On some optimisation problems in structured pattern recognition

Tran The Truyen
Department of Computing
Curtin University of Technology
http://truyen.vietlabs.com

Content

- Some tasks in pattern recognition
- Graphical models
- Conditional Random Fields
- Boltzmann Machines
- Conclusions

Content

- Some tasks in pattern recognition
- Graphical models
- Conditional Random Fields
- Boltzmann Machines
- Conclusions

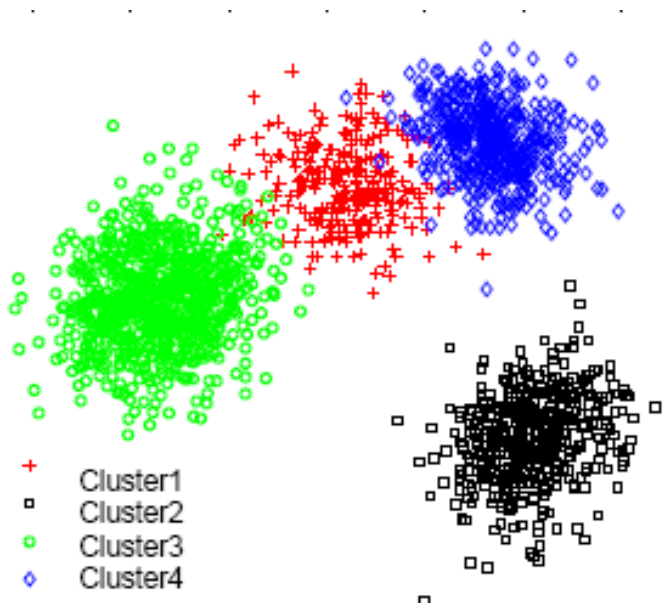
Some tasks in pattern recognition

- **Classification/regression** (a.k.a. *supervised learning*): given a set of pairs $\{z^{(i)}, x^{(i)}\}_{i=1}^{K} \in \mathcal{Z} \times \mathcal{X}$, estimate a functional f so that, for any $j > K$
  - in classification, we obtain $x^{(j)} = \arg\max_x f(x, z^{(j)})$ with high probability.
  - in regression, we obtain $\left\| x^{(j)} - f(z^{(j)}) \right\| < \epsilon$ with high probability.
- **Clustering/dimensionality reduction** (a.k.a. *unsupervised/manifold learning*): no x is given, try to estimate some *hidden variable* h that is associated with z.

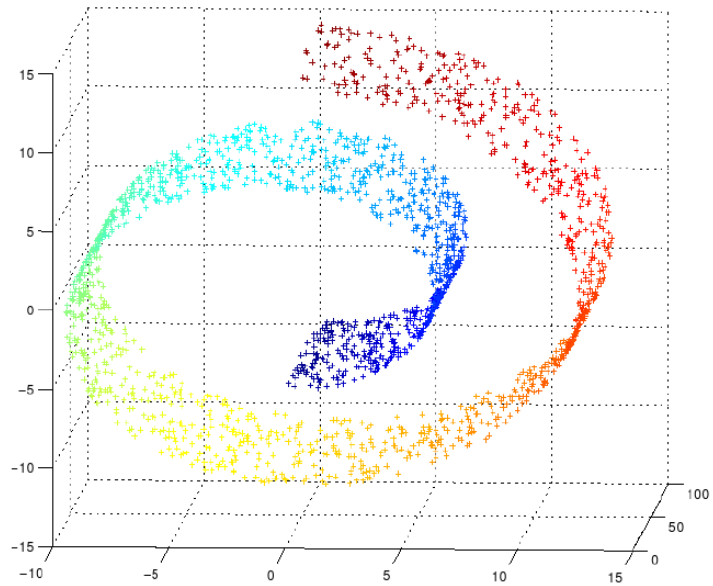
Classification example: handwritten digit recognition

- z is a vector of visual descriptions (known as *features*)
- x is a discrete label $x \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.
- Many thousands of training pairs are needed to get up to 99% recognition accuracy.

Clustering: group data points in a meaningful way



Dimensionality reduction: discover intrinsic dimensions



Supervised learning: minimising some regularised (convex) *empirical* risk

- Assuming the functional f is parameterised by $\mathbf{w} \in \mathbb{R}^N$, which is associated with the feature vector $\mathbf{g}(x, z)$.
  - Often in classification we are interested in the linear form: $f(z) = \arg\max_{x \in \mathcal{X}} \mathbf{w}^\top \mathbf{g}(x, z)$

$$\begin{aligned} \mathcal{R}(\mathbf{w}; K, \lambda) &= \frac{1}{K} \sum_{i=1}^{K} R(x^{(i)}, f(z^{(i)}; \mathbf{w})) + \lambda \Omega(\|\mathbf{w}\|) \\ \hat{\mathbf{w}} &= \arg\min_{\mathbf{w}} \mathcal{R}(\mathbf{w}; K, \lambda) \end{aligned}$$

where $R(a, b)$ is the risk function, measuring the divergence of a, b; $\Omega(.)$ is some (convex) function and $\lambda > 0$ specifies the penalty *strength*.

Supervised learning: minimising some regularised (convex) *empirical* risk (cont.)

- This may be just a standard optimisation problem that efficient methods already exist.
- But things are getting complicated in the real-world, e.g:
  - The parameter dimensionality can be extremely large, e.g. $N \sim 10^9$,
  - The number of training pairs can be big, e.g. $K \sim 10^7$, but the number of *activated* features per pair is usually small,
  - The cost of acquiring the labels $\{x^{(i)}\}_{i=1}^{K}$ can be high,
  - Training data come one-by-one, requiring constant re-estimation of $\mathbf{w}$
  - Many features are just irrelevant or noisy (corresponding $w_k = 0$)
  - What if the model fails in unseen data? (need bounding in errors)

Content

- Some tasks in pattern recognition
- Graphical models
- Conditional Random Fields
- Boltzmann Machines
- Conclusions

Graphical models

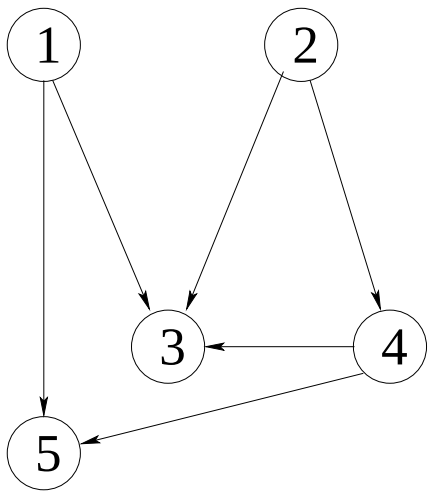
- It is the common language for many previously separate models:
  - *Ising Models (1920s)*
  - *Markov Random Fields (1980s)*
  - *Bayesian Networks (1980s)*
  - *Hidden Markov Models (1960s)*
  - *Boltzmann Machines (1980s)*
  - *Kalman Filters (1960s)*
  - *Many neural network variants (1990s)*
- and some recent developments
  - *Factor Graphs (2001)*
  - *Relational Markov Networks (2002)*
  - *Markov Logic-Networks (2006)*

Graphical models

- Are the mix between *graph theory* and *probability theory*
- A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ encodes the dependencies between variables (represented by nodes)
- The dependency strength between local set of nodes, indexed by c, is encoded in the potential functions $\phi_c(x_c) > 0$
  - The *directed* edge $e \in \mathcal{E}$ shows the dependency direction (parent-child), as in Bayesian Networks,
  - The *undirected* edge $e \in \mathcal{E}$ shows the correlations, as in Markov Random Fields

Bayesian Networks

- Potentials are probabilities: $\phi_c(x_c) = P(x_i|x_{c\setminus i})$
- The joint probability of the network: $P(x) = \prod_c P(x_i|x_{c\setminus i})$

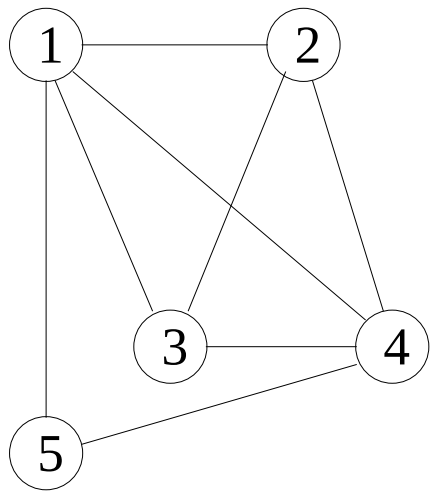


Markov Random Fields

- The joint probability of the network:

$$P(x) = \frac{1}{Z}\prod_c \phi_c(x_c)$$

where $Z = \sum_x \prod_c \phi_c(x_c)$.



Content

- Some tasks in pattern recognition
- Graphical models
- Conditional Random Fields
- Boltzmann Machines
- Conclusions

Conditional Random Fields

- Invented in 2001 by Lafferty et al
- The idea is simple: if we have the pair (x, z) but z is always known and not of the patterns we are looking for, then $P(x, z) = P(x|z)P(z)$
- We only need to model $P(x|z)$ by a Markov Random Field, which is depending on z:

$$P(x|z) = \frac{1}{Z(z)} \prod_c \phi_c(x_c, z)$$

where $Z(z) = \sum_x \prod_c \phi_c(x_c, z)$.

Maximum likelihood learning

- Assuming *log-linear* parameterisation:
  $\phi_c(x_c, z) = \exp\{\mathbf{w}^\top \mathbf{g}(x_c, z)\}$
- In standard supervised learning, the risk is

$$\begin{aligned} R(x^{(i)}, z^{(i)}) &= -\log P(x^{(i)}|z^{(i)}) \\ &= -\mathbf{w}^\top \mathbf{g}(x_c^{(i)}, z^{(i)}) + \log Z(z^{(i)}) \end{aligned}$$

- $\log Z(z^{(i)})$ is convex in $\mathbf{w}$, so is $R(x^{(i)}, z^{(i)})$
- Thus, learning is a convex optimisation problem
- So what?

Maximum likelihood learning (cont)

- The main problem is the log-partition function $\log Z(z)$
  - Generally it is *NP-hard* to compute, except for problems where the graph $\mathcal{G}$ is a tree or a chain
  - So its gradient

$$\begin{aligned} \nabla \log Z(z) &= \sum_x P(x|z) \sum_c \mathbf{g}(x_c, z) \\ &= \sum_c \sum_{x_c} P(x_c|z) \mathbf{g}(x_c, z) \end{aligned}$$

- In short, for learning, we need to compute (*infer*) $\log Z(z)$ and $P(x_c|z) = \sum_{x \neg c} P(x_c, x_{x \neg c}|z)$

Approximate inference by minimising Free energies

- Assuming some *physical interpretation* of the models, drop notation z for simplicity
- The entropy $\mathbf{H}[P] = -\sum_x P(x) \log P(x)$
- The Gibbs free energy $\mathbf{F}[P] = -\log Z$. It is generally known that physical systems evolve to achieve minimum $\mathbf{F}[P]$.
- ***Bethe free energy*** is an approximation to Gibbs
- It was proved in 2001 that minimising Bethe free energy turns out to yield the **Belief-Propagation** algorithm by Julian Pearl, the farther of graphical models
- Soon after, ***Kikuchi free energy*** was also studied, yielding better approximation.

Approximate inference by minimising Kullback-Leibler Divergence

- Assume that we can approximate $P(x)$ by some $Q(x)$ which is much easier to deal with
- The natural goal is to push $Q(x)$ to get closer to $P(x)$ as possible
- So Kullback-Leibler Divergence is a good objective function

$$\mathbf{D}(Q||P) = \sum_x Q(x) \log \frac{Q(x)}{P(x)}$$

- When $Q(x) = \prod_{i \in \mathcal{V}} Q_i(x_i)$, we obtain the well-known **Mean-Field** equation

$$Q_i(x_i) = \frac{1}{Z_i} \exp\{\log \phi_i(x_i) + \sum_{j|(i,j) \in \mathcal{E}} Q_j(x_j) \log \phi_{ij}(x_i, x_j)\}$$

where Z_i ensures that $\sum_{x_i} Q_i(x_i) = 1$.

Prediction or finding the maximiser of the distribution

- Given an input z, we are interested in knowing

$$\begin{aligned}\hat{x} &= \arg\max_x P(x|z) \\ &= \arg\max_x \left(\frac{1}{Z(z)} \prod_c \phi_c(x_c, z) \right) \\ &= \arg\min_x \left(\sum_c -\log \phi_c(x_c, z) \right)\end{aligned}$$

  The term $\sum_c -\log \phi_c(x_c, z)$ is often called the **energy**.

- If clique c is pairwise, i.e. $c = (i, j)$, we have

  $\hat{x} = \arg\min_x \left(\sum_{(i,j)\in\mathcal{E}} -\log \phi_{ij}(x_i, x_j, z) \right)$

- This is a NP-hard problem!

Approximate minimisation of energy using Min-Sum algorithm

- This is also known as *Max-Product, Belief-Propagation* or *Message Passing*, sometimes *Dynamic Programming* or *Viterbi algorithm*
- Assume pairwise cliques, let $\omega_{ij}(x_i, x_j, z) = -\log \phi_{ij}(x_i, x_j, z)$
- We maintain a set of messages passing along all edges $e \in \mathcal{E}$

$$\mu_{j \to i}(x_i) = \min_{x_j} \left\{ \omega_{ij}(x_i, x_j, z) \sum_{k \neq i | (k,j) \in \mathcal{E}} \mu_{k \to j}(x_j) \right\}$$

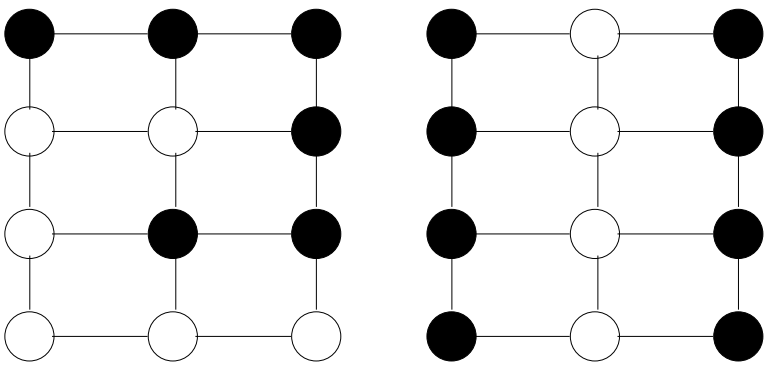
- Finally, the optimum is found by

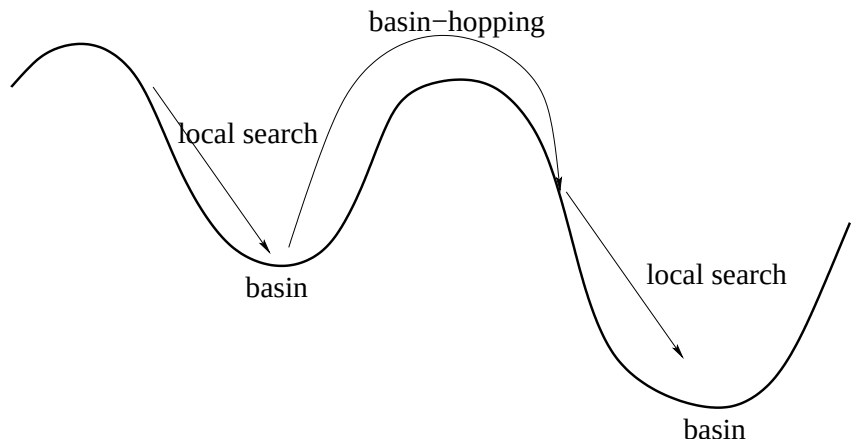
$$\hat{x}_i = \arg\min_{x_i} \left\{ \sum_{j | (i,j) \in \mathcal{E}} \mu_{j \to i}(x_i) \right\}$$

- It can be proved that Min-Sum finds global minimum if the graph is a tree or a chain.

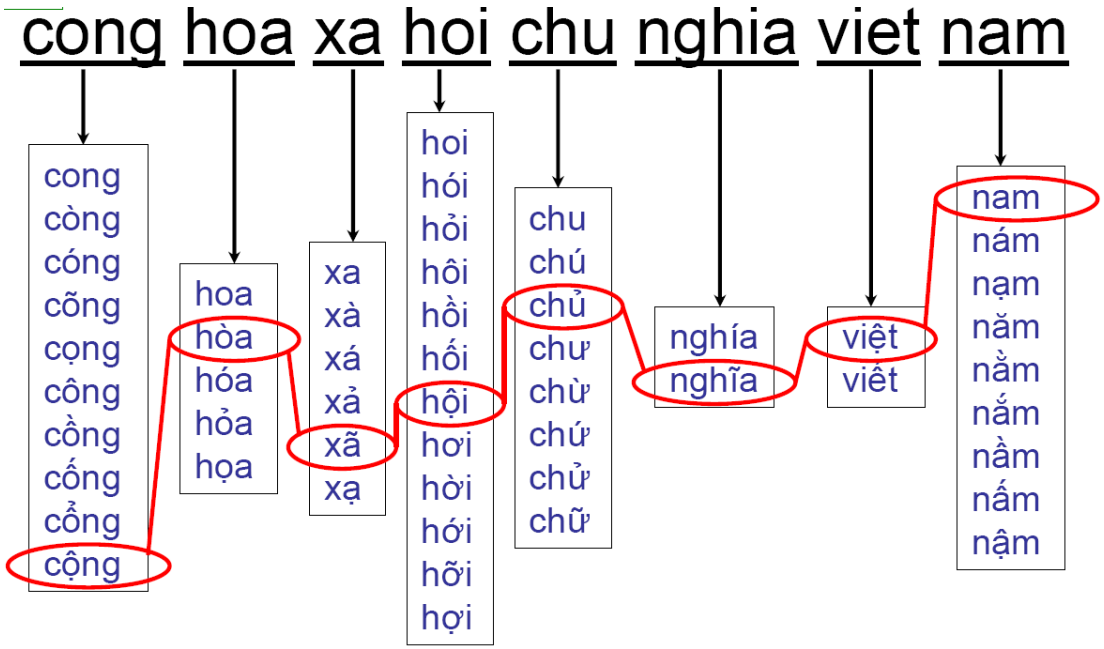
Exploiting local structures for Dynamic Programming (Truyen et al, 2007)

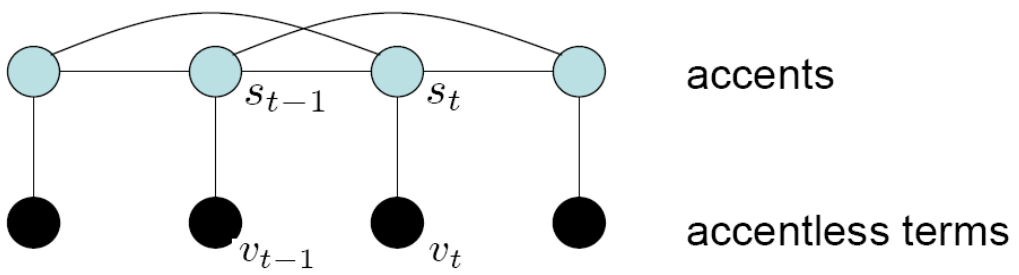
- It is known that for trees or chains, Dynamic Programming requires only two passes through all edges
- The idea is to seek for trees or chains embedded in the graph to improve the (pseudo) likelihood or the energy
- In the case of energy minimisation, this can be combined with Basin-Hopping (a.k.a. Iterated Local Search)





Applications: Vietnamese accent restoration (Truyen et al, 2008)





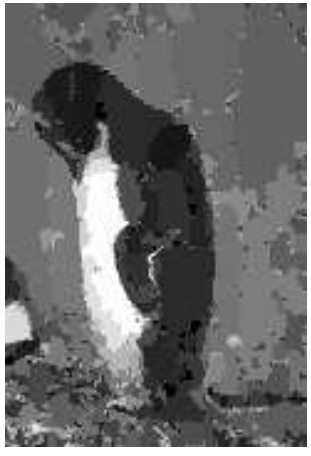
- 3×10^6 parameters
- 0.5×10^6 sentences
- Stochastic gradient descent
- 94.3% term-accuracy
- Demo: http://vietlabs.com

Applications of energy minimisation: Image denoising

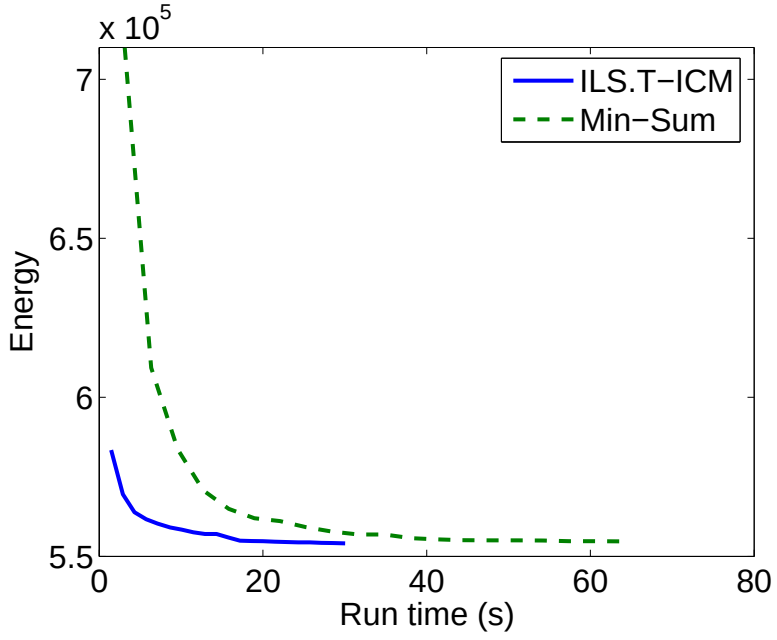
Noisy penguin

Local method (ICM)

Min-Sum



Tree-based ICM



Run-time comparison between Min-Sum vs Tree-based ICM

- ICM: Iterated Local Mode
- ILS: Iterated Local Search
- (Truyen et al, 2007)

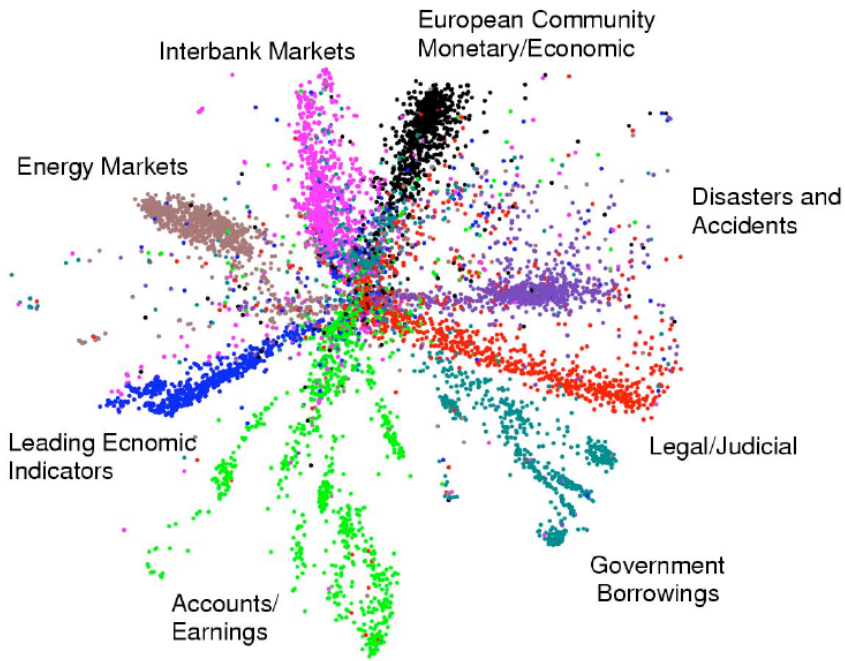
Content

- Some tasks in pattern recognition
- Graphical models
- Conditional Random Fields
- Boltzmann Machines
- Conclusions

Boltzmann Machines

- Invented in the 1980s, now gaining much attention!
- Are Markov Random Fields with some hidden/visible variables
- Are powerful models in discovering
  - Low dimensionality of the data
  - Clusters

Projecting documents onto 2-D (Hinton et al, 2007)



Learning using incomplete likelihood

- Let $x = (h, v)$ where h is the hidden subset of variables, and v the visible
- For estimating the parameters $\mathbf{w}$, we maximise the incomplete log-likelihood

$$\begin{aligned} \log P(v|\mathbf{w}) &= \log \sum_h P(h, v|\mathbf{w}) \\ &= \log \sum_h \prod_c \phi_c(h_c, v_c) - \log Z \\ &= \log Z(v) - \log Z \end{aligned}$$

As before, $\log Z(v)$ and $\log Z$ are convex in $\mathbf{w}$, so it is of D.C. form.

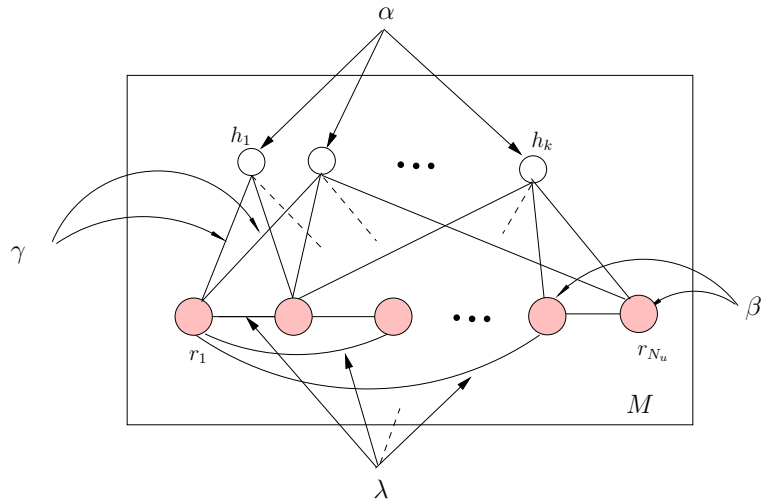
Learning using incomplete likelihood (cont)

- There exists an algorithm known as **DCA** to find maximiser of $\log P(v|\mathbf{w})$
  - At each step we solve a convex problem, which is presumably easier the original
  - It turns out that **DCA** and the well-known **EM** algorithm (Dempster et al, 1977) are **the same** in this log-linear case.
  - Due to lack of collaboration between fields, DCA was reinvented in 2001 in the name of **CCCP** (Concave-Convex Procedure)
- However, we found empirically that there is little numerical advantage using DCA/EM/CCCP
  - Generic numerical methods like Limited-memory BFGS or Conjugate Gradients are often sufficient

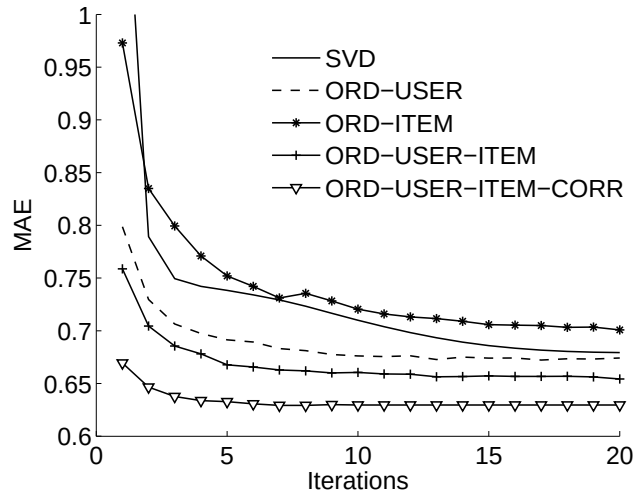
Application: Movie recommendation (Truyen et al, 2009, submitted)

- Based on ratings that users already gave to old movies, we predict if an user may like a new movie
- This is called Collaborative Filtering
  - Amazon is most well-known example
- Currently the Netflix competition with $1mil prize
  - 1.7×10^4 movies
  - 0.5×10^6 users
  - 1.0×10^8 ratings

Modelling: one Boltzmann machine per user



Prediction errors as a function of learning time



Content

- Some tasks in pattern recognition
- Graphical models
- Conditional Random Fields
- Boltzmann Machines
- Conclusions

Conclusions

- Unstructured and structured pattern recognition tries to uncover meaningful patterns of the real-world data
- It requires a great deal of optimisation techniques
- It poses many new challenges in optimisation
  - Non-convex, non-smooth
  - Ultra high dimensionality, massive computing power needed to evaluate objective function
  - Very high level of sparsity, noise removal capacity
  - Evaluation errors and stochastic gradients
  - High cost to acquire enough information for objective function evaluation
- Often, approximate methods are the only possibility
  - We still need some theoretical bounds on the tightness, currently only few have been found
- We need mathematicians!

Shameless advertisement: PhD positions available

- Lab: IMPCA, Department of Computing, Curtin University of Technology [See impca.cs.curtin.edu.au]
- Full scholarships
- Theoretical and applied sub-areas:
  - Computer vision
  - Graphical models
  - Multimedia
  - Social networks modelling and analysis
  - Compressive sensing
- See more on: truyen.vietlabs.com/scholarship.html