A Dependence Maximization View of Clustering

Le Song LESONG@IT.USYD.EDU.AU
NICTA, Statistical Machine Learning Program, Canberra, ACT 0200, Australia; and University of Sydney

Alex Smola ALEX.SMOLA@GMAIL.COM
NICTA, Statistical Machine Learning Program, Canberra, ACT 0200, Australia; and ANU

Arthur Gretton ARTHUR.GRETTON@TUEBINGEN.MPG.DE
MPI for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany

Karsten M. Borgwardt BORGWARDT@DBS.IFI.LMU.DE
LMU, Department "Institute for Computer Science", Oettingenstr. 67, 80538 München, Germany

Abstract

We propose a family of clustering algorithms based on the maximization of dependence between the input variables and their cluster labels, as expressed by the Hilbert-Schmidt Independence Criterion (HSIC). Under this framework, we unify the geometric, spectral, and statistical dependence views of clustering, and subsume many existing algorithms as special cases (e.g. k-means and spectral clustering). Distinctive to our framework is that kernels can also be applied on the labels, which can endow them with particular structures. We also obtain a perturbation bound on the change in k-means clustering.

1 Introduction

Given a set of observations $X = \{x_1, \ldots, x_m\} \in \mathcal{X}$, the goal of clustering is to associate with each observation a label chosen from a smaller set $Y = \{y_1, \ldots, y_n\} \in \mathcal{Y}$, such that the x_i grouped together share some underlying property. This is useful both in terms of gaining a better understanding of the data, and in obtaining a succinct representation.

A popular clustering algorithm is k-means (MacQueen, 1967), which can be kernelized straightforward (Schölkopf et al., 1998; Girolami, 2001). Points are grouped in k clusters so as to minimize intra-cluster variance, and the resulting k cluster centroids are used to represent their respective clusters. This problem is NP-hard; however a simple heuristic often produces acceptable results: random initialization followed by iterative update of cluster centroids and the data partition (MacQueen, 1967). Recently, (Zha et al., 2001; Ding & He, 2004) proposed to use the principal components of the data as a more efficient initialization for k-means.

Besides the geometric view of clustering, spectral methods have gained considerable popularity over the past decade, e.g. Shi & Malik (1997); Ng et al. (2002). These methods treat each datum as a node in a nearest neighbor graph and exploit the piecewise constant eigenvectors of a graph Laplacian for the initial partition. Usually, thresholding or a second k-means is performed to obtain a final cluster assignment.

A third view of clustering arises from statistical considerations: given X, we want to find Y such that the statistical dependence between X and Y is maximized. In the case of "hard" clustering, where each observation is assigned to exactly one cluster, mutual information $I(X, Y)$ has been used as the objective (Slonim, 2002, Section 4.2). It is difficult to compute mutual information in high dimensions, however, since sophisticated bias correction methods must be employed (Nemenman et al., 2002).

A natural question is how these different approaches are related. Early work by Zha et al. (2001); Ding & He (2004) show the connection between the geometric view and spectral view of k-means. In this paper, we will start from a statistical dependence view of clustering, and lead naturally to the unification of the three views at a statistical inference level.

Instead of $I(X, Y)$, however, we use the Hilbert Schmidt Independence Criterion (HSIC) (Gretton

Appearing in *Proceedings of the 24^{th} International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

et al., 2005) as our measure of statistical dependence. HSIC is the Hilbert-Schmidt norm of the cross covariance operator between reproducing kernel Hilbert spaces. It has several advantages: first, it does not require density estimation, and has good uniform convergence guarantees; second, it has very little bias, even in high dimensions; and third, a number of algorithms can be viewed as maximizing HSIC subject to constraints on Y, for particular Hilbert spaces on the inputs and labels. Thus, we propose a family of feature extraction and clustering methods, of which k-means and spectral clustering are special cases. Another distinctive feature of our framework is that rich choices of kernels are also applicable to the label spaces. This provides us with new clustering algorithms that generate cluster labels with structures.

Given the many possible clustering algorithms that arise from particular kernel choices, we face the difficulty of choosing which kernel gives the most useful partition of our data. We find the kernels that provide the most reasonable clustering in our examples are also the most stable under perturbation. We measure the stability using a novel bound on the change in clustering performance under perturbation. Unlike previous such bounds by Ng et al. (2002), Kannan et al. (2004), and Meilă (2006), ours holds even under large perturbations, and is distribution independent. We also use this bound to determine the accuracy required of low rank approximations to the input Gram matrix, while still maintaining good clustering performance.

2 Measure of Dependence

Let sets of observations X and Y be drawn jointly from some probability distribution $\Pr_{xy}$. The Hilbert Schmidt Independence Criterion (HSIC) (Gretton et al., 2005) measures the dependence between x and y by computing the norm of the cross-covariance operator over the domain $\mathcal{X} \times \mathcal{Y}$ in Hilbert Space. It can be shown, provided the Hilbert Space is universal, that this norm vanishes if and only if x and y are independent. A large value suggests strong dependence with respect to the choice of kernels.

Formally, let $\mathcal{F}$ be the reproducing kernel Hilbert Space (RKHS) on $\mathcal{X}$ with associated kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and feature map $\phi : \mathcal{X} \to \mathcal{F}$. Let $\mathcal{G}$ be the RKHS on $\mathcal{Y}$ with kernel l and feature map ψ. The cross-covariance operator $\mathcal{C}_{xy} : \mathcal{G} \mapsto \mathcal{F}$ is defined by Fukumizu et al. (2004) as

$$\mathcal{C}_{xy} = \mathbb{E}_{xy}[(\phi(x) - \mu_x) \otimes (\psi(y) - \mu_y)], \tag{1}$$

where $\mu_x = \mathbb{E}[\phi(x)]$, $\mu_y = \mathbb{E}[\psi(y)]$, and $\otimes$ is the tensor product. HSIC, denoted as $\mathcal{D}$, is then defined as the square fo the Hilbert-Schmidt norm of $\mathcal{C}_{xy}$ (Gretton et al., 2005), $\mathcal{D}(\mathcal{F}, \mathcal{G}, \Pr_{xy}) := \|\mathcal{C}_{xy}\|^2_{\text{HS}}$ .In term of kernels HSIC can be expressed as

$$\begin{aligned}\mathbb{E}_{xx'yy'}[k(x,x')l(y,y')] + \mathbb{E}_{xx'}[k(x,x')]\,\mathbb{E}_{yy'}[l(y,y')]\\ -2\,\mathbb{E}_{xy}[\mathbb{E}_{x'}[k(x,x')]\,\mathbb{E}_{y'}[l(y,y')]],\end{aligned}$$

where $\mathsf{E}_{xx'yy'}$ is the expectation over both $(x, y) \sim \Pr_{xy}$ and an additional pair of variables $(x', y') \sim \Pr_{xy}$ drawn *independently* according to the same law. Given a sample $Z = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ of size m drawn from $\Pr_{xy}$ an empirical estimate of HSIC is

$$\mathcal{D}(\mathcal{F}, \mathcal{G}, Z) = (m-1)^{-2} \operatorname{tr}(\mathbf{HKHL}), \tag{2}$$

where $\mathbf{K}, \mathbf{L} \in \mathbb{R}^{m \times m}$ are the kernel matrices for the data and the labels respectively, and $\mathbf{H}_{ij} = \delta_{ij} - m^{-1}$ centers the data and the labels in the feature space. For notational convenience, we always assume that $\mathbf{K}$ is centered, i.e. $\mathbf{K} = \mathbf{HKH}$ and use $\operatorname{tr}(\mathbf{K}\,\mathbf{L})$.

Previous work used HSIC to *measure* independence between given random variables (Gretton et al., 2005). Here we use it to *construct* a compressed representation, Y (the labels), of the data, X, such that their dependence is maximized.

There are several advantages to use HSIC as a dependence criterion. First, HSIC satisfies concentration of measure conditions (Gretton et al., 2005). That is, for random draws of observation from $\Pr_{xy}$, HSIC provides values which are very similar. This is desirable, as we want our clustering to be robust to small changes. Second, HSIC is easy to compute, since only the kernel matrices are required and no density estimation is needed. The freedom of choosing a kernel allows us to incorporate prior knowledge into the dependence estimation process. The consequence is that we are able to generate a family of methods by simply choosing appropriate kernels for X and Y.

3 Clustering via HSIC

Having defined our dependence measure, we now devise an optimization problem for feature extraction via

$$\begin{aligned} Y^* = \operatorname*{argmax}_{Y \in \mathcal{Y}} \; \operatorname{tr}(\mathbf{K}\,\mathbf{L}(Y)) \\ \text{subject to constraints on } Y, \end{aligned} \tag{3}$$

where $\mathbf{L}(Y)$ represents the kernel matrix of the generated labels. The constraints on Y serve two purposes: to ensure dependence is comparable as Y changes; and to endow Y with particular structures. For example, we can use a vector of real numbers, $\mathbf{y}_i$, as the label (from which we form the label feature matrix

$\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_m)^\top$). If we further require that the columns of $\mathbf{Y}$ be orthogonal with unit norm, we can recover kernel PCA by simply applying a linear kernel on $\mathbf{Y}$ (Schölkopf et al., 1998). However, the focus of this paper is to investigate the case of clustering, where the label space $\mathcal{Y} = \{y_1, \ldots, y_c\}$ contains only $c \ll m$ distinctive elements.

In this case, we effectively have the liberty of choosing a positive semidefinite matrix $\mathbf{A}$ of size $c \times c$ defining the similarity between elements in $\mathcal{Y}$. Then the kernel matrix of the labels can be parameterized using a partition matrix, Π, of size $m \times c$,

$$\mathbf{L} = \Pi\,\mathbf{A}\,\Pi^\top,\ (\Pi\,\mathbf{1} = \mathbf{1},\ \Pi_{ij} \in \{0,1\}) \tag{4}$$

Each row of the partition matrix, Π, contains all zeros but a single entry of 1, and $\mathbf{1}$ denotes a vector of all ones. Π effectively constrains us to assign each data point to a particular label by putting 1 in an appropriate column. In this case, (3) becomes

$$\Pi^* = \underset{\Pi}{\mathrm{argmax}}\ \ \mathrm{tr}(\mathbf{K}\,\Pi\,\mathbf{A}\,\Pi^\top) \tag{5}$$
$$\text{subject to } \Pi\,\mathbf{1} = \mathbf{1},\ \Pi_{ij} \in \{0,1\}$$

Using properties of the trace,

$$\mathrm{tr}((\Pi^\top\,\mathbf{K}\,\Pi)\,\mathbf{A}) = \mathbf{1}^\top((\Pi^\top\,\mathbf{K}\,\Pi) \circ \mathbf{A})\,\mathbf{1}, \tag{6}$$

where $\circ$ denotes elementwise matrix multiplication. This means that in (5) we try to partition the kernel matrix such that the sum of kernel entries in the corresponding blocks best align with the given structure in $\mathbf{A}$ (this is explained further in section 4).

Very often we will use the normalized partition matrix $\mathbf{P}$ instead of Π. These are related by $\mathbf{P} = \Pi\,\mathbf{D}$, where $\mathbf{D}$ is a diagonal matrix with entries $\mathbf{D}_{ii} = (\mathbf{1}^\top\,\Pi_{\cdot i})^{-1/2}$ (We use subscript $\cdot i$ to represent a column; similarly, $i\cdot$ represents a row). In this case, the sum of kernel entries in each block is normalized before the alignment.

The optimization in (5) usually involves mixed integer nonlinear programming. Polynomial time exact solutions exist only in special cases, e.g. when the data are partitioned evenly into c clusters (Loera et al., 2006). Therefore, either heuristics or relaxations are employed to solve the problem approximately. We designed an iterative algorithm for this problem by greedily maximizing the dependence in each step.

Clustering using HSIC (CLUHSIC) is presented in Algorithm 1. It first randomly initializes the partition matrix Π (step 1). Then it iterates through each data point, finding the cluster assignment of each point that maximizes the dependence (step 5). The algorithm terminates when changes of assignment of any single data point can no longer increase the dependence (step 8). We denote $\Pi^j_{i\cdot}$ as the partition matrix derived from Π by reassigning data point i to cluster j.

Algorithm 1 CLUHSIC

Input: The kernel matrices, $\mathbf{K}$ and $\mathbf{A}$, the number of data, m, and the number of cluster, c.
Output: The cluster assignment, Π^*.

```
1: $\Pi \leftarrow \Pi_0$
2: repeat
3:   $\Pi_0 \leftarrow \Pi$
4:   for all $i \in \{1 \ldots m\}$ do
5:     $j_0 \leftarrow \mathrm{argmax}_j\ \mathrm{tr}(\mathbf{K}\,\Pi^j_{i\cdot}\,\mathbf{A}\,\Pi^{j\top}_{i\cdot}),\ j \in \{1 \ldots c\}$
           subject to $\Pi\,\mathbf{1} = \mathbf{1},\ \Pi_{ij} \in \{0,1\}$
6:     $\Pi \leftarrow \Pi^{j_0}_{i\cdot}$
7:   end for
8: until $\Pi_0 = \Pi$
```

This greedy procedure is guaranteed to terminate, since in each step we increase the objective and there is only finite number of different cluster assignments. We observe experimentally that the outer loop repeats less than 20 times. Further speedup is also possible by taking into the fact that adjacent computations of $\mathrm{tr}(\mathbf{K}\,\Pi^j_{i\cdot}\,\mathbf{A}\,\Pi^{j\top}_{i\cdot})$ share many common structures.

4 CLUHSIC Family

Our formulation in (5) is very general: we can obtain a family of clustering algorithms by combining a kernel for the input space and one for the labels. While different kernels for the input space have been explored extensively (eg. Schölkopf et al., 1998; Girolami, 2001), almost no studies have explicitly investigated kernels on the label space. In this section, after a brief overview of input space kernels, we will focus on the rich choice of kernels for the labels, and the associated clustering algorithms. This way we show that k-means is a special case of the CLUHSIC family.

There exist a great number of kernels on the input spaces (Schölkopf & Smola, 2002), some basic examples being the inner product $\langle x, x' \rangle$ (or linear kernel); the polynomial kernel $(\langle x, x' \rangle + c)^d$ for some $c \geq 0$ and $d \in \mathbb{N}$; and the RBF kernel family (Gauss, Laplace).

Graph kernels are also commonly used in clustering, where the similarity measure is given by the connectivity of vertices. One first builds a nearest neighbor graph $\mathbf{W}$ on the inputs, and then takes either the pseudo-inverse $\mathbf{G}^\dagger$ or the exponentiation $\exp(-s\,\mathbf{G})$ of the graph Laplacian $\mathbf{G}$ as the graph kernel (Smola & Kondor, 2003). In some cases, one can learn a graph kernel matrix directly from the data (Xiao et al., 2006).

While the kernels mentioned above are limited to vectorial data, more sophisticated kernels on structured objects and general sets can also be applied.

We now describe possible RKHS kernels on label spaces. Although these can in principle be very general, we will present certain label spaces that are of particular interest for clustering.

Plain (kernel) k-means We set $\mathbf{A}$ to a diagonal matrix (Zha et al., 2001; Ding & He, 2004; Zass & Shashua, 2005) (3 cluster example)

$$\mathbf{A} = \operatorname{diag}\left(\begin{bmatrix} m_1^{-1} & m_2^{-1} & m_3^{-1} \end{bmatrix}\right), \tag{7}$$

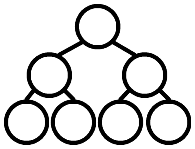
where $\operatorname{diag}(\mathbf{a})$ is a diagonal matrix with non-zero entries $\mathbf{a}$. Alternatively, we can use the normalized partition matrix $\mathbf{P}$ and set $\mathbf{A}$ as the identity matrix. Each cluster is assigned to an arbitrary column in Π. This label kernel requires the labels be mutually orthogonal and assumes no relation between clusters. This can be misleading when structure in the inputs exists.

Weighted k-means Each input x_i is associated with a weight w_i that represents its importance. Again we set $\mathbf{A}$ to a diagonal matrix (3 cluster example)

$$\mathbf{A} = \operatorname{diag}\left(\begin{bmatrix} \frac{1}{\sum_{i\in\Pi_{\cdot 1}} w_i} & \frac{1}{\sum_{i\in\Pi_{\cdot 2}} w_i} & \frac{1}{\sum_{i\in\Pi_{\cdot 3}} w_i} \end{bmatrix}\right), \tag{8}$$

where $i \in \Pi_{\cdot j}$ denotes the indices of the inputs that are in cluster j. In this case, the entries in $\mathbf{K}$ are also weighted accordingly, i.e. $w_i w_l k(x_i, x_l)$ (Dhillon et al., 2004). Weighted k-means also ignores relations between clusters.

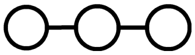
Hierarchical clustering We can easily specify a hierarchy in the cluster labels by making some off-diagonal entries of $\mathbf{A}$ nonzero. For example, we may have a two-level clustering problem: first divide the inputs in 4 clusters, and then further group them into 2 super-clusters (each containing 2 sub-clusters). This requirement can be tackled using



$$\mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \tag{9}$$

Unlike k-means, the assignment of clusters to columns of Π is no longer arbitrary. Cluster membership will be arranged according to the hierarchical structure specified in $\mathbf{A}$. Therefore, once the inputs are clustered, we can recognize the relations between the clusters by observing the labels.

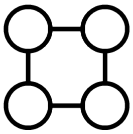
Cluster data in a chain In some cases, the inputs may reside on a manifold, and we want to cluster them by grouping adjacent data points. Obviously, the clusters are related: they from a chain or a grid. The following choice of $\mathbf{A}$ encodes this in the labels



$$\mathbf{A} = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \tag{10}$$

(a chain of 3 clusters). This label kernel enforces that adjacent clusters be similar. Therefore, adjacent columns of Π represent adjacent clusters.

Cluster data in a ring Besides a chain or a grid structure on the clusters, the inputs may reside on a closed manifold. For example, a ring structured cluster relationship can be induced as follows



$$\mathbf{A} = \begin{pmatrix} 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \end{pmatrix} \tag{11}$$

(an example with 4 clusters). The two corner entries in the anti-diagonal of $\mathbf{A}$ close the chain structure of (10) into a ring. Explicitly enforcing a ring structure in the clusters is useful since this may help avoid ambiguity during the clustering (see experiments).Finally, we note that additional structure might be discovered in the data using more complex label kernels than those covered here, for instance deeper tree hierarchies.

5 Relation to Spectral Clustering

In some cases, methods more sophisticated than randomization can be used to initialize CLUHSIC (step 1 in Algorithm 3). Zha et al. (2001) showed that spectral relaxation can be effectively used to initialize k-means clustering. We now briefly review these results from the point of view of dependence maximisation with HSIC. Although the results we present in this section are not new, our framework makes the connection between k-means and its spectral relaxation much simpler conceptually: both are dependence maximization processes, and they differ only in their label spaces.

Let $\mathcal{D}(Y^*) = \operatorname{tr}(\mathbf{K}\,\mathbf{L}(Y^*))$ be the maximal dependence captured by a label space $\mathcal{Y}$. A larger label space (for the same kernel) may capture more information about the data: given two label spaces $\mathcal{Y}_1$ and $\mathcal{Y}_2$, we have

$$\mathcal{Y}_1 \subseteq \mathcal{Y}_2 \Rightarrow \mathcal{D}(Y_1^*) \leq \mathcal{D}(Y_2^*) \tag{12}$$

where $Y_1^* \in \mathcal{Y}_1$ and $Y_2^* \in \mathcal{Y}_2$ are the maximizers of problem (3) for $\mathcal{Y}_1$ and $\mathcal{Y}_2$, respectively. This relation suggests a general strategy for problem (3): apply a cascade of relaxations on the label spaces $\mathcal{Y} \subseteq \mathcal{Y}_1 \subseteq \ldots \subseteq \mathcal{Y}_n$, and use the more relaxed solutions to initialise the less relaxed ones.

This strategy has been exploited in the spectral relaxation of k-means (Zha et al., 2001; Ding & He, 2004): first, we drop all constrains on the normalized partition matrix $\mathbf{P}$ except $\mathbf{P}^\top \mathbf{P} = \mathbf{I}$. We can then efficiently obtain an initialization for k-means via principal component analysis and a pivoted QR decomposition (Zha et al., 2001; Ding & He, 2004). Note that using a graph kernel as $\mathbf{K}$ establishes the link between k-means and spectral clustering (Shi & Malik, 1997; Ng et al., 2002; Meilă, 2003). There is, however, no guarantee that (kernel) k-means or its relaxation will lead to the desired clustering.

6 Stability under Perturbation

Given the wide variety of kernels we have proposed for HSIC-based clustering, we require a means of choosing the kernel that best reveals meaningful data structure. In this section we provide a perturbation bound for the clustering, which may be used in kernel choice: kernels that provide the most reasonable clustering are also the most stable under perturbation.

Let Δ be the perturbation such that the perturbed (centered) kernel matrix $\tilde{\mathbf{K}}$ is related to the original (centered) kernel matrix $\mathbf{K}$ by $\mathbf{K} = \tilde{\mathbf{K}} + \Delta$. Two clusterings obtained respectively before and after the perturbation are denoted $\mathbf{P}$ and $\tilde{\mathbf{P}}$ (both are normalized partition matrices, and can be treated as the labels). Let c be the number of clusters, and $\mathbf{L}$ and $\mathcal{D}$ be the label kernel matrix and the dependence estimated by $\mathbf{P}$, respectively. Denote $\mathcal{D}^\circ$ as the dependence estimated by the first $c-1$ principal eigenvectors $\mathbf{U}$ of $\mathbf{K}$. Similarly, define $\tilde{\mathbf{L}}$, $\tilde{\mathcal{D}}$, $\tilde{\mathcal{D}}^\circ$, and $\tilde{\mathbf{U}}$ for $\tilde{\mathbf{K}}$.

First, we note that the dependence $\operatorname{tr}(\mathbf{H}\,\mathbf{L}\,\mathbf{H}\,\tilde{\mathbf{L}})$ constitutes a measure of similarity between two clusterings. In fact, a variant called χ^2 functional has already been employed for comparing clusterings (Meilă, 2005). A distance, ϵ, between the clusterings can be obtained as

$$\begin{aligned}\epsilon &= \operatorname{tr}(\mathbf{H}\,\mathbf{L}\,\mathbf{H}\,\mathbf{L}) + \operatorname{tr}(\mathbf{H}\,\tilde{\mathbf{L}}\,\mathbf{H}\,\tilde{\mathbf{L}}) - 2\operatorname{tr}(\mathbf{H}\,\mathbf{L}\,\mathbf{H}\,\tilde{\mathbf{L}}) \\ &= \|\,\mathbf{H}\,\mathbf{L}\,\mathbf{H} - \mathbf{H}\,\tilde{\mathbf{L}}\,\mathbf{H}\,\|_F^2 = \|\,\mathbf{L} - \tilde{\mathbf{L}}\|_F^2 \qquad (13)\end{aligned}$$

where $\|\cdot\|_F$ denotes matrix Frobenius norm, and the centering matrix $\mathbf{H}$ does not affect the distance. We will relate ϵ to two factors: (i) the estimated dependence between the inputs and the labels; (ii) the size of the perturbation.

Let λ_{c-1} and λ_c be the $c-1$ and c-th principal eigenvalues of $\mathbf{K}$ (and define $\tilde{\lambda}_{c-1}$ and $\tilde{\lambda}_c$ accordingly for $\tilde{\mathbf{K}}$). Further define $\eta = \operatorname{tr}(\Delta\tilde{\mathbf{U}}\tilde{\mathbf{U}}^\top)$, which measures the size of the perturbation. The following theorem quantifies the relationship between ϵ and its determining factors. See (Song et al., 2007) for the proof.

Theorem 1 *The distance, ϵ, between the two clusterings, $\mathbf{P}$ and $\tilde{\mathbf{P}}$, is bounded by:*

$$\epsilon \leq 2\left(\sqrt{\delta} + \sqrt{\tilde{\delta}} + \sqrt{\gamma}\right)^2 \qquad (14)$$

where δ, $\tilde{\delta}$ and γ are defined respectively as

$$\frac{\mathcal{D}^\circ - \mathcal{D}}{\lambda_{c-1} - \lambda_c},\ \frac{\tilde{\mathcal{D}}^\circ - \tilde{\mathcal{D}}}{\tilde{\lambda}_{c-1} - \tilde{\lambda}_c},\ \frac{\mathcal{D}^\circ - \tilde{\mathcal{D}}^\circ - \eta}{\lambda_{c-1} - \lambda_c} \qquad (15)$$

We see that ϵ is mainly influenced by the ratio of the dependence gap (e.g. $\mathcal{D}^\circ - \mathcal{D}$) to the eigenvalue gap (e.g. $\lambda_{c-1} - \lambda_c$). If changes in dependence due to perturbation and discretization are small compared to eigenvalue gaps, it is likely that the clustering will remain unchanged. Our error bound is distribution free, and does not require the perturbation to be small. This is in contrast to the results by (Ng et al., 2002), Kannan et al. (2004), and Meilă (2006). Hence, our result may have wider applicability.

One case of particular interest is where the perturbation is caused by our performing a low rank approximation of the kernel matrix, $\mathbf{K} \approx \mathbf{B}\,\mathbf{B}^\top = \tilde{\mathbf{K}}$ (e.g. incomplete Cholesky factorization (Fine & Scheinberg, 2001)). In this case, the eigenvalues of $\mathbf{K}$ always dominate the corresponding ones in $\tilde{\mathbf{K}}$. Denoting as $\xi = \operatorname{tr}(\Delta)$ the approximation error, we see that $\xi \geq \mathcal{D}^\circ - \tilde{\mathcal{D}}^\circ - \eta$. Instead of using a fixed error tolerance for the decomposition, this inequality suggests a data-dependent way of setting it: with a tolerance of $\lambda_{c-1} - \lambda_c$, almost no clustering error will be incurred due to the low rank approximation. We will also demonstrate this in our experiments.

7 Experiments

Our experiments address three main questions: (i) how to choose a kernel for a particular clustering problem; (ii) how to use the perturbation bound for kernel matrix approximation; and (iii) how to cluster inputs with rich structured labels.

7.1 Kernel Choice and Stability

In our first series of experiments, we evaluate kernels with respect to their ability to solve certain clustering problems, and their stability after data perturbation.

Choice of Kernels We investigated three artificial datasets (Collinear, Ring, and XOR; see Figure 1), to show how the choice of kernel affects spectral initialization and the correct clustering.

For the Collinear dataset, both the linear and RBF kernels are capable of capturing information about the spherical shape of the clusters. $\mathbf{K}$ computed with a linear kernel has only one piecewise constant eigenvector due to the collinearity of the cluster centers. By contrast, a Gaussian kernel produces two such informative eigenvectors and allows a spectral initialization of k-means. This is because $\mathbf{K}$ produced by the Gaussian kernel has full rank (Schölkopf & Smola, 2002) and hence avoids the collinear degeneracy.

For the Ring dataset, a Gaussian kernel is not able to produce the correct clustering. This is reflected in its eigenvectors. Although some authors (eg Girolami, 2001), obtain correct clustering using an RBF kernel, this is actually a local minimum. The eigenvectors of a graph kernel provide a very good indication of the clusters, and lead to correct clustering.

For the XOR dataset, although a graph kernel provides piecewise constant eigenvectors, the piecewise structure is not consistent with the cluster structure. Hence the resulting clustering is incorrect. The polynomial kernel of degree 2 gives consistent information for spectral initialization and correct clustering.

These examples provide us with two rules for choosing kernels. First, a kernel should introduce the correct notion of similarity. Second, it should be powerful enough (e.g. a Gaussian kernel vs. a linear one) to avoid the degenerate case for spectral clustering.

Stability of Kernels In this section, we perturb the datasets from the previous section, and plot the scaling of γ computed from different kernels as a function of the amount of perturbation (Figure 2). The scaling gives a very good indication of the fitness of a kernel for a particular dataset. This also provides a method for choosing an appropriate kernel.

For the Collinear dataset, both the Gaussian kernel and the graph kernel provide similar stability. This is consistent with the fact that both can detect spherical clusters. For the Ring dataset, however, the graph kernel is clearly better than the other two. This is because the notion of similarity in this dataset is well reflected by graph connectivity. For the XOR dataset, the polynomial kernel becomes the most stable one, since it is the only one that defines a correct similarity measure on the data.

7.2 Kernel Matrix Approximation

In our second set of experiments, we focus on speeding up clustering by kernel matrix approximation. Incomplete Cholesky decomposition is used to approximate the kernel matrix, i.e. $\mathbf{K} \approx \mathbf{B}\mathbf{B}^\top$ ($\mathbf{B} \in \mathbb{R}^{m\times d}$ where $d \ll m$). Here we use the perturbation bound to set the approximation tolerance of the incomplete Cholesky decomposition. More precisely, we further decompose the kernel matrix if the approximation error ξ is larger than the eigengap $\lambda_{c-1} - \lambda_c$.

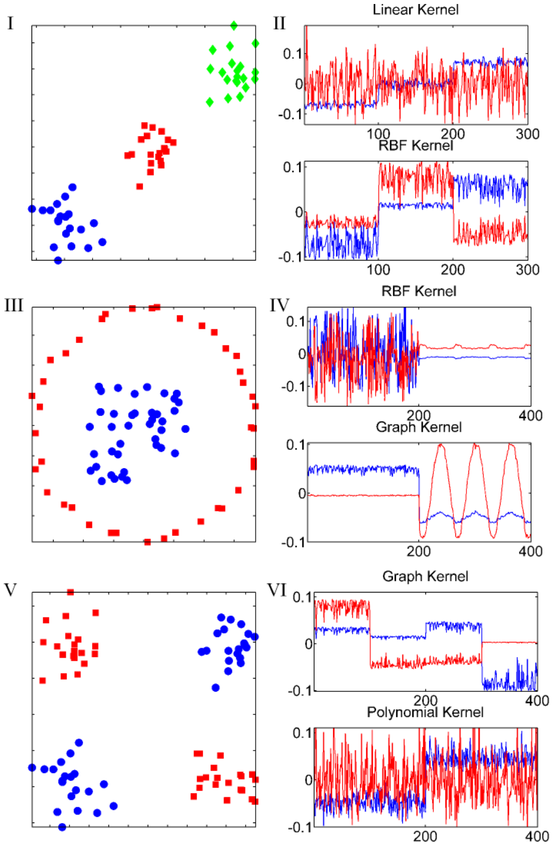


Figure 1: Three artificial datasets and the first 2 principal eigenvectors computed from various kernels matrices. **Left column, top to bottom:** I) Collinear dataset, III) Ring dataset, and V) XOR dataset (data from the same cluster scatter diagonally around the origin). Data points with identical colors and shapes belong to the same class. **Right column:** eigenvectors computed for the corresponding datasets on the left. The first principal eigenvector is colored blue and the second in red. For Colinear, results are for linear and RBF kernels (II). For Ring, results are for RBF and graph kernels (IV). For XOR, results are for graph and polynomial kernels (VI).

We carried out experiments in 9 datasets taken from the UCI and Statlib repositories, and report results in Table 1. The number of classes in the datasets varies from 2 to 11. We are interested in the number of data points clustered differently from the true labels, err1 and err2, computed respectively before and after the incomplete Cholesky decomposition. We also report the time (in seconds), t1, taken to compute the eigenvectors of the full kernel matrix, and the time, t2, to compute the singular vectors of the incomplete Cholesky approximation. The time is recorded for matlab using the routines eig and svd, respectively.

We find that by guiding the incomplete Cholesky de-

Table 1: Clustering error and speed before and after performing incomplete Cholesky decomposition. err1, t1: clustering error and time using the full kernel matrix; err2, t2: clustering error and time using the incomplete Cholesky factor. col#: number of columns in the incomplete Cholesky factor. (m, d): sample size and dimension. c: number of clusters.

	Breastcancer	Iris	Wine	Soybean	Vehicle	Glass	Segment	USPS	Vowel
(m,d)	(669,10)	(150,4)	(178,13)	(47,21)	(846,18)	(214,9)	(2310,18)	(2007,256)	(990,10)
c	2	3	3	4	4	6	7	10	11
err1	3.7	16.0	4.5	0.0	65.4	51.4	36.0	47.0	68.9
err2	3.7	18.0	5.1	0.0	65.4	51.4	36.0	46.9	68.9
t1	18.9	0.2	0.4	0.0	38.0	0.6	647.0	537.9	57.1
t2	0.0	0.0	0.0	0.0	0.3	0.0	11.4	157.3	2.1
col#	35	12	61	32	137	93	228	1518	309

composition using the eigengap and performing singular value decomposition to obtain cluster initialization, almost no *clustering errors* result (or misclassification errors in Meilă, 2006). At the same time, the computation time is greatly reduced for several datasets, up to an order of magnitude.

7.3 Clustering with Rich Label Kernels

In our final set of experiments, we demonstrate clustering with rich label kernels, for both tree and ring structures. The experiments were all conducted on image data, with pixel vectors normalized in each dimension to zero mean and unit variance. We used a Gaussian kernel $\exp(-s\|\mathbf{x}-\mathbf{x}'\|^2)$ with $s = d^{-1}$ (d:dimension of the data) for the kernel matrix $\mathbf{K}$.

Hierarchical Facial Expression Clustering Computer recognition of facial expressions is an important component in automated understanding of human communication (Pantic & Rothkrantz, 2000). We describe a hierarchical clustering of face images that takes into account both the identity of the individual and the emotion being expressed. We collected 185 images (size 217 × 308) of 3 types of facial expressions (NE: neutral, HA: happy, SO: shock) from 3 subjects (CH, LE, AR), in alternating order, with around 20 repetitions each. We registered the images by aligning the eyes, and adjusting the average pixel intensities to be the same. We plotted the data points in Figure 3(a), using the entries in the top 3 eigenvectors of $\mathbf{K}$ as the coordinates. Typical images from each clusters are plotted beside the clusters. Besides clustering the images into individuals and different expressions (9 clusters), we also require that images belong to the same person should also be grouped together (3 groups). This hierarchy is shown in Figure 3(b), as successfully obtained using $\mathbf{P}$ and $\mathbf{A}$ in (9). This can be checked by examining the columns of the resulting partition matrix: CLUHSIC clusters each person into adjacent columns according to $\mathbf{A}$. Although plain k-means is also able to cluster the images correctly into 9 cluster, the hierarchy is lost: k-means place the cluster assignments for each person into arbitrary columns.

Clustering Teapots in a Ring We used the 400 consecutive teapot images (of size 76 × 101 with RGB color) from Weinberger & Saul (2006). The images were taken successively as the teapot was rotated 360°. We plot the data points as dots in Figure 4, using their corresponding entries in the top 3 eigenvectors of $\mathbf{K}$ as the coordinates. To make the clustering problem more difficult, we perturb the images at the points where they are close (the perturbed images have indices in two ranges: from 90 to 105, and from 290 to 305), so as to bring these clusters closer together. We wish to divide the data into 10 clusters, such that images in the same cluster come from similar viewing angles, and images in adjacent clusters come from consecutive angles. We present the different clusters produced by plain k-means in Figure 4, where images in the same cluster are dots of identical shape and color. k-means incorrectly groups images of opposite angle into the same cluster (yellow square dots). CLUHSIC avoids this error by coding a ring structure into $\mathbf{A}$ as in (11). In addition, the labels generated by CLUHSIC preserve the ordering of the images (adjacent clusters represent consecutive angles, and the clusters form a ring), while k-means does not.

8 Conclusion

In this paper, we view clustering as a process of inferring the labels from the data, and employ the Hilbert-Schmidt Independence Criterion (HSIC) as the underlying inference rule. Clustering via HSIC allows us to unify various views of clustering, and to generalise to clustering in hierarchies, chains, rings, and other complex structures. Furthermore, we provide a perturbation bound on the stability of the clustering, which has practical implications for kernel selection. We expect that our framework will provide useful guidance to the practice of clustering.

Acknowledgments NICTA is funded through the Australian Government's *Baking Australia's Ability* initiative, in part through the Australian Research Council. This research was supported by the Pascal Network (IST-2002-506778).

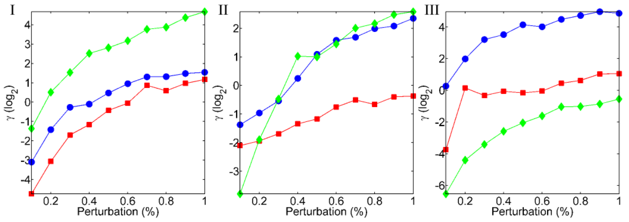
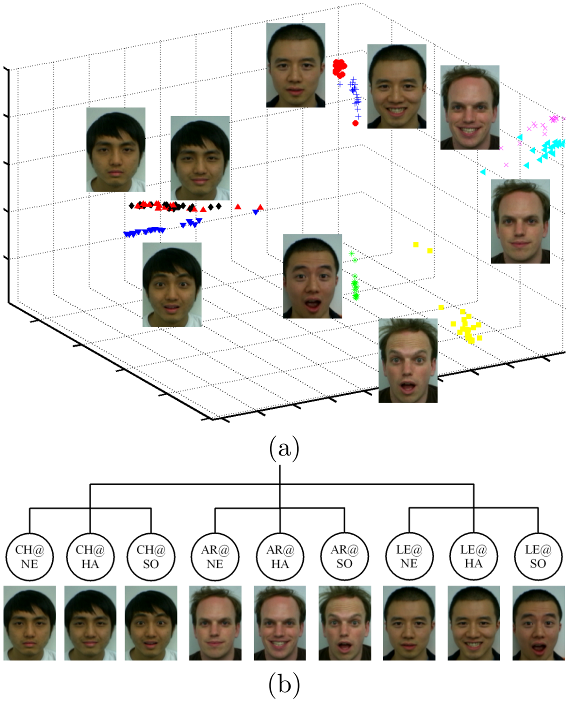


Figure 2: γ as a function of the amount of perturbation applied to the artificial datasets in Figure 1 using three kernels: RBF kernel (blue circle), graph kernel (red square) and polynomial kernel (green diamond). Results are for Colinear (I), Ring (II), and XOR (III) datasets.



(a)

(b)

Figure 3: (a) Facial expression images embedded into 3 dimensional space; different marker shape/colour combinations represent the true identity/expression clusters. (b) The two-level hierarchy recovered from the data.

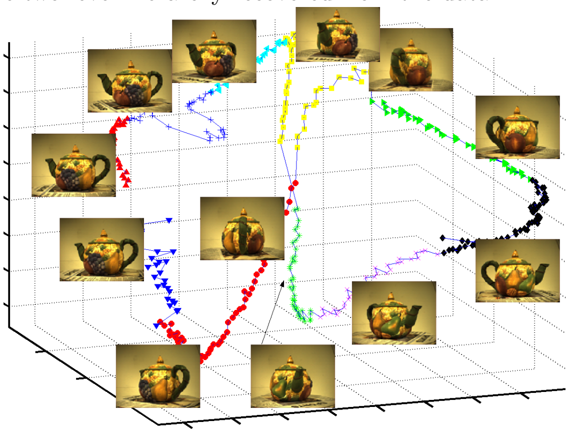


Figure 4: Teapot images, 360° rotation, k-means labeling.

References

Dhillon, I. S., Guan, Y., & Kulis, B. (2004). Kernel kmeans, spectral clustering and normalized cuts. *KDD*.

Ding, C., & He, X. (2004). K-means clustering via principal component analysis. *ICML*.

Fine, S., & Scheinberg, K. (2001). Efficient SVM training using low-rank kernel representations. *JMLR*, *2*, 243–264.

Fukumizu, K., Bach, F. R., & Jordan, M. I. (2004). Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *JMLR*, *5*, 73–99.

Girolami, M. (2001). Mercer kernel based clustering in feature space. *IEEE TNN*, *4*(13), 780–784.

Gretton, A., Bousquet, O., Smola, A., & Schölkopf, B. (2005). Measuring statistical dependence with Hilbert-Schmidt norms. In *ALT*.

Kannan, R., Vempala, S., & Vetta, A. (2004). On clusterings: good, bad and spectral. *J. of ACM*, *51*, 497–515.

Loera, J. D., Hemmecke, R., Onn, S., Rothblum, U., & Weismantel, R. (2006). Integer convex maximization. *arXiv:math.CO/0609019*.

MacQueen, J. (1967). Some methods of classification and analysis of multivariate observations. In L. M. LeCam, & J. Neyman, eds., *Proc. 5th Berkeley Symposium on Math., Stat., and Prob.*, 281. U. California Press.

Meilă, M. (2003). Data centering in feature space. *AISTATS*.

Meilă, M. (2005). The local equivalence of two distances between clustering: the misclassification error metric and the chi-squared distance. *ICML*.

Meilă, M. (2006). The uniqueness of a good optimal for k-means. *ICML*.

Nemenman, I., Shafee, F., & Bialek, W. (2002). Entropy and inference, revisited. In *NIPS*.

Ng, A., Jordan, M., & Weiss, Y. (2002). Spectral clustering: Analysis and an algorithm. In *NIPS*.

Pantic, M., & Rothkrantz, L. (2000). Automatic analysis of facial expressions: The state of the art. *PAMI*, *22*(12), 1424 – 1445.

Schölkopf, B., & Smola, A. (2002). *Learning with Kernels*. Cambridge, MA: MIT Press.

Schölkopf, B., Smola, A. J., & Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, *10*, 1299–1319.

Shi, J., & Malik, J. (1997). Normalized cuts and image segmentation. *CVPR*.

Slonim, N. (2002). *The Information Bottleneck: Theory and Applications*. Ph.D. thesis, The Hebrew University of Jerusalem.

Smola, A. J., & Kondor, I. R. (2003). Kernels and regularization on graphs. In *COLT*.

Song, L., Smola, A., Gretton, A., & Borgwardt, K. (2007). Clustering via dependence maximization. Tech. rep., NICTA, ANU.

Weinberger, K. Q., & Saul, L. K. (2006). An introduction to nonlinear dimensionality reduction by maximum variance unfolding. *AAAI*.

Xiao, L., Sun, J., & Boyd, S. (2006). A duality view of spectral methods for dimensionality reduction. *ICML*.

Zass, R., & Shashua, A. (2005). A unifying approach to hard and probablistic clustering. *ICCV*.

Zha, H., Ding, C., Gu, M., He, X., & Simon, H. (2001). Spectral relaxation for k-means clustering. In *NIPS*.