Exponential Family Bipartite Matching

Tibério Caetano

(with James Petterson, Julian McAuley and Jin Yu)

Statistical Machine Learning Group
NICTA and Australian National University
Canberra, Australia
http://tiberiocaetano.com

AML Workshop, NIPS 2008
Outline

- The Problem
- The Model
- Model Estimation
- Experiments
The Problem
Overview
Assumptions

- Each couple $ij$ has a pairwise happiness score $c_{ij}$
- Monogamy is enforced, and no person can be unmatched
- Goal is to maximize overall happiness
Applications

- Image Matching (Taskar 2004, Caetano et al. 2007)
Applications

- Machine Translation (Taskar et al. 2005)
Goal: Find a *perfect match* in a complete bipartite graph:
Solution is a permutation $y : \{1, \ldots, n\} \mapsto \{1, \ldots, n\}$

Aggregate pairwise happiness of a collective marriage $y$:

$$\sum_i c_{iy(i)}$$

Best collective marriage $y^*$:

$$y^* = \arg\max_y \sum_i c_{iy(i)}$$

Maximum-Weight Perfect Bipartite Matching Problem. Also called Assignment Problem.

Exactly solvable in $O(n^3)$ (e.g. Hungarian algorithm)
The Model
Our Contribution

- We relax the assumption that we know the scores $c_{ij}$.

- In reality we measure edge features $x_{ij} = (x_{ij}^1, \ldots, x_{ij}^d)$

- Instead we parameterize the edge features and perform Maximum-Likelihood Estimation of the parameters.

- $c_{ij} = f(x_{ij}; \theta)$
Probability of a match given a graph:

\[ p(y|x; \theta) = \exp(\langle \phi(x, y), \theta \rangle - g(x; \theta)) \]

\[ y = \text{match}, \ x = \text{graph} \]

Most likely match: \( y^* = \arg\max_y \langle \phi(x, y), \theta \rangle \)

IDEA: construct \( \phi(x, y) \) such that \( y^* \) agrees with best match

\[ \langle \phi(x, y), \theta \rangle = \sum_i c_{iy(i)} \]
Maximum-Likelihood Marriage Estimator

\[ \langle \phi(x, y), \theta \rangle = \sum_i c_{iy(i)} \] suggests:

\[ \phi(x, y) = \sum_i \psi_{iy(i)} \]

\[ c_{iy(i)} = \langle \psi_{iy(i)}, \theta \rangle \]

I.e. the pairwise happiness is now parameterized. I.e. the goal will be to learn which features of people are more relevant to make them happier collectively (not individually!!)
Maximum-Likelihood Marriage Estimation

\[ \ell(Y|X; \theta) = \sum_{n=1}^{N} \left( g(x^n; \theta) - \langle \phi(x^n, y^n), \theta \rangle \right) \]

Partition function:

\[ \exp(g) = \sum_y \exp \sum_{i=1}^{N} c_{iy(i)} = \sum_y \prod_{i=1}^{N} \exp c_{iy(i)} := B_{iy(i)} \]

Permanent: \#P-complete
Maximum-Likelihood Marriage Estimation

- For learning we need to do gradient descent in $\ell(\theta)$:

$$\nabla_\theta \ell(X, Y; \theta) = \sum_{n=1}^{N} \nabla_\theta g(x^n; \theta) - \phi(x^n, y^n)$$

- BAD NEWS

$$\nabla_\theta g(x; \theta) = \sum_y \phi(x, y)p(y|x; \theta) = E_{y \sim p(y|x;\theta)}[\phi(x, y)]$$
Model Estimation
GOOD NEWS

A sampler of perfect matches has been recently proposed (Huber & Law, SODA ’08), which is $O(n^4 \log n)$ to generate a sample. This sampler is EXACT.

Previous fastest sampler (Jerrum, Sinclair & Vigoda J. ACM ’04) was $O(n^7 \log^4 n)$ and was INEXACT (truncated Markov Chain). This was IMPRACTICAL.
General Idea of Sampler

- Construct an upper bound on the partition function

- Use self-reducibility of permutations to generate successive upper bounds of partial partition functions

- Use sequence of upper bounds to generate an accept-reject algorithm
General Idea of Sampler

\[ \Omega = \mathcal{X}, |\Omega| = 6 \]

\[ \Omega_1 = \{ x : x_{1,2} = 1 \}, |\Omega_1| = 2 \]

\[ \Omega_2 = \{ x : x_{1,2} = 1, x_{2,3} = 1 \}, |\Omega_2| = 1 \]
The Sampler

Let $x$ be a sample obtained after the algorithm is run. Then:

$$p(\Omega) = \sum_{y \in \Omega} w(y) = Z$$
$$p(\Omega_1) = \sum_{y \in \Omega_1} w(y)$$
$$p(\Omega_2) = \sum_{y \in \Omega_2} w(y) = w(x)$$

Its probability is:

$$\frac{U(\Omega_1)}{U(\Omega)} \frac{U(\Omega_2)}{U(\Omega_1)} = \frac{U(\Omega_2)}{U(\Omega)} = \frac{w(x)}{U(\Omega)}$$

But the probability of accepting is $\frac{Z}{U(\Omega)}$

So $p(x) = \frac{w(x)/U(\Omega)}{Z/U(\Omega)} = \frac{w(x)}{Z} \Rightarrow$ EXACT SAMPLER
The Upper Bound

\( h(r) = \begin{cases} 
    r + (1/2) \ln(r) + e - 1, & r \geq 1 \\
    1 + (e - 1)r, & r \in [0, 1]
\end{cases} \) (1.5)

Our bound is as follows:

**Theorem 1.2.** Let \( A \) be a matrix with entries in \([0, 1]\). Let \( r(i) \) be the sum of the \( i \)th row of the matrix.

\( \text{per}(A) \leq \prod_{i=1}^{n} \frac{h(r(i))}{e} \). (1.6)

(Huber & Law, SODA 2008)
Monte Carlo

1. Why is this good?

2. From samples $y_i \sim p(y|x; \theta)$, approximate expectation:

$$E_{y \sim p(y|x; \theta)}[\phi(x, y)] \approx \frac{1}{m} \sum_{i=1}^{m} \phi(x, y_i)$$
Given the approximated gradient, we perform a quasi-Newton optimization to obtain the Maximum-Likelihood Estimate (we actually use a prior and do MAP estimate).
Experiments
Matching with vs without learning
Matching with vs without learning
Matching with vs without learning
Ranking
Ranking

- Ranking

Google search results for "nips" query
Can be formulated as a Matching Problem (Le et al, 2007)
Data: $q_n, \{d^i_n\}_i, \{s^i_n\}_i$

$q_n$: $n^{th}$ Query

$\{d^i_n\}_i$: Set of documents retrieved by query

$\{s^i_n\}_i$: Labeled scores for documents retrieved by query.

Typically $s^i_n \in \{0, \ldots, N\}$ where 0 = ‘bad’ and $N$ = ‘excellent’.
The Problem  The Model  Model Estimation  Experiments

Ranking

- \( c_{ij} = s(d^i, q)f(y) \)

- Where \( f \) is monotonically decreasing

- Therefore

\[
\arg\max_y \sum_i c_{iy(i)} = \text{argsort}_y (s(d^y(1), q), \ldots, s(d^y(\text{last}), q))
\]

- \( \langle v, w(y) \rangle \) is obtained by sorting \( v \) according to \( y \) if \( w \) is non-increasing
LETOR Dataset (TD2003)

TD2003 (30 runs)
Ranking

LETOR Dataset (TD2004)
Ranking

LETOR Dataset (OHSUMED)
Final remarks

- We use a linear model, with competitive results.

- Best competitors are highly non-linear models.

- We can instead use kernels and obtain a non-linear exponential family model, and it is still a convex problem.
Thanks

Thanks