Incorporating Domain Knowledge in Matching Problems via Harmonic Analysis

Deepti Pachauri
(joint work with Maxwell Collins, Risi Kondor, Vikas Singh)

University of Wisconsin-Madison
University of Chicago

International Conference on Machine Learning 2012
Matching Problems are Ubiquitous

Photo Tourism
Matching Problems are Ubiquitous

Shape Matching
Matching Problems are Ubiquitous

Shape Matching

General Strategy
Write the functional form of the matching problem and then use an appropriate optimization engine to find a solution.
Matching Problems are Ubiquitous

General Strategy
Write the functional form of the matching problem and then use an appropriate optimization engine to find a solution.

Use past knowledge to make future instances easier . . . ?
<table>
<thead>
<tr>
<th>Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Motivation</strong></td>
</tr>
<tr>
<td><strong>Problem Setup</strong></td>
</tr>
<tr>
<td>Graph Matching and QAPs</td>
</tr>
<tr>
<td><strong>Why learn QAPs?</strong></td>
</tr>
<tr>
<td><strong>Algebraic Structure of $S_n$ and Harmonic Analysis</strong></td>
</tr>
<tr>
<td><strong>Learning in Fourier Space</strong></td>
</tr>
<tr>
<td><strong>Evaluations</strong></td>
</tr>
</tbody>
</table>
The solution of the matching problem is a permutation matrix $y$ such that $yGy^\top = G'$.
$G = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$

$G' = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$

Solution of matching problem is a permutation matrix $y$ such that $yGy^\top = G'$. 

$\sigma := (51342)$
Solution of matching problem is a permutation matrix \( y \)

\[
y = \begin{pmatrix}
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0
\end{pmatrix}
\]

such that \( yGy^\top = G' \)
Quadratic Assignment Problem (QAP)

\[ y^* = \arg \max_y \sum_{ii'} c_{ii'} y_{ii'} + \sum_{ii'jj'} d_{ii'jj'} y_{ii'} y_{jj'} \]
Quadratic Assignment Problem (QAP)

\[ y^* = \arg \max_y \sum_{ii'} c_{ii'} y_{ii'} + \sum_{ii'jj'} d_{ii'jj'} y_{ii'} y_{jj'} \]

Computationally expensive: \( n \geq 40 \) infeasible in general.
Supervised Learning

Given

Training data: \(((x_1, y_1), \ldots, (x_m, y_m))\)

\[ f^\omega(x_i) \approx y_i \]

\((x_1, y_1) : f^\omega(x_1) \approx y_1 \)

\((x_2, y_2) : f^\omega(x_2) \approx y_2 \)

\((x_3, y_3) : f^\omega(x_3) \approx y_3 \)

and so on .......
Learning for QAPs?

Given

Training data: \(((x_1, \sigma_1), ..., (x_m, \sigma_m))\)

\[
\begin{align*}
\arg \max f^\omega (x_i) & \approx \sigma_i \\
(x_1, \sigma_1) &: \arg \max f^\omega (x_1) \approx \sigma_1 \\
(x_2, \sigma_2) &: \arg \max f^\omega (x_2) \approx \sigma_2 \\
(x_3, \sigma_3) &: \arg \max f^\omega (x_3) \approx \sigma_3 \\
\text{and so on ...}
\end{align*}
\]
## Learning for QAPs?

### Given

**Training data** : \(((x_1, \sigma_1), \ldots, (x_m, \sigma_m))\)

\[
\arg \max \; f^\omega (x_i) \approx \sigma_i
\]

\((x_1, \sigma_1) : \arg \max \; f^\omega (x_1) \approx \sigma_1\)

\((x_2, \sigma_2) : \arg \max \; f^\omega (x_2) \approx \sigma_2\)

\((x_3, \sigma_3) : \arg \max \; f^\omega (x_3) \approx \sigma_3\)

and so on .......

\[\ldots \text{and we want to solve} \; \arg \max \; f^\omega (x_i) \text{ cheaply}.\]
Inspired in part by

Caetano et al., PAMI 2009
- Structure learning approach to find most violated constraints using linear assignment.

Xu et al., JMLR 2009
- Use discriminative learning to acquire a domain–specific heuristic for controlling beam–search.

Stobbe et al., AISTATS 2012
- Fourier space sparsity to recover a set function from very few samples.
Harmonic Analysis

Fourier transform of a function $f : \mathbb{R} \mapsto \mathbb{C}$

$$\hat{f}(\lambda) = \sum_{x \in \mathbb{R}} f(x) e^{2\pi i x \lambda} \quad \lambda \in \mathbb{R},$$
Structure of $\sigma \in S_n$

Harmonic Analysis on Symmetric Groups $S_n$

$$\hat{f}(\rho_\lambda) = \sum_{\sigma \in S_n} f(\sigma) \rho_\lambda(\sigma) \quad \rho_\lambda \in \mathcal{R}$$

- $\lambda$ is the integer partition of $n$, $\lambda \vdash n$
- $\rho_\lambda(\sigma)$ is the irreducible representation of $S_n$

$$\rho_\lambda(\sigma) = \begin{pmatrix}
\rho_{1,1} & \cdots & \rho_{1,d_\lambda} \\
\cdots & \cdots & \cdots \\
\rho_{d_\lambda,1} & \cdots & \cdots
\end{pmatrix}$$
Motivation

Learning QAPs

Algebra of $S_n$

Algorithm

Experiments

Properties $S_n$

**Convolution**

$$(f * g)(\sigma) = \sum_{\tau \in S_n} f(\sigma \tau^{-1}) g(\tau) \quad \hat{f} * \hat{g}(\lambda) = \hat{f}(\lambda) \hat{g}(\lambda)$$

**Correlation**

$$(f * g)(\sigma) = \sum_{\tau \in S_n} f(\sigma \tau) g(\tau)^* \quad \hat{f} \star \hat{g}(\lambda) = \hat{f}(\lambda) \hat{g}(\lambda)^\dagger$$

- $S_{n-1}$ is a subgroup of $S_n$
Properties $\mathbb{S}_n$

**Convolution**

$$(f * g)(\sigma) = \sum_{\tau \in \mathbb{S}_n} f(\sigma \tau^{-1}) g(\tau) \quad \hat{f} \ast \hat{g}(\lambda) = \hat{f}(\lambda) \hat{g}(\lambda)$$

**Correlation**

$$(f \star g)(\sigma) = \sum_{\tau \in \mathbb{S}_n} f(\sigma \tau) g(\tau)^* \quad \hat{f} \star \hat{g}(\lambda) = \hat{f}(\lambda) \hat{g}(\lambda)^\dagger$$

- $\mathbb{S}_{n-1}$ is a subgroup of $\mathbb{S}_n$
- The set $\sigma \mathbb{S}_{n-1}$ is called a **left coset** of $\sigma$
- Two left (right) cosets are either disjoint or the same
Cosets provide a partition of $S_n$:
Motivation
Learning QAPs
Algorithm of $S_n$
Experiments

\[ f : S_n \to \mathbb{C} \]

Graph function of $G$

\[ f_A(\sigma) = A_{\sigma(n), \sigma(n-1)} \]

Properties:

- $S_{n-2}$-invariant function on adjacency matrix $A$ (Kondor, 2010)
- \textit{Band-limited} in Fourier domain (Rockmore, 2002)
- Under relabeling, $f_{A\pi} = f_{\bar{A}}^\pi$
Graph Matching Problem

**Standard QAP:**
Given a pair of graphs

\[
\max_{\sigma \in S_n} f(\sigma) = \sum_{i,j=1}^{n} A_{i,j} A'_{\sigma(i),\sigma(j)}
\]

**Graph Correlation:**

\[
f(\sigma) = \frac{1}{(n-2)!} \sum_{\pi \in S_n} f_A(\sigma \pi) f_{A'}(\pi)
\]

\((A, A')\) could be weighted or unweighted adjacency matrices.
Learning Graph Matching

**Given:** A training set of related graph pairs with $D$ encodings of adjacency matrices: $(G_m, G'_m)$, $m = \{1, \cdots, M\}$.

**Goal:** “Learn” parameters $\omega$ such that QAP procedure finds a *good* solution (*quickly*) for the test case (unseen graph pairs).
Learning Graph Matching

**Given:** A training set of related graph pairs with $D$ encodings of adjacency matrices: $(G_m, G'_m), m = \{1, \cdots, M\}.$

**Goal:** “Learn” parameters $\omega$ such that QAP procedure finds a good solution (quickly) for the test case (unseen graph pairs).

- Define parameter vector $\omega \in \mathbb{R}^D$
Learning Graph Matching

**Given:** A training set of related graph pairs with $D$ encodings of adjacency matrices: $(G_m, G'_m)$, $m = \{1, \cdots, M\}$.

**Goal:** “Learn” parameters $\omega$ such that QAP procedure finds a *good* solution (*quickly*) for the test case (unseen graph pairs).

- Define parameter vector $\omega \in \mathbb{R}^D$

**QAP Objective for Learning:**

$$f^\omega(\sigma) = \sum_{d=1}^{D} \omega_d f^d(\sigma)$$

where $f^d(\sigma) = \frac{1}{(n-2)!} \sum_{\pi \in S_n} f^d_A(\sigma \pi) f^d_{A'}(\pi) = \sum_{i,j} A^d_{ij} A'^d_{\sigma(i)\sigma(j)}$
Learning Correct bounds on Coset Tree
Learning Correct bounds on Coset Tree
Learning Correct bounds on Coset Tree
Learning Correct bounds on Coset Tree
# Fourier Domain QAP Solver

## Fast Fourier Transform

\[
\hat{f}_\omega(\lambda) = \sum_{i=1}^{n} \frac{d_\lambda}{nd_\mu} \rho_\lambda([[i, n]]) \bigoplus_{\mu \in \lambda \downarrow n-1} \hat{f}_i(\mu)
\]

## Fourier Space Bounds [Kondor et.al.]

\[
B_{n \rightarrow i} = \sum_{\mu \vdash n-1} \|\hat{f}_i(\mu)\|_*
\]
Risk Minimization

**Loss Function**

\[
\sum_{k=1}^{n} \sum_{i \in \text{children}((n-k+1)^*)} \left[ \| \hat{f}_{i^*}^\omega (\mu) \|_* - \| \hat{f}_{i_{n-k}^*}^\omega (\mu) \|_* + 1 \right]^+
\]

- \(i_{n-k}^*\) is the correct node at level \(n - k\) in coset tree.
Risk Minimization

Jensen’s Inequality

**For parameterization:** \( \hat{f}_i^\omega(\mu) = \sum_{d=1}^{D} \omega_d \hat{f}_i^d(\mu) \)

\[
\| \hat{f}_i^\omega(\mu) \|_* = \| \sum_{d=1}^{D} \omega_d \hat{f}_i^d(\mu) \|_* \leq \sum_{d=1}^{D} \omega_d \| \hat{f}_i^d(\mu) \|_*
\]
Jensen’s Inequality

For parameterization: \( \hat{f}^\omega_i(\mu) = \sum_{d=1}^{D} \omega_d \hat{f}^d_i(\mu) \)

\[
\|\hat{f}^\omega_i(\mu)\|_* = \| \sum_{d=1}^{D} \omega_d \hat{f}^d_i(\mu) \|_* \leq \sum_{d=1}^{D} \omega_d \|\hat{f}^d_i(\mu)\|_*
\]

Fourier space Stochastic Gradient Descent Solver

Each update takes the form

\[
\omega_d \leftarrow \omega_d - \eta \left\{ \|\hat{f}^d_i(\mu)\|_* - \|\hat{f}^d_{i^*_{n-k}}(\mu)\|_* + \frac{\nu}{M\mathcal{O}(n^2)} \omega_d \right\}
\]
Risk Minimization

**Jensen’s Inequality**

*For parameterization:* \( \hat{f}^\omega_i(\mu) = \sum_{d=1}^{D} \omega_d \hat{f}_d^i(\mu) \)

\[
\| \hat{f}^\omega_i(\mu) \|_* = \| \sum_{d=1}^{D} \omega_d \hat{f}_d^i(\mu) \|_* \leq \sum_{d=1}^{D} \omega_d \| \hat{f}_d^i(\mu) \|_*
\]

**Fourier space Stochastic Gradient Descent Solver**

Each update takes the form

\[
\omega_d \leftarrow \omega_d - \eta \left\{ \| \hat{f}_d^i(\mu) \|_* - \| \hat{f}_d^{i*}(\mu) \|_* + \frac{\nu}{M\hat{O}(n^2)} \omega_d \right\}
\]

**Convergence:** emulate proof for \( D \)-dimensional *Perceptron*. 
### Experimental Results

**Setup**

- **Edge:** Delaunay triangulation on interest points
- **Distance:** Euclidean distance between interest points
- **Shape Context (60 in all):** Similarities based on local shape-based appearance of interest points
Experimental Results

Setup

- **Edge:** Delaunay triangulation on interest points
- **Distance:** Euclidean distance between interest points
- **Shape Context (60 in all):** Similarities based on local shape-based appearance of interest points

Task

- Learn $\omega$ using training instances
- Solve the learnt problem “cheaply” (e.g., greedy or linear assignment)
- Evaluate compromise on accuracy?
- Evaluate improvements in running time?
Experimental Results: CMU House

Figure: (Green) the ground truth and (red) the learnt correspondences.
Experimental Results: CMU Hotel

**Figure:** (Green) the ground truth and (red) the learnt correspondences.
Experimental Results: Silhouette

Figure: (Green) the ground truth and (red) the learnt correspondences.
Accuracy vs. Offset: CMU House

**Figure:** Our method compared with no-learn baseline. (Red) learning and (blue) no-learning.
Accuracy vs. Offset: CMU Hotel

**Figure:** Our method compared with no-learn baseline. (Red) learning and (blue) no-learning.
Accuracy vs. Offset: Silhouette

**Figure:** Our method compared with no-learn baseline. (Red) learning and (blue) no-learning.
Incorporating domain knowledge help solving hard problems.

Harmonic analysis provide nice structure for matching problems.

Other parameterization schemes might provide further insights.

Please come to the poster session. Poster 15 in Informatics Forum.
Thank You!