An Out-of-Sample Extension for Spectral Clustering based on Weighted Kernel PCA

Carlos Alzate    Johan A. K. Suykens

ESAT-SCD-SISTA  
Katholieke Universiteit Leuven  
Leuven, Belgium

September 25, 2006

First Research Contact Day of the Computational Intelligence and Learning (CIL) doctoral school
Outline

1. Spectral Clustering
   - Cut Criteria

2. Weighted Kernel PCA
   - LS-SVM Approach to Kernel PCA
   - Introducing Weights
   - Relation with Spectral Clustering
   - Out-of-Sample Extension

3. Empirical Results
   - Toy Example
   - Iris Dataset
   - Golub Microarray

4. Conclusions

5. Future Work
Motivation

- It is not clear what we are optimizing when doing spectral clustering.
- Due to the lack of a clear optimization problem, the parameters selection is not straightforward.
- Clustering of new points should rely on approximation techniques.
Spectral Clustering

Class of clustering algorithms that use the eigenvectors of an affinity matrix derived from the data.

The data are represented as an undirected graph.

The objective is to minimize the cost of cutting the graph into two disjoint sets $A, B$.

The Cut

$$\text{cut}(A, B) = \sum_{a \in A, b \in B} w(a, b)$$

where $w(a, b)$ is the weight between node $a$ and $b$. 
Spectral Clustering

The diagram illustrates a weighted graph with nodes and edges labeled with weights. The weights shown are $W_{12}$, $W_{14}$, and $W_{23}$. The graph is partitioned, possibly indicating a clustering process.
Spectral Clustering
The Mincut

\[ \min_q J_{\text{mincut}} = q^T (D - W) q \]

such that \( q \in \{-1, 1\}^N \)

\( D \): degree matrix, \( W \): affinity matrix, \( q \): cluster membership indicator.

- NP-hard!
- Efficient solution by relaxing
  \[ q \rightarrow \tilde{q}^T \tilde{q} = 1 \]
- Bias for small sets.

The Mincut Relaxation

\[ L\tilde{q} = \lambda \tilde{q} \]

\( L \): graph Laplacian.
Solution: Fiedler vector.
Normalized Cut

\[ \min_q J_{ncut} = \frac{q^T L q}{q^T D q} \]

such that

\[ \begin{cases} 
q \in \{-b, 1\}^N \\
q^T D 1_N = 0 
\end{cases} \]

- NP-complete!
- Efficient solution by relaxing
  \[ q \rightarrow \tilde{q}^T \tilde{q} = 1 \]
- Size of the clusters is taken into account.

Normalized Cut Relaxation

- Generalized eigenvalue problem:
  \[ L\tilde{q} = \lambda D\tilde{q} \]
Markov Random Walks

- Probabilistic interpretation.
- $P = D^{-1}W$.
- $ij$-th entry of $P \rightarrow$ probability of moving from node $i$ to node $j$.

Solution

$Pr = \xi r$.

Solution is the eigenvector corresponding to the second largest eigenvalue.

- Equivalent to the normalized cut:

$$r = \tilde{q}, \lambda = 1 - \xi$$
Kernel Alignment

- Measure of similarity between a kernel and an objective function:

\[
\max_q A(K, q) = q^T \Omega q
\]

such that \( q \in \{-1, 1\}^N \)

where \( \Omega \) is the kernel matrix.

Kernel Alignment Relaxation

After relaxing \( q \) the dual solution is an eigenvalue problem:

\[
\Omega \tilde{q} = \lambda \tilde{q}
\]

which corresponds to kernel PCA!
LS-SVM Approach to Kernel PCA

Clear primal optimization problem to which kernel PCA is the dual. Underlying loss function is explicit $\rightarrow L_2$.

$$\max_{w,e} J_p(w, e) = \gamma \frac{1}{2} e^T e - \frac{1}{2} w^T w$$

such that

$$e = \Phi_c w$$

$\Phi_c$ is the $N \times n_h$ feature matrix:

$$\Phi_c = \begin{bmatrix} \varphi(x_1)^T - \hat{\mu}_{\varphi}^T \\ \varphi(x_2)^T - \hat{\mu}_{\varphi}^T \\ \vdots \\ \varphi(x_N)^T - \hat{\mu}_{\varphi}^T \end{bmatrix}$$

Dual

Eigendecomposition of the centered kernel matrix $\Omega_c$:

$$\Omega_c \alpha = \lambda \alpha$$
Introducing Weights

**Weighted Kernel PCA**

Introducing a weighting matrix $V$ into the formulation:

$$\max_{w,e} J_p(w, e) = \frac{1}{2} e^T V e - \frac{1}{2} w^T w$$

such that $e = \Phi w$

$$\Phi = [\varphi(x_1)^T; \varphi(x_2)^T; \ldots; \varphi(x_N)^T], \quad V = V^T > 0.$$  

**Dual**

Non-symmetric eigenvalue problem:

$$V \Omega \alpha = \lambda \alpha$$

**Equivalence**

If $V = D^{-1}$ then weighted kernel PCA is equivalent to the random walks algorithm.
Changing the weighting matrix $V$ leads to different spectral clustering algorithms:

<table>
<thead>
<tr>
<th>Method</th>
<th>Original Problem</th>
<th>$V$</th>
<th>Relaxed Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alignment</td>
<td>$\Omega q = \lambda q$</td>
<td>$I_N$</td>
<td>$\alpha^{(1)}$</td>
</tr>
<tr>
<td>NCut</td>
<td>$Lq = \lambda Dq$</td>
<td>$D^{-1}$</td>
<td>$\alpha^{(2)}$</td>
</tr>
<tr>
<td>Random walks</td>
<td>$D^{-1}Wq = \lambda q$</td>
<td>$D^{-1}$</td>
<td>$\alpha^{(2)}$</td>
</tr>
<tr>
<td>NJW</td>
<td>$D^{-\frac{1}{2}}WD^{-\frac{1}{2}}q = \lambda q$</td>
<td>$D^{-1}$</td>
<td>$D^{\frac{1}{2}}\alpha^{(2)}$</td>
</tr>
</tbody>
</table>
Clustering New Points

- No straightforward extensions for out-of-sample data points in the spectral clustering framework.
- Extensions can be done via approximation techniques such as Nyström [Bengio et al., 2003].

Score Variables

- No approximation needed! New points can be clustered using the projection onto the eigenvector solution:

\[
\begin{align*}
z(x_{\text{new}}) &= w^T \varphi(x_{\text{new}}) = \sum_{l=1}^{N} \alpha_l K(x_l, x_{\text{new}}). \\
q_{x_{\text{new}}} &= \text{sign}(z(x_{\text{new}}) - \theta)
\end{align*}
\]
### Empirical Results

#### Golub Microarray

<table>
<thead>
<tr>
<th>Method</th>
<th>Nyström RBF k.</th>
<th>Weighted KPCA RBF k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alignment</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>NCut</td>
<td>0.63</td>
<td>0.72</td>
</tr>
<tr>
<td>NJW</td>
<td>0.72</td>
<td>0.86</td>
</tr>
</tbody>
</table>
Conclusions

- Unifying view of spectral clustering based on the weighted kernel PCA formulation.
- Out-of-sample extension based on the primal-dual formulation insights.
- Model selection criterion using the variance of the projections on a validation set.
### Future Work

- Extensions to K-way clustering (more than two clusters).
- Semi-supervised clustering (when some training points have labels).