With increasing data available, the model becomes independent of the learned prior

- Dimensional reduction: Derived basis functions

$$\psi_k(\mathbf{x}) = d_{k,k} \sum_l v_{l,k} \phi_l(\mathbf{x})$$

with a small $d_{k,k}$ are ignored

## When Hastie’s Statement is Applicable

- If the hyper parameters (in our case: $\mathbf{m}, \Sigma$ ) are known a prior, i.e., they represent the empirical parameter distribution, then all output functions are independent
  - Or: if output functions have no common prior distribution (predicting apples and oranges)

- In contrast, if the prior is learned then all measurements influence all predictions!

## Frequentist Equivalent: Mixed Models

$$y_{*,j} = \Phi_j \mathbf{m} + Z_j \mathbf{b}_j + \varepsilon_j$$

$$\mathbf{b}_j \propto N_K(0, \Sigma)$$
$$\varepsilon_j \propto N_{N_j}(0, \sigma^2 \Lambda_j)$$

- Known: $\Phi_j, Z_j$
- (unknown but) Fixed effect: $\mathbf{m}$
- Random effect: $\mathbf{b}_j$
- Special case: $Z_j = \Phi_j$
  - *regression model with random coefficients*
- Relationship to HB-model: $\mathbf{w}_j = \mathbf{m} + \mathbf{b}_j$

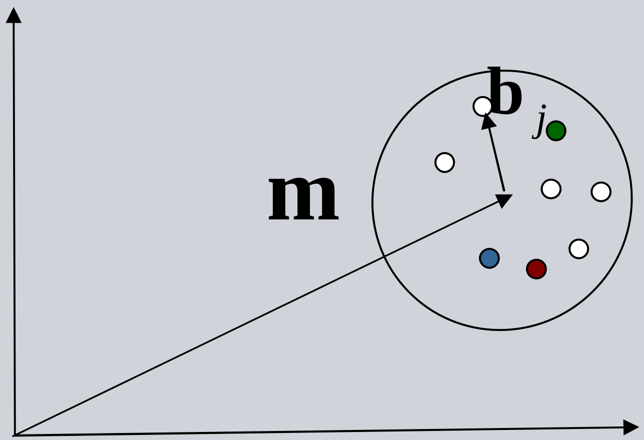
- New: correlated contributions that cannot be explained by the inputs ("noise")
- Collaborative effect!

- MM: As Bayesian as a frequentist will ever get
- HB: as frequentist as a Bayesian will ever get

## Non-probabilistic Equivalent: Regularized Multi-task Learning

$$\min_{\mathbf{w}_0, \mathbf{v}_j} \left\{ \sum_{j=1}^{M} \sum_{i=1}^{N} L\left(f_j(\mathbf{x}_i), y_{ij}\right) + \frac{\lambda_1}{M} \sum_{j=1}^{M} \left\| \mathbf{b}_j \right\|^2 + \lambda_2 \left\| \mathbf{m} \right\|^2 \right\}$$

$$f_j(\mathbf{x}) = \mathbf{w}_j^T \mathbf{x} = (\mathbf{m} + \mathbf{b}_j)^T \mathbf{x}$$




- Assume an "isotropic" covariance
- 2-norm constraint
- Learn the shared mean of linear weights
- Convex optimization problem

Evgeniou, Micchelli, Pontil (2006)

## Non-probabilistic Equivalent: Convex Multi-task Feature Learning

$$\min_{\{\mathbf{w}_j\}} \left\{ \sum_{j=1}^{M} \sum_{i=1}^{N} L\left(\mathbf{w}_j^T \mathbf{x}_i, y_{ij}\right) + \lambda \sum_{d=1}^{D} \sqrt{\sum_{j=1}^{M} w_{jd}^2} \right\} \Leftrightarrow$$

$$\min_{\{\mathbf{w}_j\},\beta} \left\{ \sum_{j=1}^{M} \sum_{i=1}^{N} L\left(\mathbf{w}_j^T \mathbf{x}_i, y_{ij}\right) + \sum_{d=1}^{D} \left( \beta_d^{-1} \sum_{j=1}^{M} w_{jd}^2 \right) + \frac{\lambda^2}{4} \|\beta\|_1 \right\}$$

Assume a shared diagonal covariance

Lead to a jointly sparse result (select features subset for all tasks)

- Convex optimization
- L1-L2 norm
- Argyriou, Evgeniou, Pontil (2006)
- A similar model via an extension of relevant vector machine, by J. Zhang (2005)

## Gaussian Process Hierarchical Bayes (GP-HB)

- As already discussed, a system of fixed basis functions and Gaussian weight prior

$$f(\mathbf{x}_i) = \sum_{l}^{M} w_l \phi_l(\mathbf{x}_i) \qquad \mathbf{w} \propto N(\mathbf{m}, \Sigma)$$

- … is technically equivalent to a Gaussian process with covariance $k(\mathbf{x}_i, \mathbf{x}_k) = \phi^T(\mathbf{x}_i)\Sigma\phi(\mathbf{x}_k)$

and mean function $m(\mathbf{x}) = \sum_{l}^{M} m_l \phi_l(\mathbf{x}_i)$

- Thus:
  - as parametric HB boils down to learning $\mathbf{m}, \Sigma$,
  - GP-HB boils down to learning $m(\mathbf{x}), k(\mathbf{x}_i, \mathbf{x}_k)$

# Comparing Representations and Kernels

- Based on our discussion we can derive the following kernels

| Approach | Basis fcts. | Gram Matrix |
|---|---|---|
| Empirial | $Y$ | $K = \frac{1}{M} YY^T$ |
| SVD | $UD^{(rr)} = \Phi V^{(rr)}$ | $K = \frac{1}{M} U\left(D^{(rr)}\right)^2 U^T$ |
| Hierarchical Bayes | $\Psi$ | $K = \Psi\Psi^T$ |

## GP-HB: Learning in Function Space

- Now we consider GP-HB in *function* space
- A prior for mean and covariance kernel is defined for a finite set of points (typically the training data and some test points))

$$N(\mathbf{m} \mid \mu, \eta^{-1} K)\mathcal{IW}(K \mid \alpha, \kappa)$$

- MAP estimates for kernel and mean are calculated using EM equations
- $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ is the base kernel. $\kappa$ is the respective Gram matrix.

## EM Learning for GP-HB

- In typical applications noisy measurements for the different situations are available (for missing data: simply set noise variance to infinity)
- Complete data likelihood

$$N(\mathbf{m} \mid \mu, \eta^{-1} K)\mathcal{IW}(K \mid \alpha, \kappa)\prod_{j=1}^{M} N(y_{*,j} \mid f_{*,j}, S_j) N(f_{*,j} \mid \mathbf{m}, K)$$

$$S_j = diag(\sigma^2_{i,j})$$

- E-step

$$P(f_{*,j} \mid y_{*,j}, \mathbf{m}, K) = N(f_{*,j} \mid \mathbf{r}_j, V_j)$$

$$V_j = (K^{-1} + S_j^{-1})^{-1} \quad \text{and} \quad \mathbf{r}_j = V(S_j^{-1} y_{*,j} + K^{-1}\mathbf{m})$$

- M-Step

$$\mathbf{m} = \frac{1}{M+\eta}\left(\sum_{j=1}^{M} \mathbf{r}_j + \eta\mu\right)$$

$$K = \frac{1}{M+L+\eta+\alpha+1}\left(\eta(\mathbf{m}-\mu)^T(\mathbf{m}-\mu) + \sum_{j=1}^{M}(\mathbf{m}-\mathbf{r}_j)^T(\mathbf{m}-\mathbf{r}_j) + \sum_{j=1}^{M} V_j + 2\kappa\right)$$

## Induction: Generalizing to New Inputs

- To generalize to new inputs (induction) one can use approximations. The simplest one, based on the Nyström approximation, gives

$$k(x_i, x_k) = \kappa^T(\cdot, x_i) K^{-1} \kappa(\cdot, x_k)$$

- Schwaighofer, Tresp, Yu (2004)
- Yu, Tresp, Schwaighofer (2005)
- Lawrence and Platt (2004): similar approach but without priors on mean and kernel

## Induction: Generalizing to New Inputs

- To generalize to new inputs (induction) one can use different approximations

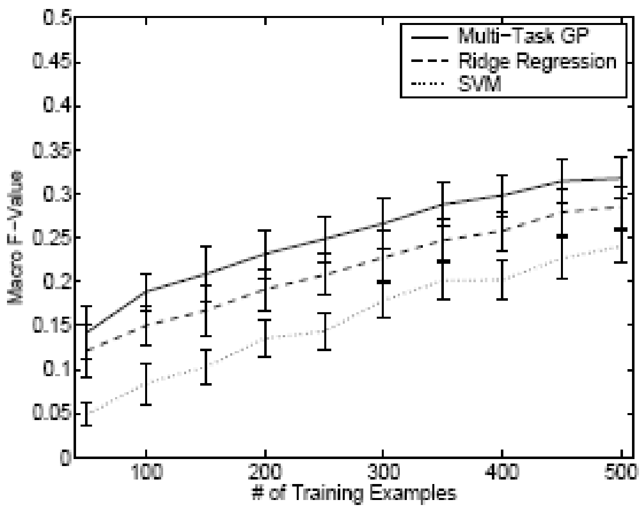
- Schwaighofer, Tresp, Yu (2004)
- Yu, Tresp, Schwaighofer (2005)
- Lawrence and Platt (2004): similar approach but without priors on mean and kernel

# Predicting Reuter's labels

- 10000 documents with a total of 81 labels (situations) with TFIDF features; On average each document has 3.96 labels.
- The test set contains 9700 examples; All: evaluation on all the test points. Partially Labeled: each test document with at least one label in some category.

*Table 1.* Comparison of four algorithms for text categorization on RCV1

| | ALL | | | PARTIALLY LABELED | | |
|---|---|---|---|---|---|---|
| | AUC | F-MICRO | F-MACRO | AUC | F-MICRO | F-MACRO |
| MULTI-TASK GP | 0.773 | 0.605 | 0.260 | 0.826 | 0.623 | 0.281 |
| REGULARIZED MULTI-TASK LEARNING | 0.701 | 0.571 | 0.232 | 0.709 | 0.545 | 0.216 |
| RIDGE REGRESSION | 0.756 | 0.584 | 0.245 | 0.771 | 0.564 | 0.240 |
| SVM | 0.697 | 0.573 | 0.221 | 0.716 | 0.547 | 0.212 |



## Fast Implementation of GP-HB

Table 5: RMSE of various matrix factorization methods on the Netflix test set

| Method | RMSE |
|---|---|
| Baseline | 0.9514 |
| VB [6] | 0.9141 |
| SVD [5] | 0.920 |
| BPMF [10] | 0.8954 |
| NSVD | 0.9216 |
| NPCA | **0.8926** |

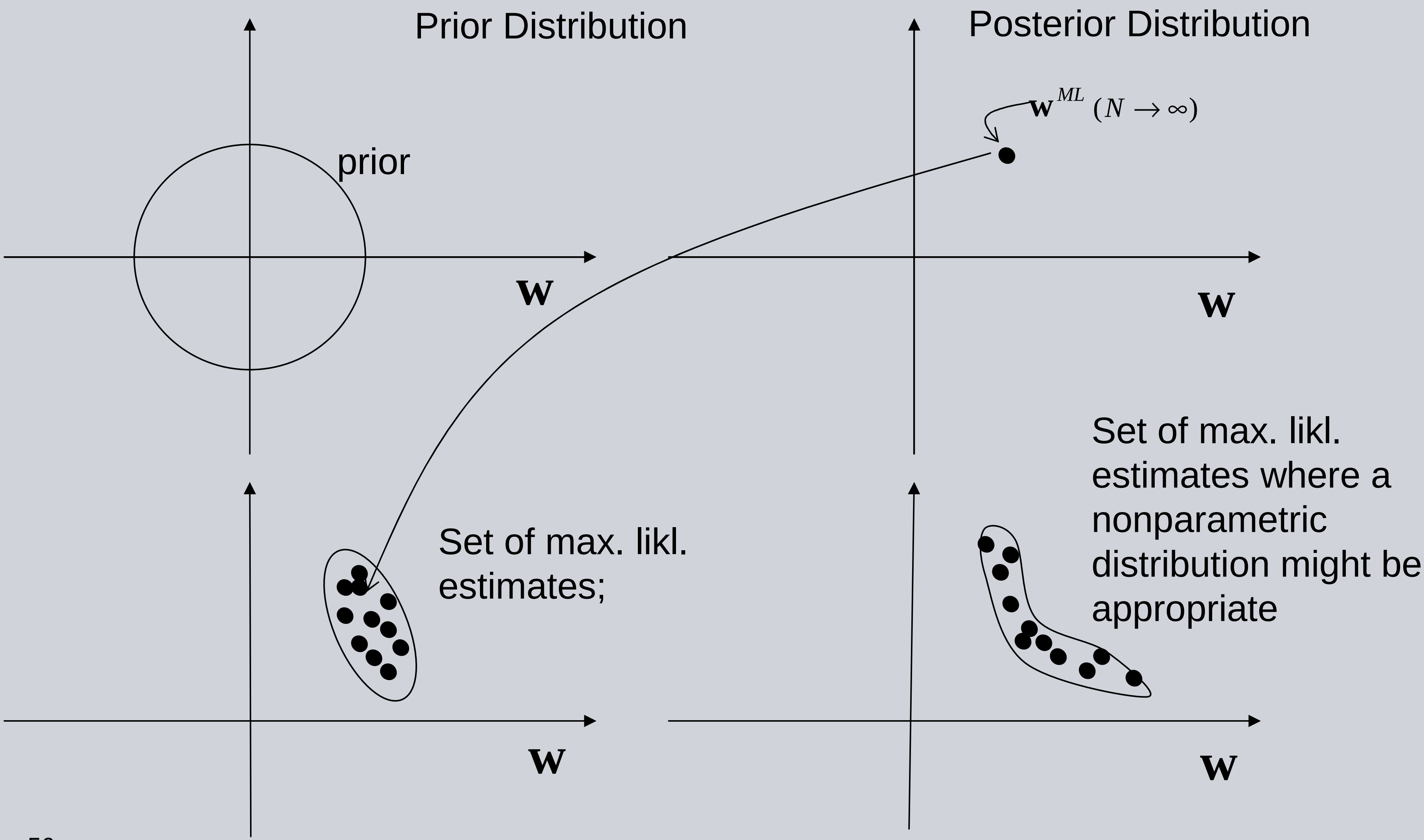
- Straightforward of the EM approach on Netflix will take thousands of hours per iteration

- Fast implementation plus model simplification leads to 5h/iterations
- VB: variational Bayes matrix factorization. SVD: SVD for sparse matrices. BPMF: Bayesian Probabilistic Matrix Factorization. NSVD: Max Margin Matrix Factorization. NPCA: nonparametric PCA (GP-HB)
  - Yu, Zhu, Lafferty, Gong (2009)

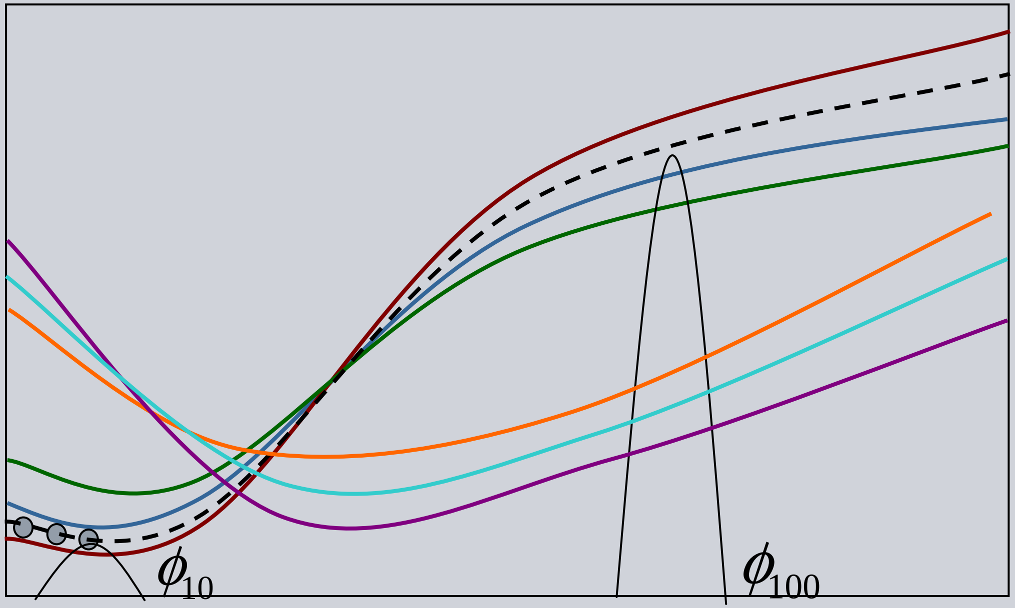
# I.C.
# Nonparametric Hierarchical Bayes

- The prior needs to be quite expressive!

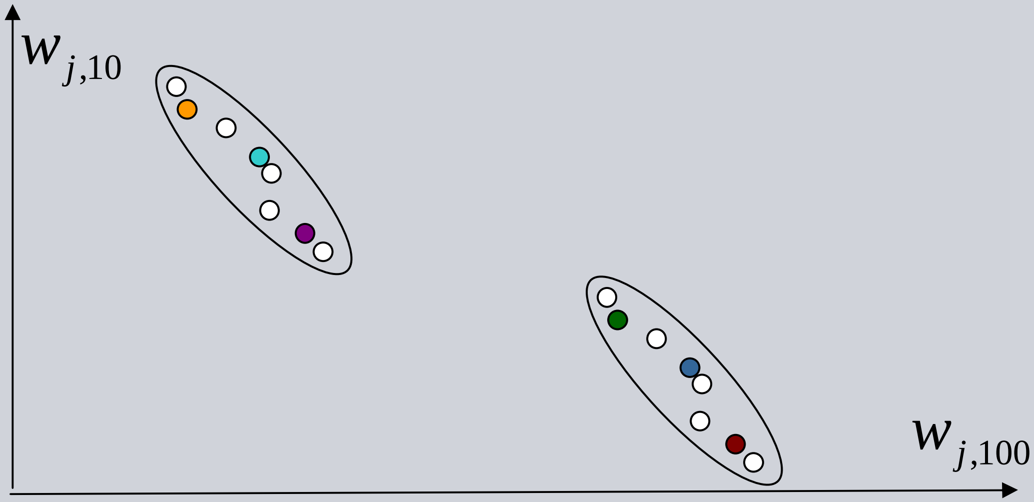
## A Problem with Low-dimensional HB Approaches



## Another View




- A latent mixture model for the distribution of the parameters
- Latent variable (clustering) model of functions, not data points!
- Multi-modal learned prior distribution



## (Soft) Grouping of Variables or Functions

- Colors: cluster assignment (grouping of outputs/functions, not data points)
- In each cluster, parameters are shared

$$
\left(\begin{array}{c|ccccccc}
\mathbf{x}_1 & y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} & y_{1,5} & y_{1,6} & y_{1,M} \\
\cdot & & & & & & & \\
\cdot & & & & & & & \\
\mathbf{x}_N & y_{N,1} & y_{N,2} & y_{N,3} & y_{N,4} & y_{N,5} & y_{N,6} & y_{N,M}
\end{array}\right)
$$

## Finite Models: A Particular Mixtures of Experts Models (Regression)

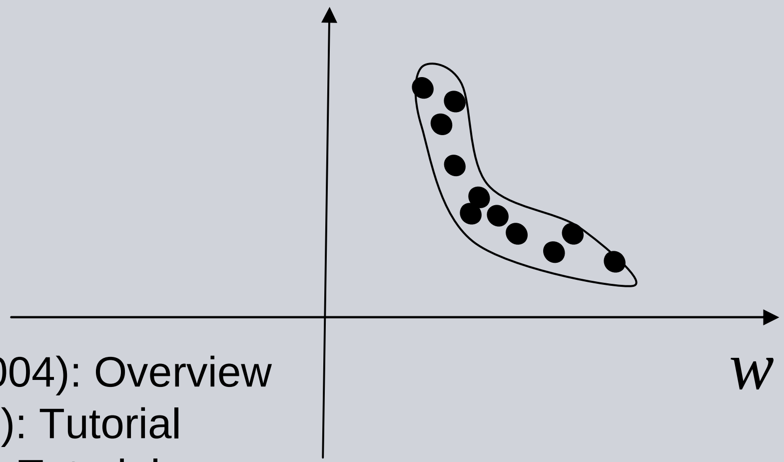
- After training, let parameter vector $\mathbf{w}_l$ be assigned to cluster *l*
- As a prediction for situation *j*, based its past data $\mathcal{D}_j$ one obtains

$$E(f_j(\mathbf{x})) = \sum_{l=1}^{L} f(\mathbf{x}, \mathbf{w}_l) P(Z_j = l \mid \mathcal{D}_j)$$

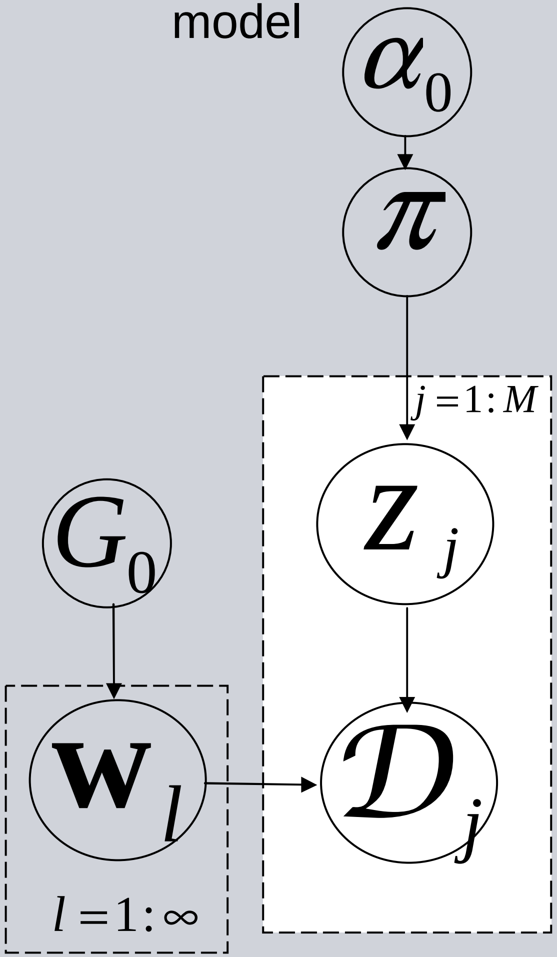
- Can be interpreted as a mixture of expert approach with experts $f(\mathbf{x}, \mathbf{w}_l)$ and weight $P(Z_{j=l} \mid \mathcal{D}_j)$
- Note that in contrast to the typical mixture of expert approach, we assign a whole function (i.e., situation) to a component
- Tresp and Yu (2004)

## Dirichlet Process Mixture Models for Multitask Learning

- If, in a Bayesian approach, we let the *number of components go to infinity*, we obtain a *Dirichlet process mixture model*
- Automatic model selection: in the sampling procedure only a finite number of states is being used
- This is equivalent to a nonparametric hierarchical Bayesian approach




Stick breaking representation of a Dirichlet process mixture model




- Tresp, Yu (2004): Overview
- Jordan (2005): Tutorial
- Tresp (2006): Tutorial
- Xue, Liao, Carin, Krishnapuram (2007)

## Summary Hierarchical Bayes

- **Main benefit:** data for a given situation is supported by data from other situations
- **Training:**
  - Inputs (objects) can be arbitrary in different situations

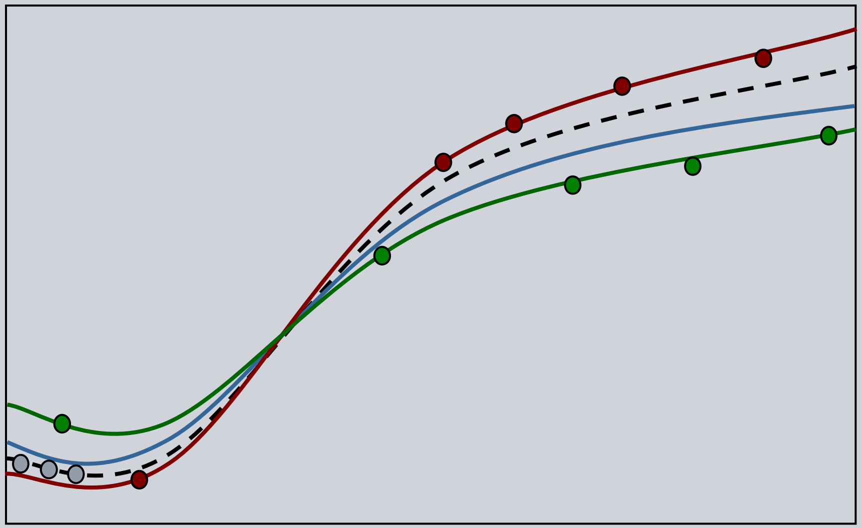
    (from another view: ***no problems with missing outputs***)

- **Generalization**
  - to new objects (inputs) is possible
  - to new situations (output dimensions) is possible
- **Output driven regularization / dimensionality reduction!**
- **Not limited to models that are linear in the parameters**
- More helpful references:
- Caruana (1995), Thrun (1996): early work
  - Zhang, Ghahramani and Yang (2005):find latent independent components (not just uncorrelated components)
  - Barutcuoglu, Schapire and Troyanskaya (2006): application to gene function prediction
  - Krishnapuram, Yu, Yakhnenko, Rao, Carin (2008): recent NIPS workshop

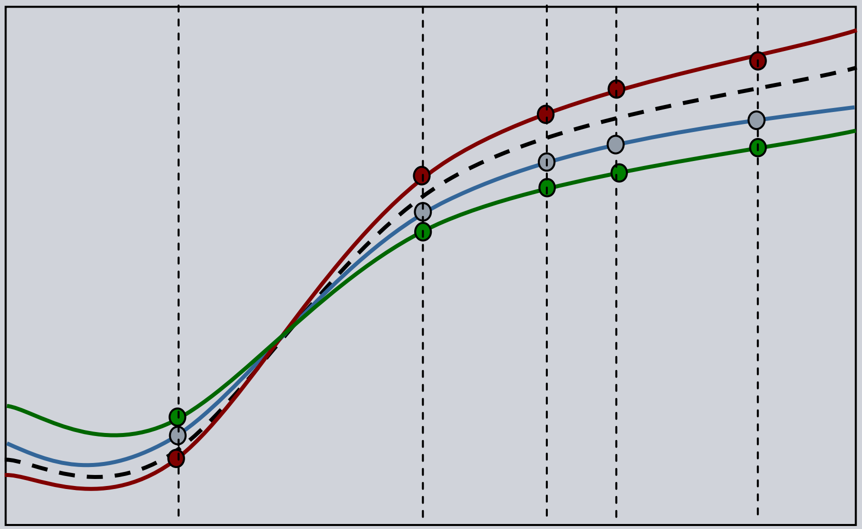
# II. Projection Methods

- For the set of objects all (or many) outputs (labels) are available

- before



- now



## Projection Methods:

- Recall: Hierarchical Bayes
  defines new derived basis functions

$$\psi_k(\mathbf{x}) = d_{k,k} \sum_l v_{l,k} \phi_l(\mathbf{x})$$

- The projection methods considered here have a similar goal: they define new basis functions as a linear combination of the existing basis functions, such that the (independent) prediction of the outputs is improved

## Projection Methods: Principle Component Regression

- Principle component regression (PCR) is based on an optimal approximation of the design matrix

$$\min \| \Phi - W^T V \|_F \quad \text{where} \quad V^T V = I$$

- The derived basis functions are

$$\psi_j(\mathbf{x}) = \sum_k v_{k,j} \phi_k(\mathbf{x})$$

- In our context, the disadvantage of PCR is that it only considers input information

## Projection Methods: Canonical Correlation

- It is desirable to also take into account output information

- An example is Canonical Correlation Analysis (CCA), which solves

$$\max_{u,v} u^T \Phi^T Y v \qquad u^T u = 1, \quad v^T v = 1$$

- The solution is based on a generalized eigenvector problem

- Related: Partial Least Squares (PLS), Linear Discriminant Analysis (LDA)

  - Shawe-Taylor and Christianini (2004)

## MORP: A New Projection Methods

- MORP: Multi-output regularized projection uses the cost function

$$(1-\beta)\|\Phi - VA\|_F + \beta\|Y - VB\|_F$$
$$s.t.\ V^T V = I,\ V = \Phi W$$
$$A = V^T \Phi,\ B = V^T Y$$

- The solution takes on the form

$$\psi_k(x) = d_{k,k} \sum_l w_{l,k} \phi_l(\mathbf{x})$$

- Where $d_k$ and $w_{l,k}$ are found by solving a generalized eigenvalue problem

## MORP Applications

- Task: Assigning several labels to images
    - Images can be assigned to 37 categories

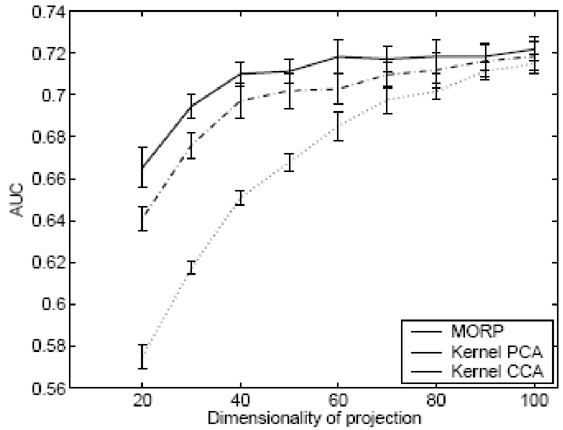
- Task: Predicting ratings of paintings for several users
    - Ratings from several users are assigned to a painting

- Task: Predicting Reuters labels
    - A news text can be assigned to several classes

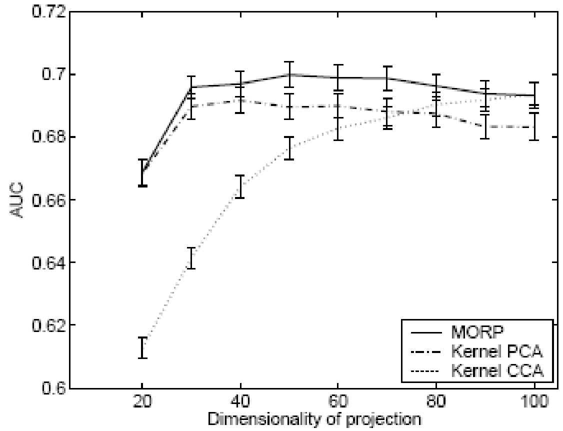
- Yu, Yu, and Tresp (2005)
- Yu, Yu, Tresp, Kriegel (2006)

# MORP: Predicting Image Labels based on Image Features

The experiment is based on a subset of Corel image database, containing 1021 images that have been manually assigned into 35 categories (labels) based on their contents. On average, each image belongs to 3.6 categories and each category on average contains 98 positive examples.




Test on same categories



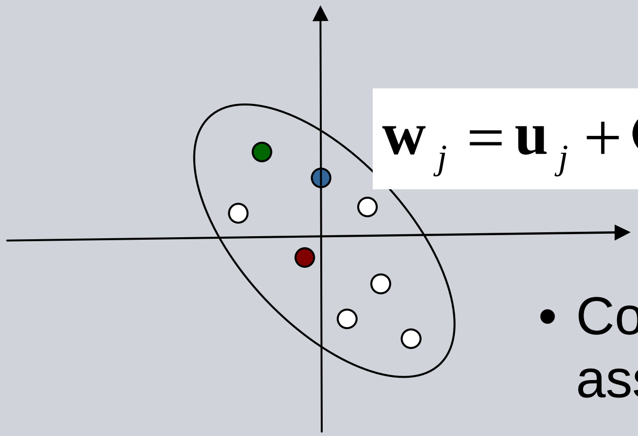

Test on new categories with
previously learned representation

## Another Projection Approach using SVD

$$\min_{\{\mathbf{u}_j, \mathbf{v}_j\}, \Theta} \left\{ \sum_{j=1}^{M} \sum_{i=1}^{N} L\left(f_j(\mathbf{x}_i), y_{ij}\right) + \lambda \left\|\mathbf{u}_j\right\|^2 \right\}$$

Subject to $\Theta\Theta^T = I_{h\times h}$, $f_j(\mathbf{x}) = \mathbf{w}_j^T\mathbf{x} = \mathbf{u}_j^T\mathbf{x} + \mathbf{v}_j^T\boxed{\Theta\mathbf{x}}$

project $x$ into a lower-dim space



$$\mathbf{w}_j = \mathbf{u}_j + \Theta^T \mathbf{v}_j$$

- Connection to hierarchical Bayes: it implicitly assumes a learned covariance for $w$ with the form

$$I + \Theta^T\Theta$$

- Ando and Zhang (2005)

## Summary: Projection Methods

- Suitable when for a given *x*, the target is known at all (or most) situations in training but in testing, no outputs are available
- Close connection to *Hierarchical Bayes* modeling

- Suitable for predicting many labels of objects (text annotaions, image annotations) based on object features!

- Generalization
  - to new objects (inputs) is possible
  - to new situations (output dimensions) is possible

- Output driven dimensionality reduction!
- Limited to models that are linear in the parameters resp. kernel representations
- There is a huge literature on projection methods

  (e.g., papers in Hardoon, Leen, Kaski and Shawe-Taylor (2008)

# III. Multivariate Models and Structured Outputs

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & & y_{1,M} \\ \cdot & & & \\ \cdot & & & \\ \mathbf{x}_N & y_{N,1} & & y_{N,M} \end{pmatrix}$$

## Main Difference

- With Hierarchical Bayes and with Projection Methods: after training, there is no coupling between the various outputs

$$P(y_{i,*} \mid \mathbf{x}_i, \mathbf{w}) = \prod_{j=1}^{M} P(y_{i,j} \mid \mathbf{x}_i, \mathbf{w}_i)$$

- Now we consider models, for which -after training- the dependencies between the outputs are part of the model

$$P(y_{i,1}, \cdots, y_{i,M} \mid \mathbf{x}_i, \mathbf{w})$$

## Predicting a Single Output

- From $P(y_{i,1},\cdots,y_{i,M} \mid \mathbf{x}_i)$

we can marginalize and obtain

$$P(y_{i,j} \mid \mathbf{x}_i) = \sum_{y_{i,1},\cdots,y_{i,M} \ \backslash \ y_{i,j}} P(y_{i,1},\cdots,y_{i,M} \mid \mathbf{x}_i)$$

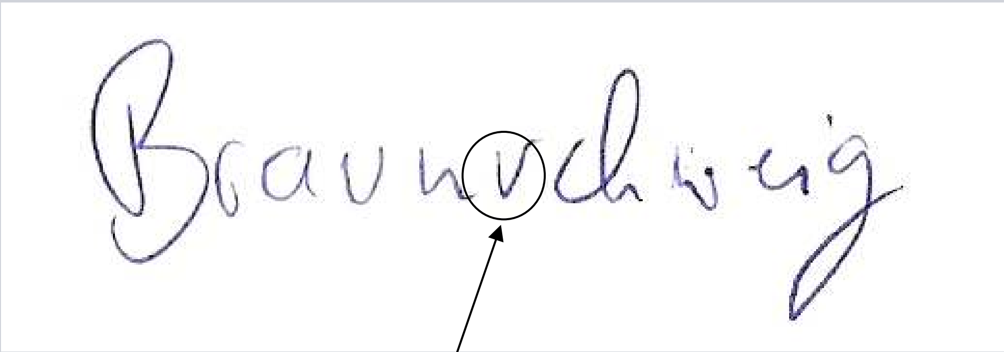
Thus the marginal of a single output variable given the input is, in general, a *complex mixture model*

## Motivation for Multivariate Models / Structured Outputs Models

- $P(y_{i,j} \mid \mathbf{x}_i)$ might be highly complex (a complex mixture model) and a direct model becomes impractical from bias/variance considerations
- Statistical strength is shared since parameters depend on all outputs
- Structured outputs:
  - Dimensionality reduction via independence assumptions
  - Dimensionality reduction via  parameter sharing
  - Dimensionality reduction via locality: a given output variable is directly dependent on only a subset of the inputs
- *In contrast a hierarchical Bayes model might be more suitable if conditional model follow similar and simple models*

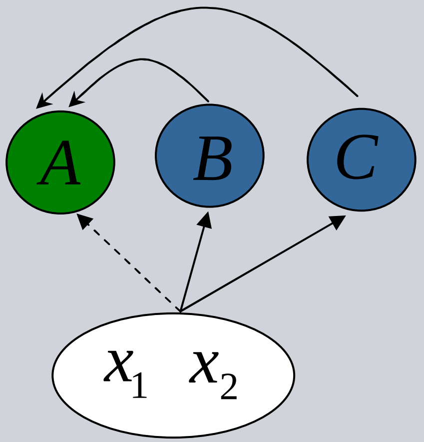
## Intuition: Structured Output Prediction Problems

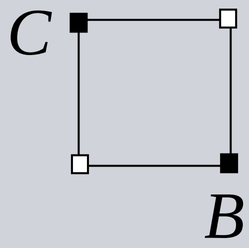
- Exploit correlations and constraints in the outputs
- Based on independent classification, since the “v” had a higher probability than an “s”, an OCR gives “Braunvchweig” as an answer
  - Since “sch” is very common in German, an “s” becomes more likely
  - “Braunschweig” is in the dictionary



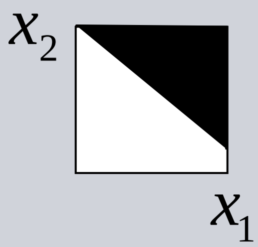
“s” or “v”

## Intuitive Example

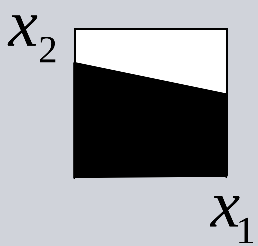



$A \mid B, C$

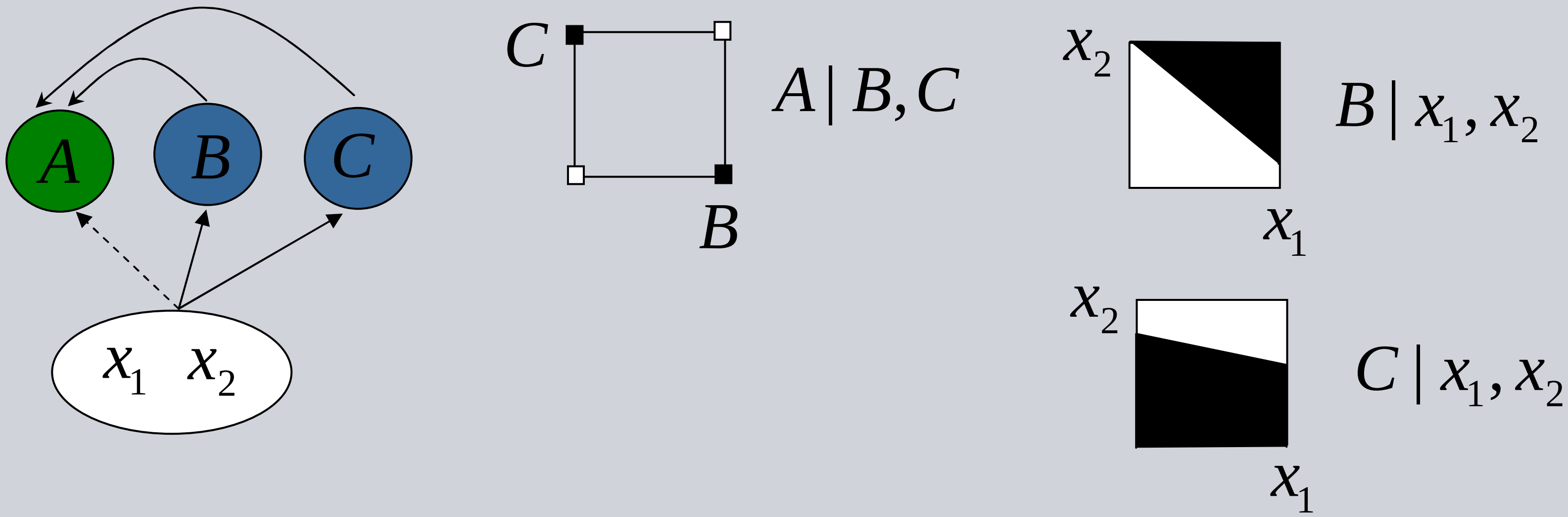


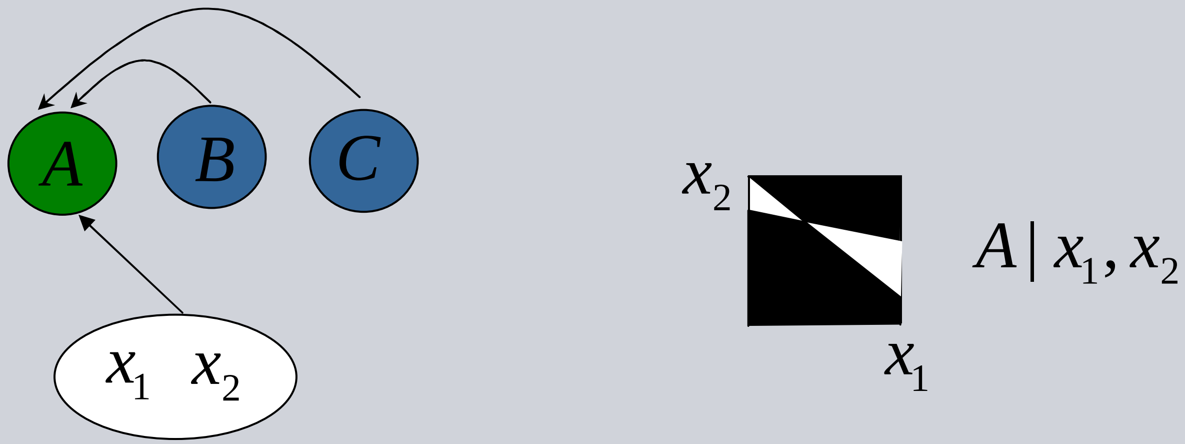

$B \mid x_1, x_2$




$C \mid x_1, x_2$

## Intuitive Example



## Examples

- Text to text-content (annotation)
- Text to parse trees
- Machine translation: English to French

- Images to image segmentation
- Images to image content
- Images to image annotation
- Images to image 3D pose
- Images to image robot arm coordinates
- From projections to reconstructed de-noised image (CT, MRI)

- DNA to DNA-segmentation
- DNA to protein structure

## Important Model Class: Conditional Log-Linear Models

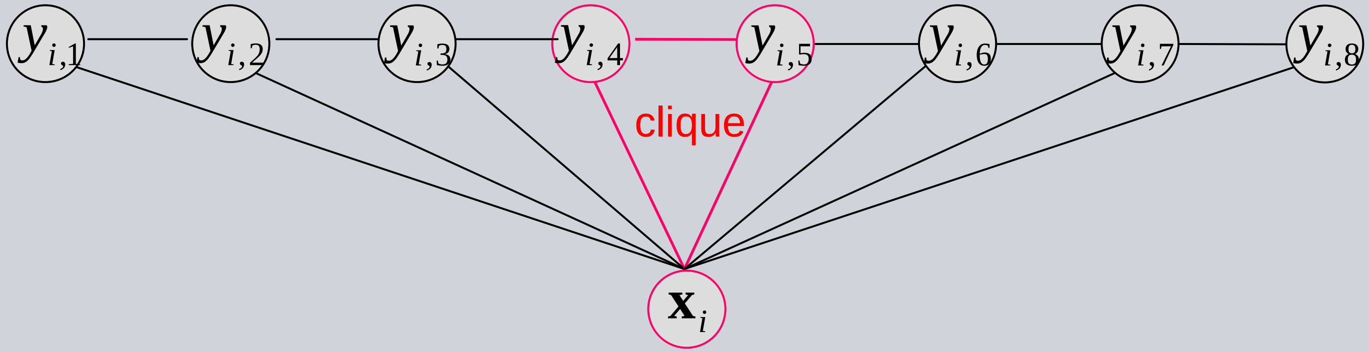
- How does one design interesting multivariate models?
- An interesting class: conditional log-linear models (a.k.a generalized linear models)
- Model design boils down to the design of interesting features

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i, y_{i,*})$$

$$\log P(y_{i,*} \mid \mathbf{x}_i) = -\log Z(\mathbf{x}_i) + \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i, y_{i,*})$$

- Feature functions (input, output): $f_k(\mathbf{x}_i, y_{i,*})$ Parameters: $\lambda_k$

## Conditional Log-Linear Models from Graph Structure

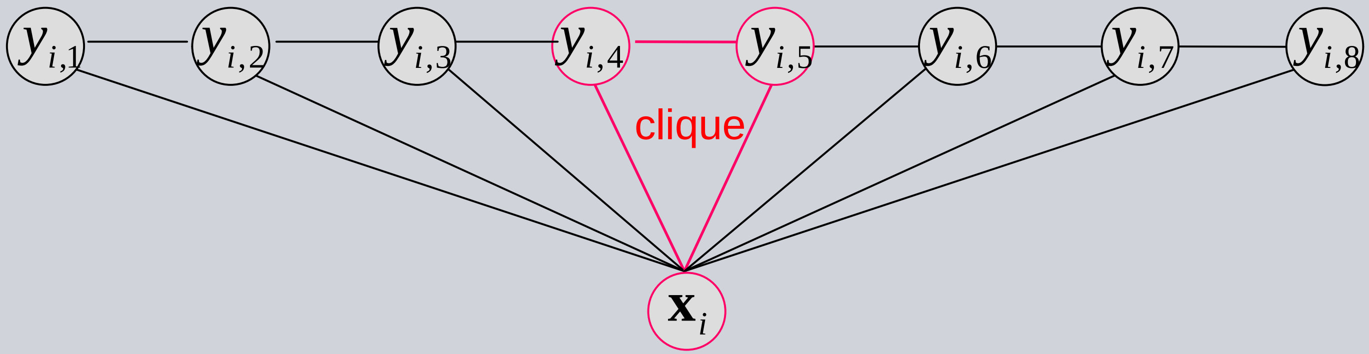



- Given a undirected graphical structure and its independence assumptions, a probability distribution factorizes in clique potentials as

$$P(\mathbf{x}_i, y_{i,*}) = \frac{1}{Z} \prod_c g_c(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \prod_c g_c(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

## Conditional Log-Linear Models from Graph Structure



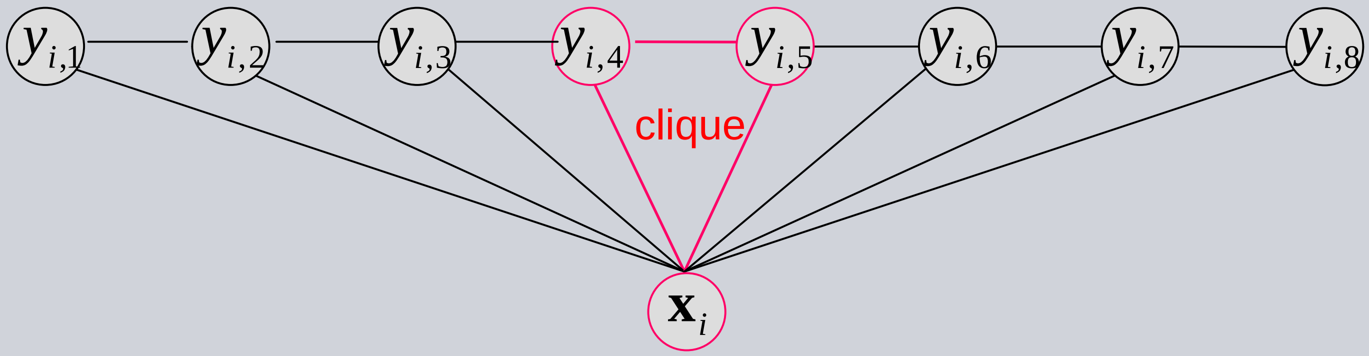

- A particular parameterization

$$g(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)}) = \exp \sum_{k=1}^{K} \lambda_{c,k} f_{c,k}(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_{c,k} f_{c,k}(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

- If the features imply an independency structure, conditional log-linear models are also known as
  - Conditional Markov networks
  - Conditional (Markov) Random Fields (CRFs)

## Parameters Sharing

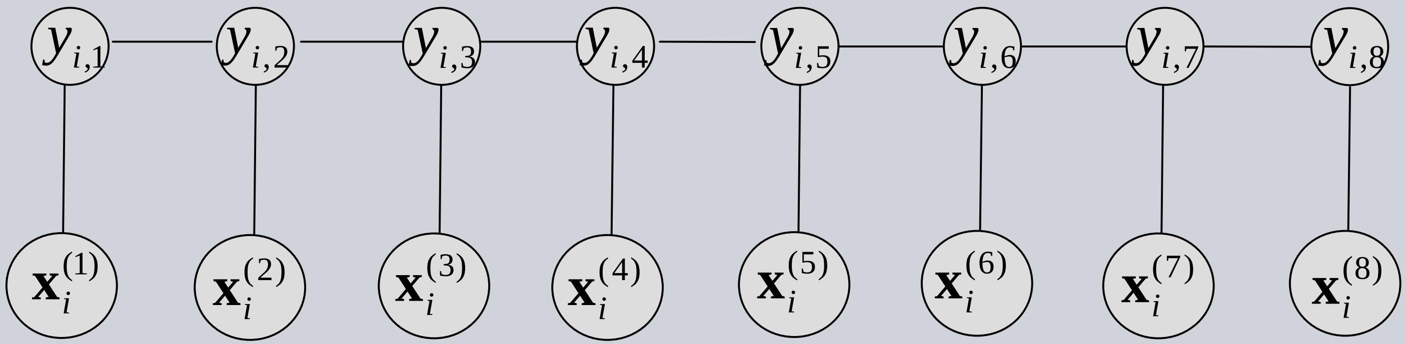



- Often one assumes some invariance, e.g.,

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

- Each clique uses the same feature functions
  - Data efficiency
  - Can handle sequences with varying lengths

# Social Network Type System




- Often, feature functions only involve some (local) input set: effective input dimensionality reduction
- Examples: social network analysis, hypertext classification, image reconstruction

- This is typical for situations where $\mathbf{x}_i^{(k)}, y_{i,k}$ represent attributes and class labels of object *k: in this case there is often only one data point available (e.g., only one social network)*

$$\boxed{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(M)}}, \boxed{y_1, \ldots y_M}$$

- Also: semi-supervised learning is often applicable (Zhu, 2005)

## Typical Design Approaches for Multivariate / Structured Output Models

1. Conditional from joint $$P(x, y \mid \mathbf{w}) \rightarrow P(y \mid x, \mathbf{w})$$

2. Direct approach $$P(y \mid x, \mathbf{w})$$

3. From marginal to conditional $$P(y \mid \mathbf{w}) \rightarrow P(y \mid \mathbf{w}(x))$$

## Multivariate Modeling: Conditional from Joint

- Train a joint probabilistic model

$$P(y_{i,1},\cdots,y_{i,M},x_i \mid \mathbf{w})$$

- … and then condition after training

$$P(y_{i,1},\cdots,y_{i,M} \mid x_i,\mathbf{w})$$

Applicable
- If the joint model  is easy to train
- If conditioning on the input is simple

## Conditional from Joint: Mixture Model

- Joint distribution (complete data)

$$P(y_{i,*}, \mathbf{x}_i, Z_i = l) = P(Z_i = l) P(y_{i,*}, \mathbf{x}_i \mid Z_i = l)$$

- Integration out the latent variable  leads to the log-likelihood (EM-training)

$$l = \sum_{i=1}^{N} \log \sum_{l} P(Z_i = l) P(y_{i,*}, \mathbf{x}_i \mid Z_i = l)$$

  - Prediction
    of a single output: $$P(y_{i,j} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \sum_{l=1}^{l} P(Z_i = l) P(y_{i,j}, \mathbf{x}_i \mid Z = l)$$

  - Sharing strength: component assignments of a data point in training depend on all outputs
  - Infinite number of clusters -> Dirichlet process mixture model

## Conditional from Joint: Mixture Model

- Colors: cluster assignment (grouping of data points)
- In each cluster (grouping of rows) parameters are shared

$$\begin{pmatrix} x_1 & y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} & y_{1,5} & y_{1,6} & & y_{1,M} \\ \cdot & & & & & & & & \\ \cdot & & & & & & & & \\ x_N & y_{N,1} & y_{N,2} & y_{N,3} & y_{N,4} & y_{N,5} & y_{N,6} & & y_{N,M} \end{pmatrix}$$

## Conditional from Joint: Based on Correlation Estimates

- Empirical covariance joint model $$\frac{1}{N}([XY])([XY])^T$$

- Reduced rank joint model: $$[XY] \approx UD^{(rr)}V^T$$

- Reduced rank covariance $$C^{(rr)} = \frac{1}{N}V\left(D^{(rr)}\right)^2 V^T$$

- Prediction of a single output based on reduced rank covariance

- Sharing strength: singular vectors depend on all data

## Conditional from Joint: Additional Models

Similarly:

- memory-based
    - Collaborative filtering using cosine or Pearson similarity score
- Clustering of rows

(For applications of both approaches to collaborative filtering, see Breese, Heckerman, Kadie (1998))

- Bayesian networks / Markov networks
    - Train a joint Bayesian network / Markov network and then condition on the evidence and marginalize

- Traditional Hidden Markov Models

## Multivariate Modeling: Direct Approach

Form the conditional version of a joint model or directly formulate a conditional model and *train the conditional model directly*

$$P(y_{i,1}, \cdots, y_{i,M} \mid \mathbf{x}_i)$$

Example: CRFs

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

Log-likelihood:

$$l = -\sum_{i=1}^{N} \log Z(\mathbf{x}_i) + \sum_{i=1}^{N} \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

Prediction: e.g., by finding the most likely configuration:

$$\max_{y_{i,*}} \left[ -\log Z(\mathbf{x}_i) + \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)}) \right]$$

## Multivariate Modeling: From Marginal to Conditional

- Form a joint model of the outputs

$$P\left(y_{i,1},\cdots,y_{i,M} \mid \mathbf{w}\right)$$

one can let the parameters be dependent on input *x*

$$P(y_{i,1},\cdots,y_{i,M} \mid \mathbf{w}(\mathbf{x}_i))$$

## From Marginal to Conditional: Mixtures of Experts

- Assume a mixture model for the output variables

$$P(Z_i = l)P(y_{i,*} \mid Z = l) = \kappa_l N(y_{i,*} \mid \mu_l, \sigma^2 I)$$

- We can define a conditional model as

$$\kappa_l(\mathbf{x}_i) N(y_{i,*} \mid \mu_l(\mathbf{x}_i), \sigma^2 I)$$

With normalized “gating functions” $\kappa_l(\mathbf{x}_i)$ and “expert function” $\mu_l(\mathbf{x}_i)$

$$P(y_{i,j} \mid \mathbf{x}_i) = \sum_l \kappa_l(\mathbf{x}_i) N(y_{i,j} \mid \mu_l(\mathbf{x}_i), \sigma^2 I)$$

Tresp (2001)

## From Marginal to Conditional: Log-Linear Models

- Start with

$$P(y_{i,*}) = \frac{1}{Z} \exp \sum_{k=1}^{K} \lambda_k f_k(y_{i,*})$$

- Modeling assumption

$$\lambda_k \to \sum_l \lambda_{k,l} \phi_{k,l}(\mathbf{x}_i)$$

- Again a log-linear model with

$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_{k=1}^{K} \sum_l \lambda_{k,l} \phi_{k,l}(\mathbf{x}_i) f_k(y_{i,*})$$

- Design approach for CRFs (both input and output feature functions are indicator functions)

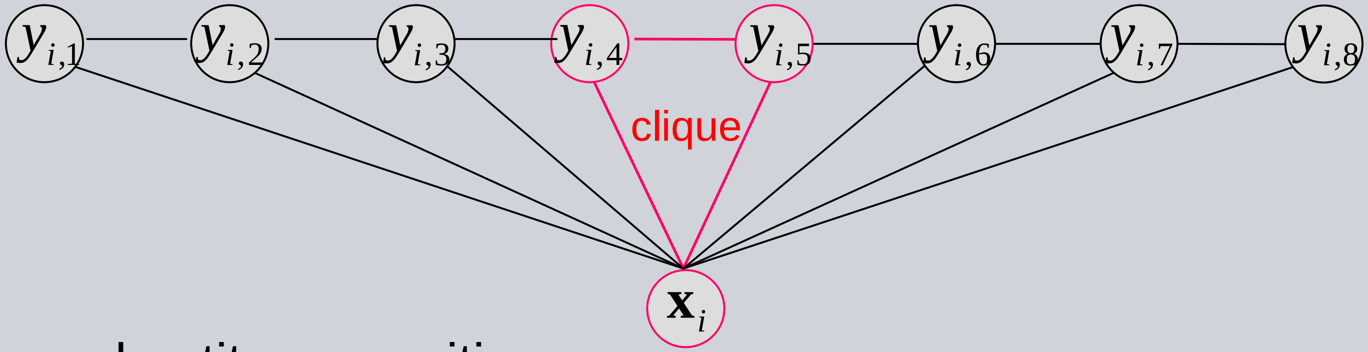
## Examples

1. High input dimensionality

2. High output dimensionality

3. High input and high output dimensionality

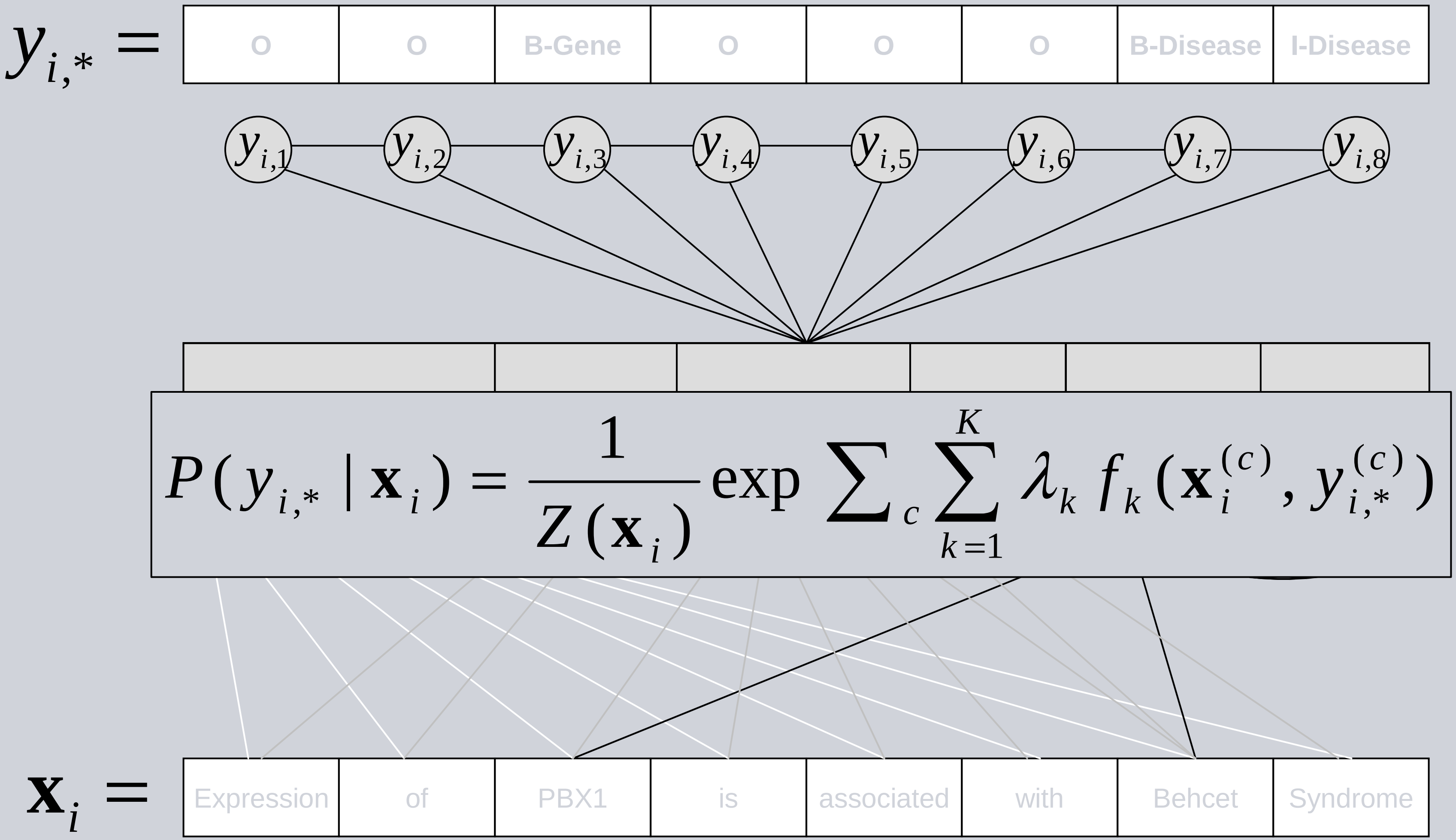
## High Input-Dimensionality




- CRFs for named entity recognition
  - Input: 50 000 and more textual features
  - Output: Sequence of maybe 10 entity classifications (with maybe 5 states for each entity: null, city, organization, person name, occupation) (*Lafferty, McCallum, Pereira, 2001*)

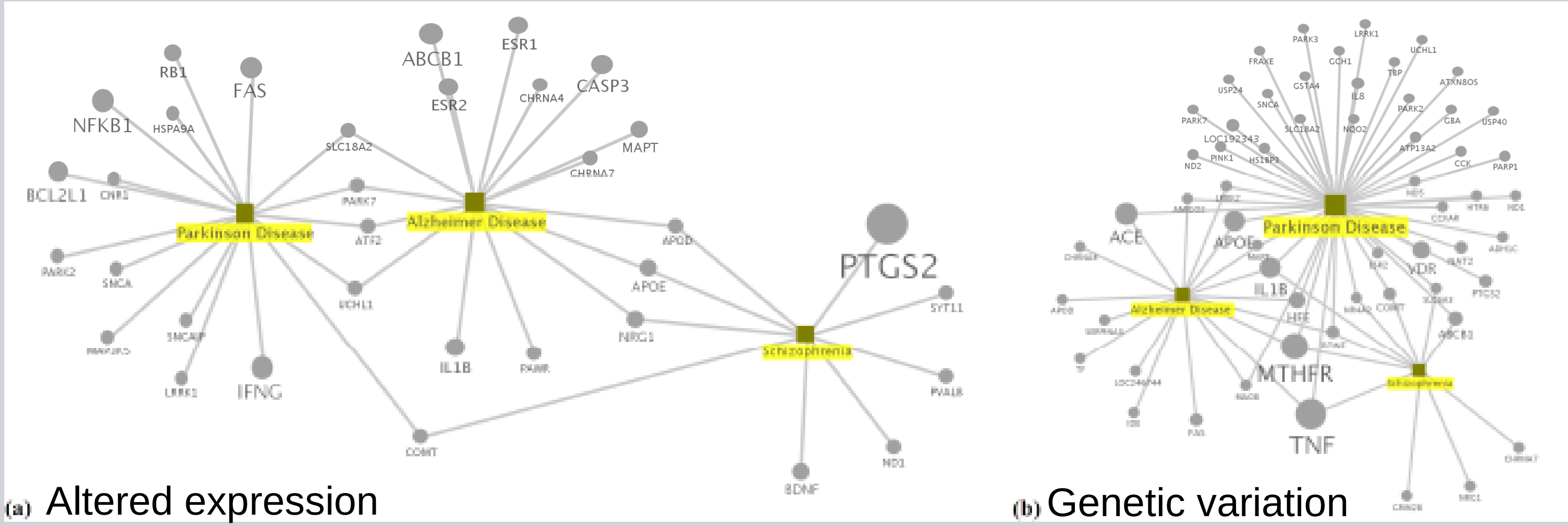
- Increasingly replacing Hidden Markov Models in many applications
- Interactions between outputs are explicitly modeled (since low-dimensional)
- Parameter sharing
- Prediction: iterative process
- Clear performance benefits from training a multivariate model!

## High Input-Dimensionality: Conditional Random Fields




$$P(y_{i,*} \mid \mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp \sum_c \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_i^{(c)}, y_{i,*}^{(c)})$$

# High Input-Dimensionality:
# CRFs for Named Entity Recognition and Relation Extraction



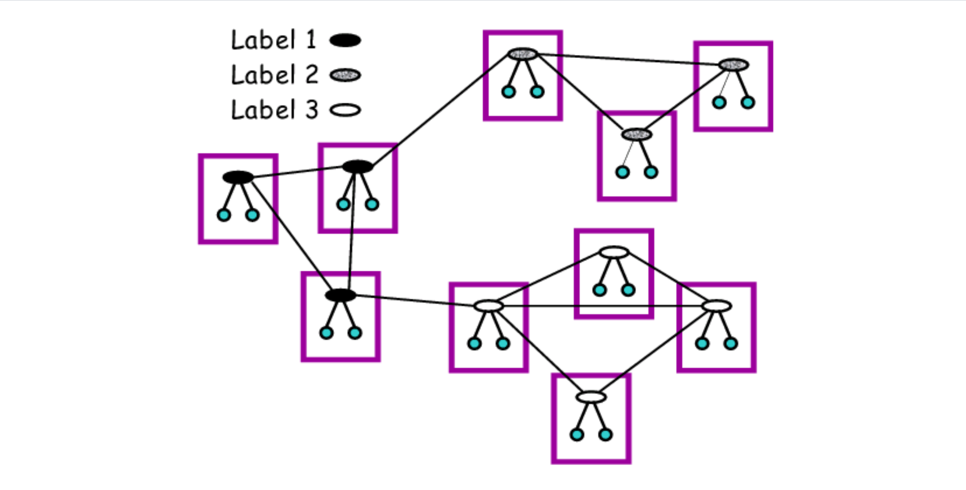

(a) Altered expression (b) Genetic variation

- Mining of the complete *GeneRIF Db* for gene-disease relations
- Disease genes according to *GeneCards Db* is 3.962 compared to 4.856 disease genes in our network (as of May 2009)

- Bundschus, Dejori, Stetter, Tresp and Kriegel (2008)

## High Input Dimensionality: Social Network Analysis

- Outputs *y* correspond to attributes of entities (wealth, social status)
- Inputs are grouped and describe properties of nodes (e.g., persons)
- Often there is only one network (one data point): learning via parameter sharing
- New challenge since  number of neighbors is varying: aggregation



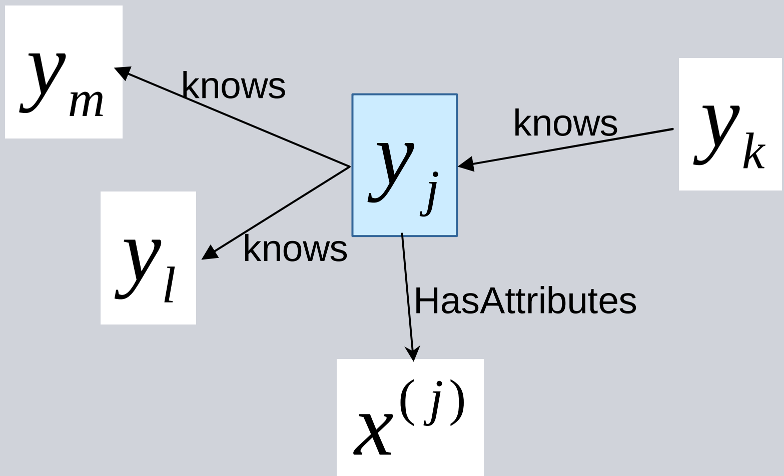

$$\boxed{\mathbf{x}^{(1)},\cdots,\mathbf{x}^{(M)}},\boxed{y_1,\ldots y_M}$$

- Chakrabarti,  Dom and Indyk (1998)
- Neville and Jensen (2000)
- Taskar, Abbeel and  Koller (2002)
- Lu and Getoor (2003)
- Neville and Jensen (2004)

## High Input Dimensionality:
## Collective Classification in Social Network Analysis

- Collective classification: a class label of an entity depends on the class label of entities to which a relationship exists ("knows") (homophily)
- Inference in the network via Gibbs sampling, relaxation labeling, iterative classification or loopy belief propagation
- Simple propagation models, e.g., Gaussian random in semi-supervised learning give very competitive results.

$$\boxed{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(M)}}, \boxed{y_1, \dots y_M}$$




Examples

- The wealth of person *j* depends on features of the person *j*, and on the wealth of the persons that person *j* knows (person *m* and person *l*) and the wealth of persons which know person *j* (person *k*)
- The classification of document *j* depends on the classes of cited and citing documents and on document attributes (hypertext classification)

## High Output-Dimensionality

- Typical examples:
  - Recommendation system
  - Order recommendation
- Input *x* often rather unimportant
  - For a new x (user) prediction is possible when some ratings are available for that user (x alone is typically inefficient)
- Often: outputs characterize relationships to objects
  - number of potential binary relationships quadratic in the number of objects
- Must be able to deal with missing outputs in training!
- Memory-based approaches, clustering, naïve Bayes, dependency networks, matrix factorization approaches (e.g., SVD-based)
- *GP-HB is typically applicable here as well*

## High Output-Dimensionality: Recommendation System

Properties of user 1

Preference for item 1

Preference for item *M*

$$\begin{pmatrix} \mathbf{x}_1 & y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} & y_{1,5} & y_{1,6} & y_{1,M} \\ \cdot & & & & & & & \\ \cdot & & & & & & & \\ \mathbf{x}_N & y_{N,1} & y_{N,2} & y_{N,3} & y_{N,4} & y_{N,5} & y_{N,6} & y_{N,M} \end{pmatrix}$$

Properties of user *N*

- Competitive solutions: matrix decomposition approaches (see Netflix competition)

## High Output-Dimensionality: Prediction of Patient Procedures

- Input: patient properties (age, sex, prime complaint, …)
- Outputs: possible procedures (367) and diagnosis (703)
- Prediction of procedure and diagnosis based on patient properties and based on procedures already administered and available diagnosis

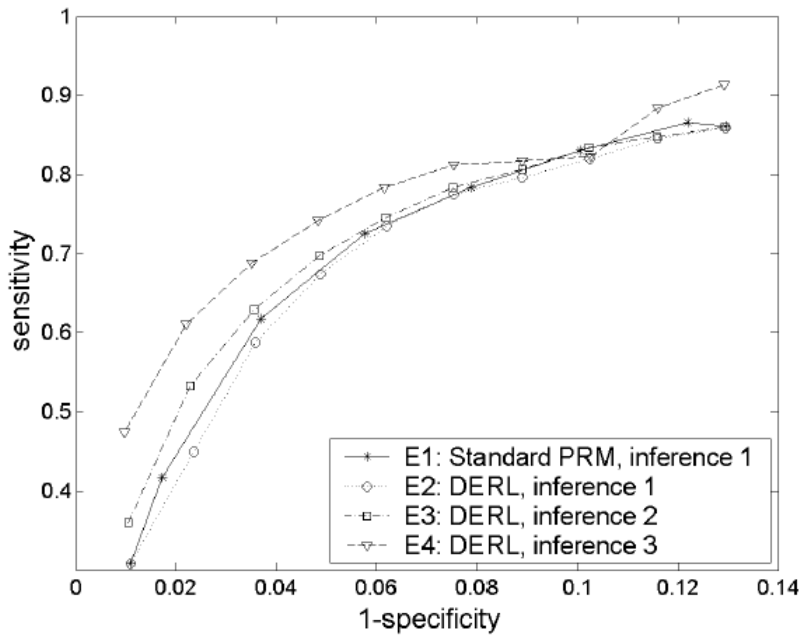



*Figure 6.* ROC curves for predicting procedures, given prime complaint *respiratory problem* and patient and hospital characteristics.

PRM, E2: no coupling between outputs
E3: only prime complaint available
E4: prime complaint and first procedure available

- Xu, Tresp, Yu, Yu and Kriegel (2005)

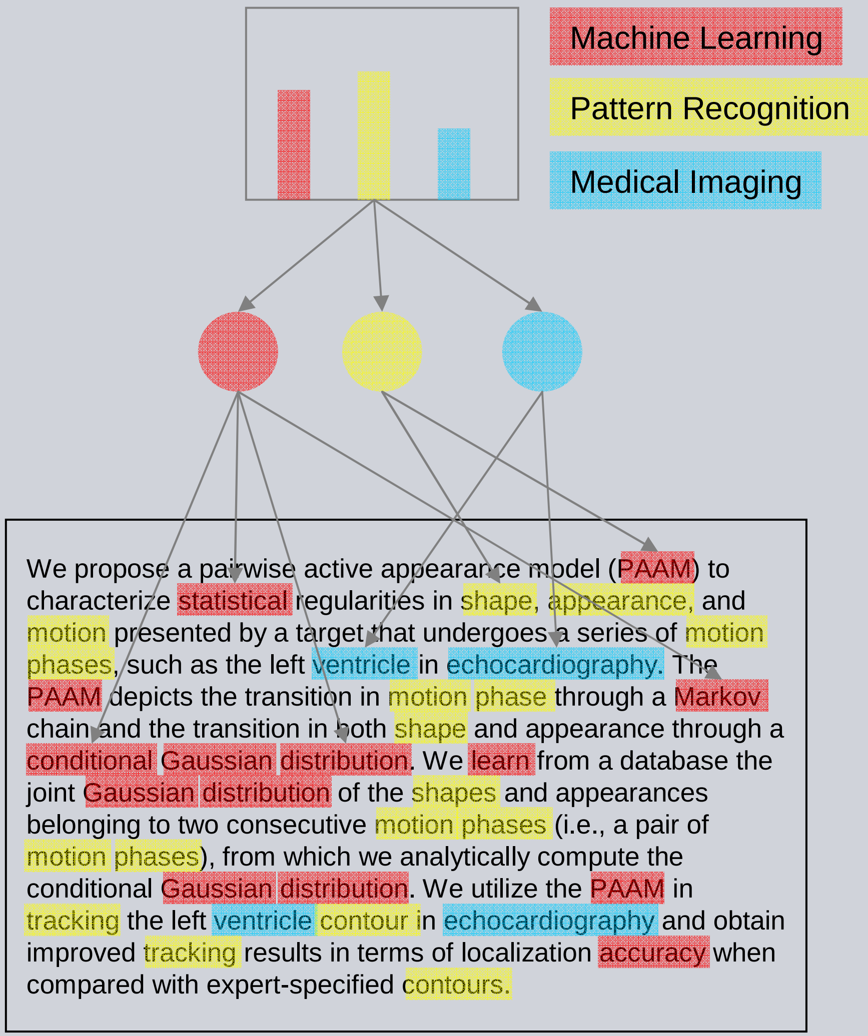
## High Dimensionality in Input and Output

- Manual, automatic and semi automatic annotation of unstructured data (text document, images, multimedia)
- Basis for the Semantic Web
- > 10000 input features (sparse)
- > 1000 possible annotations / ontological concepts (sparse)
- Different levels of annotation
  - Most important keywords: diabetes
  - Assignment of ontological concepts
    - This document covers the *metabolic disease* diabetes
  - Content: extracted statements in formalized representation
    - *This report states that the patient John Dow has a severe form of the metabolic disease diabetes*
- Worst case: Mapping from an exponential number of possible sentences to an exponential number of possible annotations
  - NLP approaches
  - Statistical approaches
    - NER and RE with CRFs
    - Text classification based on bag-of-words representation

## High Dimensionality in Input and Output: Topic Concept model

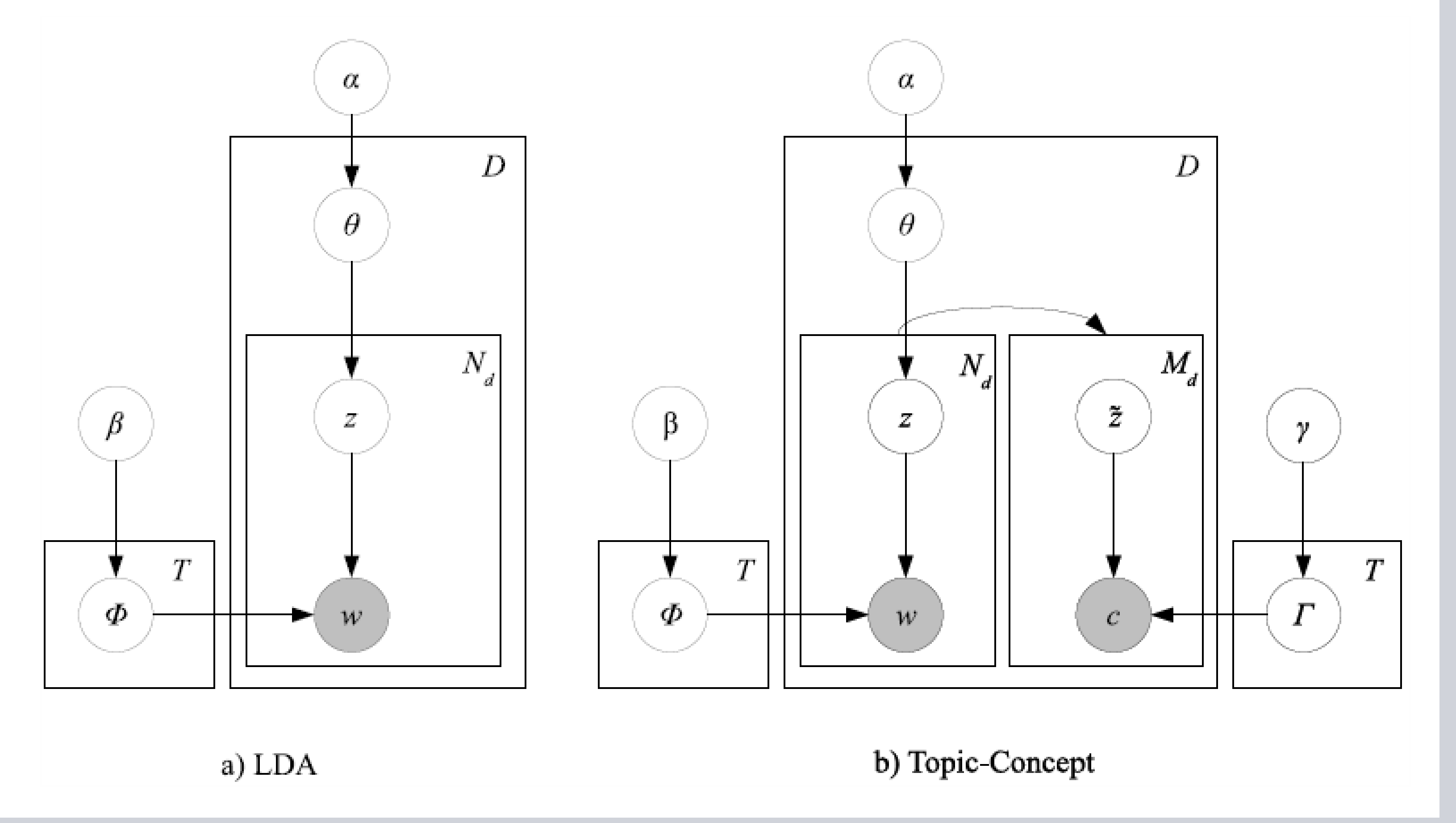
- The document is described by its topics
  - Statistically extracted and modeled using latent Dirichlet allocation (LDA)
  - Dimensionality reduction of the input
- Similarly, the annotations (simple labels) are described based on topics

- Learned mapping between both representations




Bundschus, Dejori, Yu, Tresp and Kriegel (2008)

# High Dimensionality in Input and Output: Topic Concept model (2)



## Summary:
## Multivariate Modeling / Structured Output Prediction

- Multivariate models are finding an increasing number of applications
- Since parameters often influence all outputs, learning is data efficient
- Most interesting is structured output prediction, where the constraints between outputs implied by a graphical model are exploited, which leads to a reduction in model complexity (exploitation of independencies)
- In addition,  parameter sharing leads to data efficient models
- At the same time, the dependency between input and a single output variable can be highly complex (highly complex mixture model)
- Highly active area of research (e.g., Gökhan, Hofmann, Schölkopf, Smola, Taskar, Vishwanathan, 2007, Borgwardt, Tsuda, Vishwanathan, Yan, 2008)

## Summary: Hierarchical Bayes versus Multivariate Modeling

- Hierarchical Bayes finds common patterns in different columns
  - Common representations (basis functions) to describe columns is found (linear HB, SVD)
    - Each column is represented by a parameter vector
  - In a mixture model: columns are grouped and share parameters
    - A common parameter vector is assigned to several output dimensions or columns (in the same cluster)

- In a multivariate analysis
  - SVD finds common representations of for rows
    - Each row is represented by a parameter vector
  - In a mixture model: columns are grouped and share parameters
    - A common parameter vector is assigned to several data points or input dimensions (in the same cluster)

# V. Link Prediction / Relationship prediction

## From Attributes to Relations

- So far we mostly focused on the situation where the outputs *y* correspond to *attributes* of one entity (one sentence, one social network) or, sometimes equivalently, to attributes of many entities (many words, many members of a social network)
  - In a social network analysis: relationships were assumed known but some object attributes were assumed unknown

- Now we want to study applications were the *relationships* between objects are partially unknown:
  - In a social network analysis: relationships between entities (knows, friendOf) are unknown

- Getoor, Friedman, Koller and Taskar (2002)
  - Taskar, Wong, Abbeel and Koller (2003)

## Predicting a Single Relationship Type

- We will be concerned with the situation where only one relationship type is concerned
- In this case a matrix representation is appropriate where $y_{i,j}$

  describes the relationship between row entity *i* and column entity *j*

- A new aspect: attributes for both input entities and output entities are available!
- Symmetrical representation
- Note that, as before, the whole network of interlinked entities should be considered to represent a single data point, thus the matrix does not represent i.i.d samples

- In the spirit of the previous discussion we will focus on generalizations of mixture models and of SVD approaches

## Hierarchical Bayesian versus Multivariate Mixture Models

- Hierarchical Bayes:
  - In a mixture model: columns are grouped and share parameters
    - A common parameter vector is assigned to several output dimensions or columns (in the same cluster)

- In a multivariate analysis
  - In a mixture model: columns are grouped and share parameters
    - A common parameter vector is assigned to several data points or input dimensions (in the same cluster)

- Now
  - A mixture model for both rows and columns

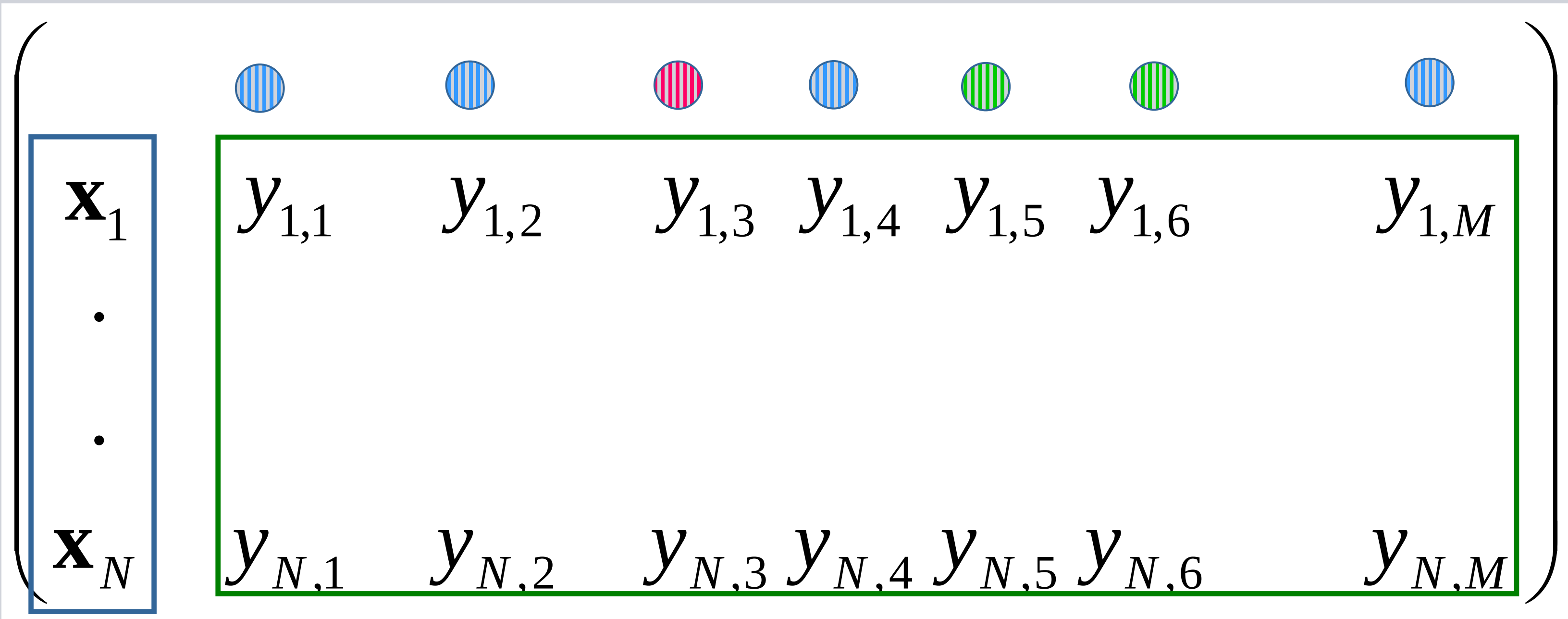
## Recall: Mixture Analysis of Multivariate data

- Colors: cluster assignment (grouping of data points)

$$
\begin{pmatrix}
\mathbf{x}_1 & y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} & y_{1,5} & y_{1,6} & y_{1,M} \\
\cdot & & & & & & & \\
\cdot & & & & & & & \\
\mathbf{x}_N & y_{N,1} & y_{N,2} & y_{N,3} & y_{N,4} & y_{N,5} & y_{N,6} & y_{N,M}
\end{pmatrix}
$$

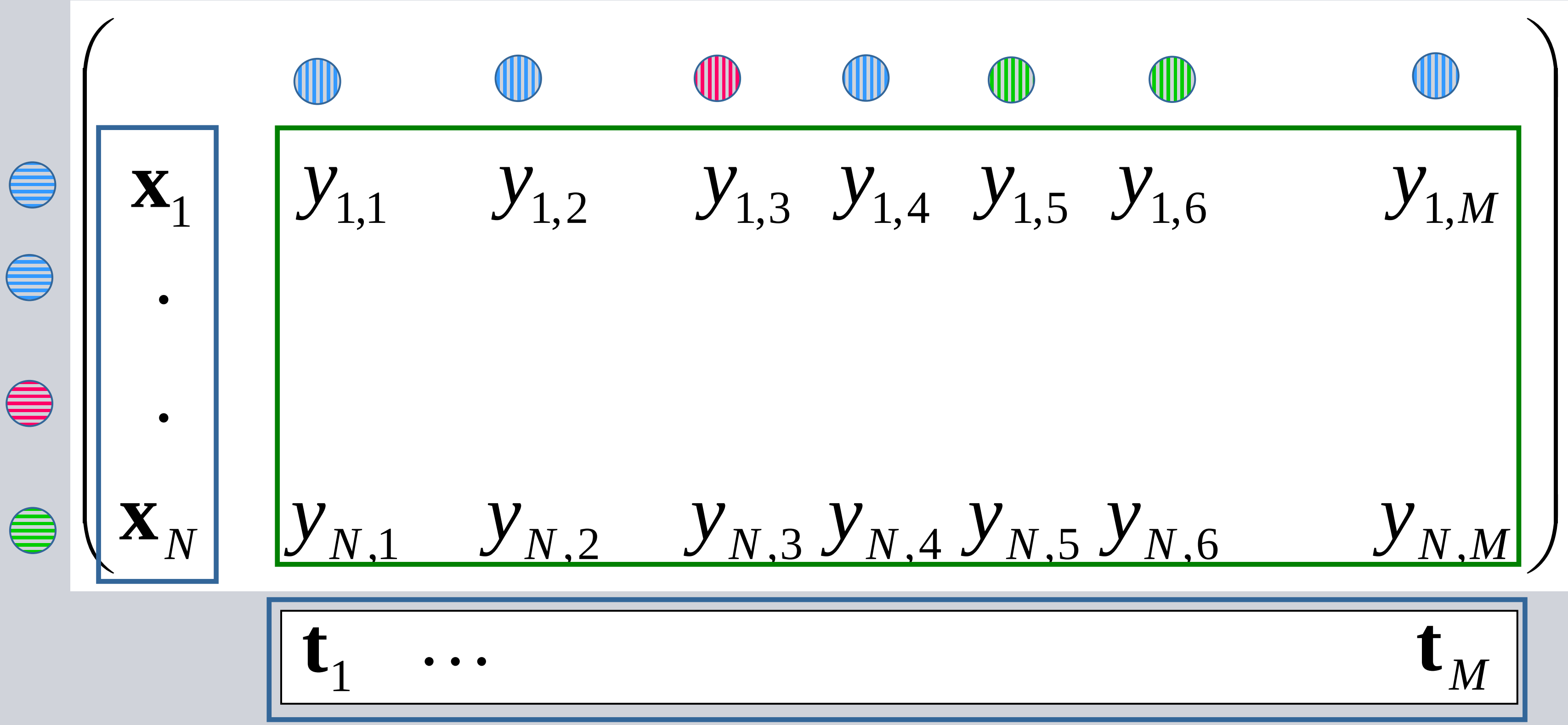
## Recall: Mixture Analysis of Outputs

- Dirichlet process mixture models (Nonparametric Hierarchical Bayes)
- Colors: cluster assignment (grouping of outputs/functions, not data points)



## Mixture Analysis of Input Objects and Output Objects

- Colors: cluster assignment (grouping of outputs/functions, not data points)
- ***t***: attributes of output objects
- Infinite Hidden Relational Model (IHRM, Xu et al. 2006, Kemp et al. 2006)




- Note: not really one matrix anymore: a relational data base would require at least two tables

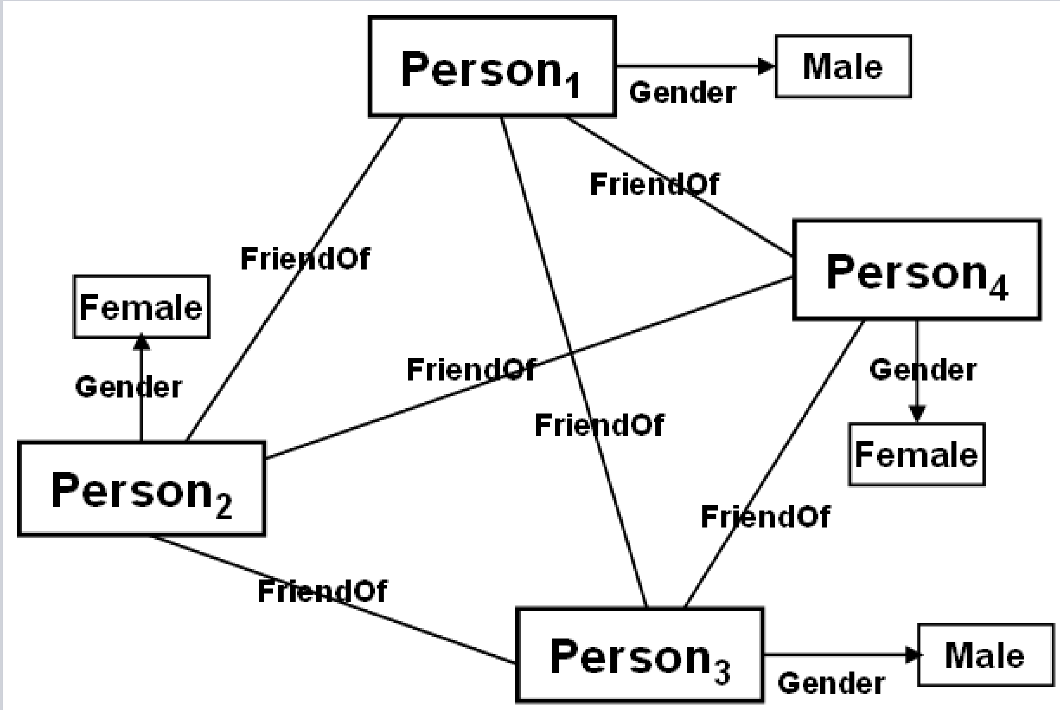
# Example: Social Network

**To introduce the IHRM we use a social network example**

- Some persons are known to be friends
- Persons can either be male or female
- Can we predict friendship?

**Graphical representation:**

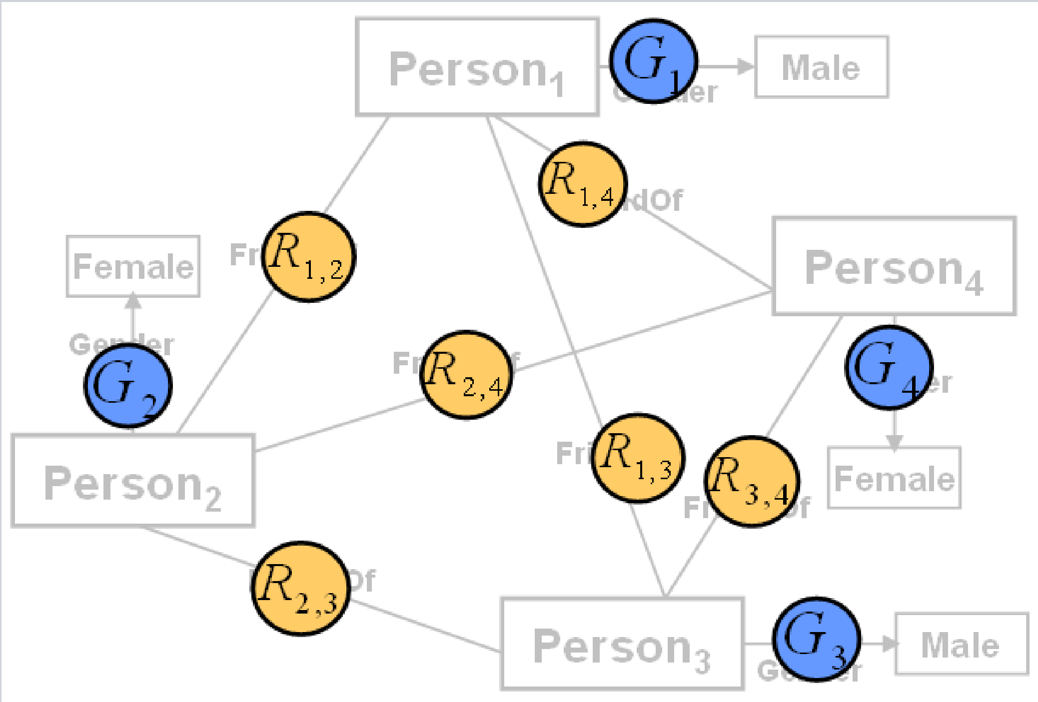
- Sociogram
- Entity-relationship graph
- RDF-Graph




- Xu, Tresp, Yu, Yu (2008)

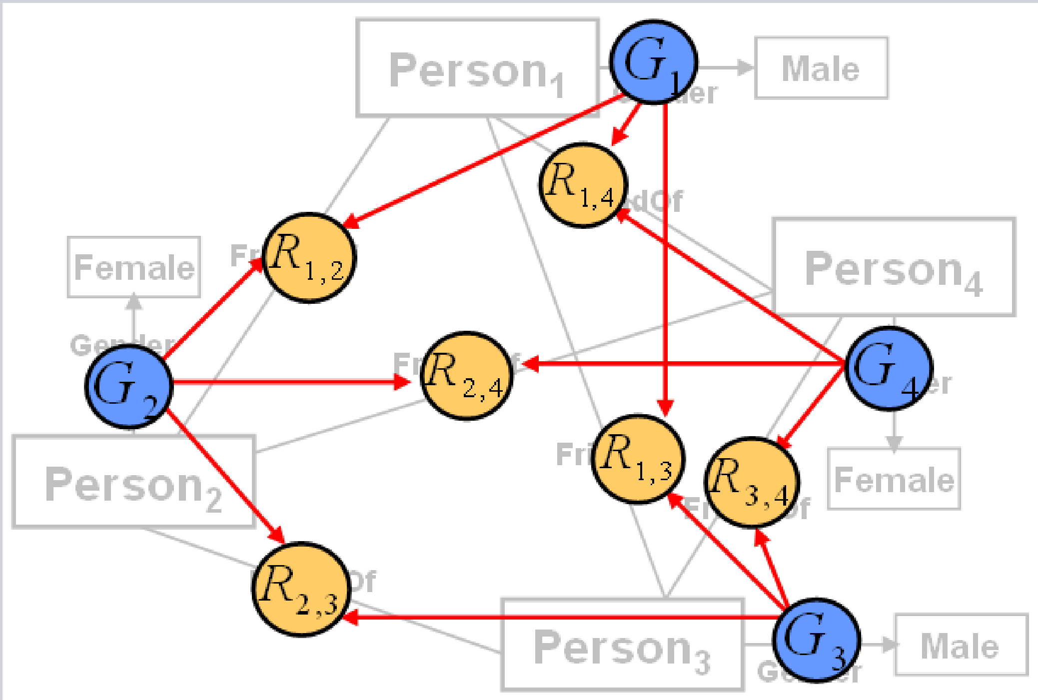
# Relational Graph and Random Variables

- Each random variable stands for the truth value of a statement



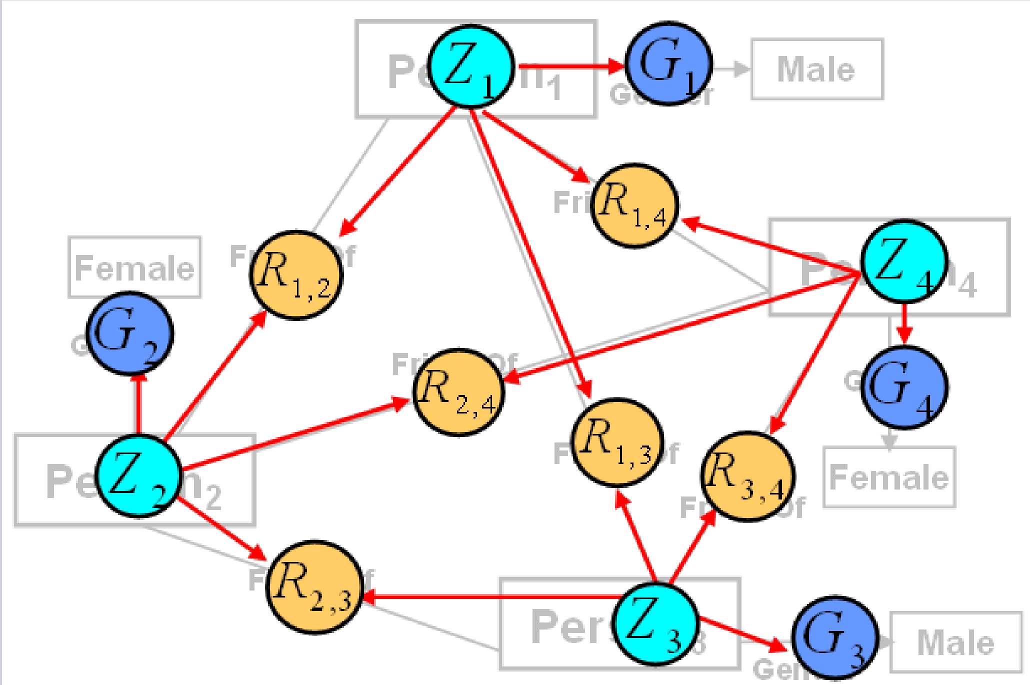
# A Possible Ground Bayesian Networks

- The red directed arcs indicate direct probabilistic dependencies
- Here we assume that friendship can be predicted by the attributes (gender)
- We obtain a ground Bayesian network
- Problems:
  - Only local dependencies; no global propagation of information
  - No collaborative effect (exploiting friendship patterns)



# Hidden Relational Model (HRM)

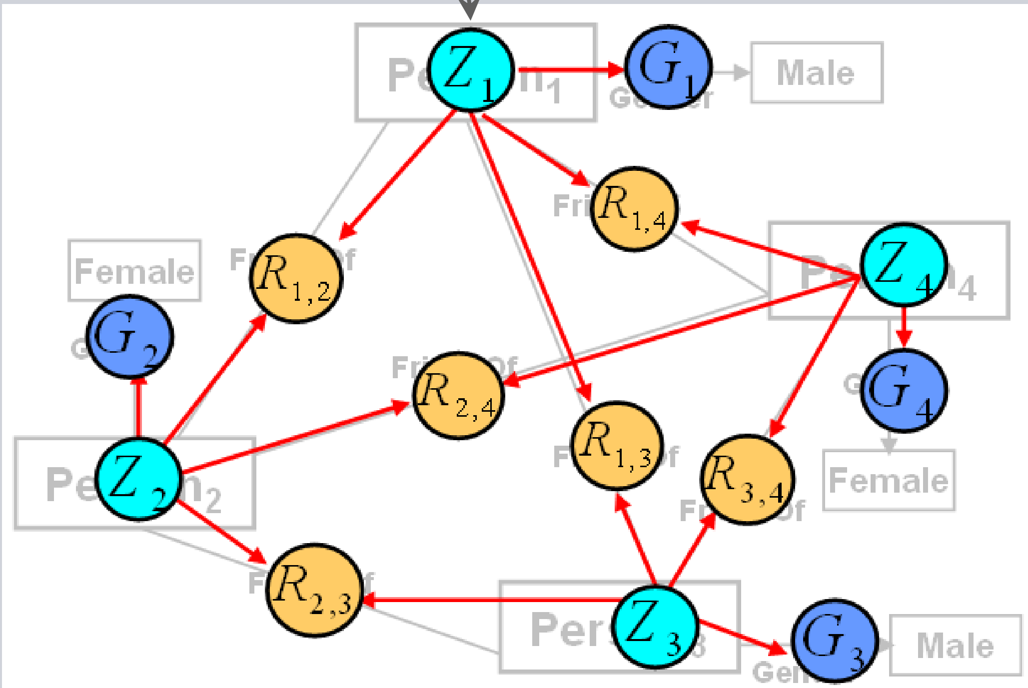
- In the HRM we introduce a latent (cluster) variable for each object
- The latent variable is the parent of all nodes involving statements that include the object
- The latent variable represents the unknown information that would be sufficient to predict links (latent attributes)
- The state of the latent variable depends on
  - The attributes (gender)
  - The links an object is involved in and the states of the latent variables of the objects involved in the link.
- Identification of *roles of actors*



# *Infinite* Hidden Relational Model (IHRM)

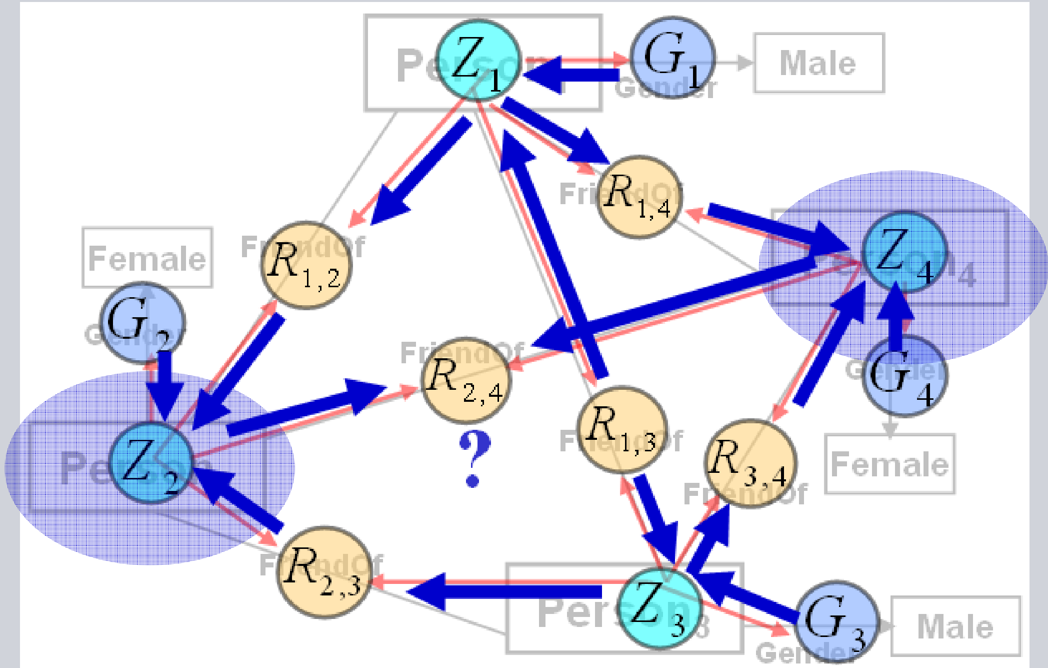
- In the IHRM the number of states in each latent variable is infinite
- We achieve a nonparametric *hierarchical Bayesian model* in form of a *Dirichlet process mixture model*
- A property of the Dirichlet process mixture models: During inference, the number of hidden states is adapted to the data in a self organized way
  - Important if different object types are involved

Infinitely many states!



# Information Propagation in IHRM

- Information propagates along "relational paths"
- All known information propagates to the relation of interest via hidden variables of the involved objects



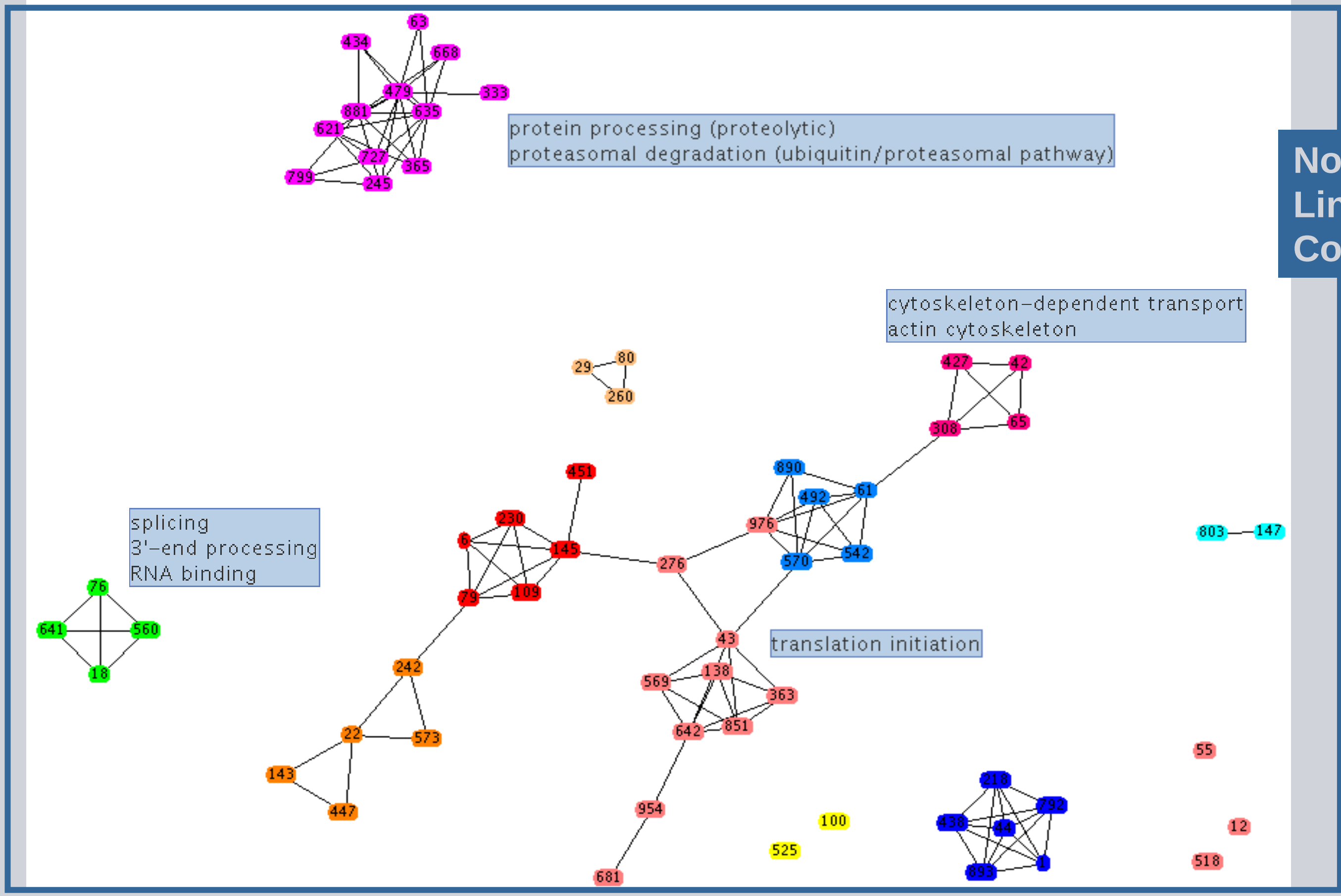
# Advantages of the IHRM

- Easy to apply without any extensive structural learning
  - Structural learning in Statistical Relational Learning can be quite demanding
- Information can flow through the network of latent variables and have a global effect
  - Collaborative effect (exploiting friendship patterns)
- The ground network is guaranteed to have no directed loops
- Clustering in relational domain (multi-relational clustering)
  - Analysis of clustering structure based on relational information
  - Each entity class can learn its optimal number of clusters
- No computationally-expensive feature construction (aggregation) and no global normalization

# Inference/Learning in the IHRM

- A full Bayesian approach for learning and inference in the IHRM is feasible (and even practical) using Gibbs sampling
- Mean-field approximations
- Gibbs sampling simulates the model (i.e., samples from parameters and variables) conditioned on the observations

# IHRM Model for Modeling Protein Interactions




**Node: gene**
**Link: interaction**
**Color: cluster.**

- Reckow and Tresp (2008)

## Stochastic Relational Model:
## Multi-task Learning using Task-specific features

- Similar architecture but the latent components consist of $K$ continuous variables generated from Gaussian processes

$$v_1(t_1)\cdots v_K(t_K) \qquad \cdots \qquad v_K(t_1)\cdots v_K(t_K)$$

$$\begin{array}{l}
u_1(x_1)\cdots u_K(x_1) \\
\cdots \\
u_1(x_N)\cdots u_K(x_N)
\end{array}
\begin{pmatrix}
\mathbf{x}_1 & y_{1,1} & & y_{1,M} \\
\cdot & & & \\
\cdot & & & \\
\mathbf{x}_N & y_{N,1} & & y_{N,M}
\end{pmatrix}$$

$$\mathbf{t}_1 \quad \cdots \quad \mathbf{t}_M$$

$$f(\mathbf{x}_i, \mathbf{t}_j) = \frac{1}{\sqrt{K}} \sum_{k=1}^{K} u_k(\mathbf{x}_i) v_k(\mathbf{t}_j)$$

$$F = \frac{1}{\sqrt{K}} U V^T$$

- Yu, Chu, Yu, Tresp, Xu (2006)
- Yu and Chu (2007)
- Bonilla, Chai, and Williams (2007)
- Zhu, Yu, Gong (2008)

- Yu, Chu, Yu, Tresp, Xu (2006)
- Zhu, Yu, Gong (2008)

## Stochastic Relational Model
## Multi-task Learning using Task-specific features (2)

$$
\begin{array}{cc}
 & \begin{array}{ccc} & v_1(t_1)\cdots v_K(t_K) \quad \cdots \quad v_K(t_1)\cdots v_K(t_K) & \end{array} \\
\begin{array}{c} u_1(x_1)\cdots u_K(x_1) \\ \cdots \\ u_1(x_N)\cdots u_K(x_N) \end{array} & \begin{pmatrix} \mathbf{x}_1 & y_{1,1} & y_{1,M} \\ \cdot & & \\ \cdot & & \\ \mathbf{x}_N & y_{N,1} & y_{N,M} \end{pmatrix} \\
 & \begin{array}{ccc} \mathbf{t}_1 & \cdots & \mathbf{t}_M \end{array}
\end{array}
$$

- Given two prior kernel functions based on row & column features:

$$\Omega_0(\mathbf{x}_i, \mathbf{x}_{i'}),\ \Sigma_0(\mathbf{t}_j, \mathbf{t}_{j'})$$

SRM defines a distribution for the rank-k relational function *f(x,t)*

- Generalization of matrix factorization using attributes in a hierarchical Bayesian framework

- Efficient Gibbs sampler is developed to do full Bayesian inference (code is available online)
- Applied to Netflix data (480189x17770), gave excellent performance
- In the limit k->infinity, *f(x,t)* follows a Gaussian process .

$$GP(0, \Omega \otimes \Sigma)$$

$$Cov(f_{ij}, f_{i',j'}) = \Omega(x_i, x_{i'})\Sigma(t_j, t_{j'})$$

## Summary: Link/Relationship Prediction

- The IHRM is a natural generalization of mixture models and of nonparametric Bayesian models to relational domains: both attributes and relationships can be predicted
- The  SRM is a natural generalization of PCA to a relational domain

- Both the IHRM and the SRM can be generalized to domains with multiple relation types (i.e., multiple tables)

## What we Did Not Cover: Max Margin Approaches

These approaches are related to CRFs but optimize a margin-based cost function

$$\forall i,\ \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i : \langle w, \delta \mathbf{f}_i(\mathbf{y}) \rangle > 0,$$

$$\delta \mathbf{f}_i(\mathbf{y}) \equiv \mathbf{f}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{f}(\mathbf{x}_i, \mathbf{y})$$

- No normalization function
- Potentially: advantages in terms of accuracy and tunability to specific lass functions

- Taskar, Guestrin and Koller (2004)
- Tsochantaridis, Hofmann, Joachims and Altun (2004)
- Tsochantaridis, Joachims, Hofmann, and Altun (2006)
- Rousu, Saunders, Szedmak and Shawe-Taylor (2006)
- Rousu, Saunders, Szedmak and Shawe-Taylor (2007)
- Altun, Hofmann and Tsochantaridis (2007)
- Weston, Bakir, Bousquet, Mann, Noble and Schölkopf (2007)

## What we Did Not Cover: Neural Networks

- The very first Neural Networks had multiple outputs (e.g., Nettalk)
- There are Neural Networks for multi-task learning and for structured prediction
  - E.g., papers by Yann LeCun, Yoshua Bengio

- Also *ICML 2009 Workshop on Learning Feature Hierarchies. Organizers: Kai Yu, Ruslan Salakhutdinov, Yann LeCun, Geoff Hinton, Yoshua Bengio*

## Conclusions

- We have shown that in many situations it makes sense to predict *M* outputs than to only predict one

- Hierarchical Bayes and Projection Methods are applicable when the functional form of the dependencies between input and each output is similar and is known
- Hierarchical Bayes is more flexible since it can easily deal with nonlinear models and with missing outputs
- Nonparametric Hierarchical Bayes (Gaussian processes, Dirichlet process mixture models) provide flexible model classes

- Multivariate modeling exploits dependencies between inputs and outputs but also dependencies in between outputs
- Often all outputs are sensitive to a parameter and learning is data efficient

- Structures Output Prediction exploits both prior knowledge about the structural independencies between outputs and parameter sharing
- An important model class concerns conditional random fields (CRFs)
- At the same time, the dependency between input and a single output variable can be highly complex

- In Link Prediction / Relationship Prediction the outputs model the relationships between entities

## Acknowledgements

Collaborators:

- Anton Schwaighofer, Microsoft Research
- Shipeng Yu, Siemens Healthcare
- Zhao Xu, FhG IAIS
- Markus Bundschus, Ludwig-Maximilians University Munich
- Christoph Lippert, Max Planck Institute for Biological Cybernetics

# References: General


- T. Hastie, R. Tibshirani, J. Friedman. The elements of statistical learning. Springer 2001
- A. Gelman, J. B. Carlin, H. S. Stern, D. B. Rubin. Bayesian Data Analysis, 2nd edition. Chapman, 2003

## References: Hierarchical Bayes / Multitask


- R. Caruana. Learning many related tasks at the same time with backpropagation. NIPS 1995
- S. Thrun. Is Learning the n-th Thing Any Easier Than Learning the First? NIPS 1996
- A. Gelman, J. B. Carlin, H. S. Stern, D. B. Rubin. Bayesian Data Analysis, 2nd edition. Chapman 2003
- D. M. Blei, A. Y. Ng, M. I. Jordan. Latent Dirichlet Allocation. JMLR 2003
- A. Schwaighofer, V. Tresp, K. Yu. Learning gaussian process kernels via hierarchical bayes. NIPS 2004
- N. D. Lawrence, J. C. Platt. Learning to learn with the informative vector machine. ICML 2004
- K. Yu, V. Tresp, A. Schwaighofer. Learning gaussian processes from multiple tasks. ICML 2005
- J. Zhang. Sparsity Models for Multi-task Learning. NIPS Workshop on Inductive Transfer 2005
- J. Zhang, Z. Ghahramani, Y. Yang. Learning Multiple Related Tasks using Latent Independent Component Analysis. NIPS 2005
- A. Argyriou, T. Evgeniou, M. Pontil. Multi-task Feature Learning. NIPS 2006
- T. Evgeniou, C.A. Micchelli, M. Pontil. Learning multiple tasks with kernel methods. JMLR 2006
- R. Raina, A. Y. Ng, D. Koller. Constructing informative priors using transfer learning. ICML 2006
- S. Yu, K. Yu, V. Tresp. Collaborative ordinal regression. ICML 2006
- J. Rousu, C. Saunders, S. Szedmak, J. Shawe-Taylor. Kernel-based Learning of Hierarchical Multilabel Classification Models. JLMR 2006
- Z. Barutcuoglu, R. Schapire, O. Troyanskaya. Hierarchical multi-label prediction of gene function. Bioinformatics 22, 2006
- B. Krishnapuram, S. Yu, O. Yakhnenko, R. B. Rao, L. Carin. NIPS Workshop: Cost Sensitive Learning. 2008
- K. Yu, S. Zhu, J. Lafferty, Y. Gong. Fast Nonparametric Matrix Factorization for Large-scale Collaborative Filtering. SIGIR 2009

# References: Projection Methods


- J. Shawe-Taylor, N. Cristianini. Kernel Methods for Pattern Analysis. Cambridge University Press 2004
- R. K. Ando, T. Zhang. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. JMLR 2005
- K. Yu, S. Yu, V. Tresp. Multi-output regularized projection. IEEE CVPR 2005
- K. Yu, S. Yu, V. Tresp. Multi-label informed latent semantic indexing. SIGIR 2005
- S. Yu, K. Yu, V. Tresp, H.-P. Kriegel, M. Wu. Supervised probabilistic principal component analysis. KDD 2006
- S. Yu, K. Yu, V. Tresp, H.-P. Kriegel. Multi-output regularized feature projection. IEEE TKDE 2006
- D. Hardoon, G. Leen, S. Kaski, J. Shawe-Taylor. NIPS Workshop: Learning from Multiple Sources. 2008

## References: Dirichlet Process Mixture Models for Multitask Learning


- K. Yu, W.-Y. Ma, V. Tresp, Z. Xu, X. He, H. J. Zhang, H.-P. Kriegel. Knowing a tree from the forest: Art image retrieval using a society of profiles. ACM Multimedia 2003
- Kai Yu, A. Schwaighofer, V. Tresp, W.-Y. Ma, H. J.Zhang. Collaborative ensemble learning: Combining collaborative and content-based information filtering via hierarchical bayes. UAI 2003
- V. Tresp, K. Yu. An introduction to nonparametric hierarchical Bayesian modeling with a focus on multi-agent learning. In Proceedings of the Hamilton Summer School on Switching and Learning in Feedback Systems. 2004
- K. Yu, V. Tresp, S. Yu. A nonparametric hierarchical bayesian framework for information filtering. SIGIR 2004
- K. Yu, S. Yu, V. Tresp. Dirichlet enhanced latent semantic analysis. AISTAT 2005.
- M. I. Jordan. Dirichlet Processes, Chinese Restaurant Processes and All. Tutorial at NIPS 2005
- V. Tresp. Dirichlet processes and nonparametric bayesian modelling. Tutorial at the Machine Learning Summer School 2006
- Y. Xue, X. Liao, L. Carin, B. Krishnapuram. Multi-task learning for classification with Dirichlet process priors. JMLR 2007

# References: Multivariate Models and Structured Outputs (1)


- S. Chakrabarti, S. Dom, P. Indyk. Enhanced hypertext categorization using hyperlinks. SIGMOD 1998
- J. S. Breese, D. Heckerman, C. M. Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. UAI 1998
- J. Neville, D. Jensen. Iterative classification in relational data. AAAI 2000
- V. Tresp. Mixtures of gaussian processes. NIPS 2001
- J. Lafferty, A. McCallum, F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. ICML 2001
- B. Taskar, P. Abbeel, D. Koller. Discriminative probabilistic models for relational data. UAI 2002
- Q. Lu, L. Getoor. Link-based classification. ICML 2003
- J. Neville, D. Jensen. Dependency networks for relational data. ICDM 2004
- B. Taskar, C. Guestrin, D. Koller. Max-Margin Markov Networks. NIPS 2004
- I. Tsochantaridis, T. Hofmann, T. Joachims, Y. Altun. Support vector machine learning for interdependent and structured output spaces. ICML 2004
- X. Zhu. Semi-supervised learning literature survey. TR University of Wisconsin 2005
- B. Taskar, V. Chatalbashev, D. Koller, C. Guestrin. Learning Structured Prediction Models: A Large Margin Approach. Tutorial at ICML 2005
- I. Tsochantaridis, T. Joachims, T. Hofmann, T. Y. Altun. Large margin methods for structured and interdependent output variables. JMLR 2006

## References: Multivariate Models and Structured Outputs (2)


- H. B. Gökhan, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar S. V. N. Vishwanathan (editors). Predicting Structured Data. MIT press 2007.
- Y. Altun, T. Hofmann, I. Tsochantaridis: Support Vector Machine Learning for Interdependent and Structured Output Spaces. Chapter 5 in Predicting Structured Data, MIT Press 2007
- J. Rousu, C. Saunders, S. Szedmak, J. Shawe-Taylor. Efficient algorithms for Max-Margin Structured Classification. Chapter 6 in Predicting Structured Data, MIT Press 2007
- J. Weston, G. Bakir, O. Bousquet, T. Mann, W.S. Noble, B. Schölkopf. Joint Kernel Maps. Chapter 4 in Predicting Structured Data, MIT Press 2007
- K. Borgwardt, K. Tsuda, S. V. N. Vishwanathan, X. Yan. NIPS Workshop: Structured Input - Structured Output 2008
- M. Bundschus, M. Dejori, M. Stetter, V. Tresp, H.-P. Kriegel. Extraction of semantic biomedical relations from text using conditional random fields. BMC Bioinformatics 2008
- M. Bundschus, M. Dejori, S. Yu, V. Tresp, H.-P. Kriegel. Statistical modeling of medical indexing processes for biomedical knowledge information discovery from text. BIOKDD 2008

## References: Link / Relationship Prediction


- L. Getoor, N. Friedman, D. Koller, B. Taskar. Learning Probabilistic Models of Link Structure. JMLR 2002
- B. Taskar , M.-F. Wong , P. Abbeel, D. Koller. Link Prediction in Relational Data. NIPS 2003
- Z. Xu, V. Tresp, K. Yu, S. Yu, H.-P. Kriegel. Dirichlet enhanced relational learning. ICML 2005
- Z. Xu, V. Tresp, K. Yu, H.-P. Kriegel. Infinite hidden relational models. UAI 2006
- C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, N. Ueda. Learning Systems of Concepts with an Infinite Relational Model. National Conference on Artificial Intelligence 2006
- K. Yu, W. Chu, S. Yu, V. Tresp, Z. Xu. Stochastic relational models for discriminative link prediction. NIPS 2006.
- E.V. Bonilla, K.M.V. Chai, C.K.I. Williams. Multi-task Gaussian process prediction. NIPS-07
- S. Zhu, K. Yu, Y. Gong. Stochastic Relational Models for Large-scale Dyadic Data using MCMC. NIPS 2008
- Z. Xu, V. Tresp, S. Yu, K. Yu. Nonparametric relational learning for social network analysis. SNA-KDD 2008
- C. Lippert, S. Weber, Y. Huang, V. Tresp, M. Schubert, H.-P. Kriegel. Relation-Prediction in Multi-Relational Domains using Matrix-Factorization. NIPS Workshop: Structured Input - Structured Output 2008
- S. Reckow, V. Tresp. Integrating ontological prior knowledge into relational learning. NIPS Workshop: Structured Input - Structured Output 2008