Spectral Kernel Methods for Clustering

Nello Cristianini
BIOWulf Technologies
nello@support-vector.net

John Shawe-Taylor
Jaz Kandola
Royal Holloway, University of London
{john, jaz}@es.us.es.

Abstract

In this paper we introduce new algorithms for unsupervised learning based on the use of a kernel matrix. All the information required by such algorithms is contained in the eigenvectors of the matrix or of closely related matrices. We use two different but related cost functions, the Alignment and the 'cut cost'. The first one is discussed in a companion paper [3], the second one is based on graph theoretic concepts. Both functions measure the level of clustering of a labeled dataset, or the correlation between data clusters and labels. We state the problem of unsupervised learning as assigning labels so as to optimize these cost functions. We show how the optimal solution can be approximated by slightly relaxing the corresponding optimization problem, and how this corresponds to using eigenvector information. The resulting simple algorithms are tested on real world data with positive results.

1 Introduction

Kernel based learning provides a modular approach to learning system design [2]. A general algorithm can be selected for the appropriate task before being mapped onto a particular application through the choice of a problem specific kernel function.

The kernel based method works by mapping data to a high dimensional feature space implicitly defined by the choice of the kernel function. The kernel function computes the inner product of the images of two inputs in the feature space. From a practitioners viewpoint this function can also be regarded as a similarity measure and hence provides a natural way of incorporating domain knowledge about the problem into the bias of the system.

One important learning problem is that of dividing the data into classes according to a cost function together with their relative positions in the feature space. We can think of this as clustering in the kernel defined feature space, or non-linear clustering in the input space.

In this paper we introduce two novel kernel-based methods for clustering. They both assume that a kernel has been chosen and the kernel matrix constructed. The methods then make use of the matrix's eigenvectors, or of the eigenvectors of the closely related Laplacian matrix, in order to infer a label assignment that approximately optimizes one of two cost functions. See also [4] for use of spectral decompositions of the kernel matrix. The paper includes some analysis of the algorithms together with tests of the methods on real world data with encouraging results.
2 Two partition cost measures

All the information needed to specify a clustering of a set of data is contained in the matrix $M_{ij} = \text{cluster}(x_i) = \text{cluster}(x_j)$, where $(A \equiv B) \in \{-1, +1\}$. After a clustering is specified, one can measure its cost in many ways. We propose here two cost functions that are easy to compute and lead to efficient algorithms.

Learning is possible when some collusion between input distribution and target exists, so that we can predict the target based on the input. Typically one would expect points with similar labels to be clustered and the clusters to be separated.

This can be detected in two ways: either by measuring the amount of label-clustering or by measuring the correlation between such variables. In the first case, we need to measure how points of the same class are close to each other and distant from points of different classes. In the second case, kernels can be regarded as oracles predicting whether two points are in the same class. The ‘true’ oracle is the one that knows the true matrix $M$. A measure of quality can be obtained by measuring the Pearson correlation coefficient between the kernel matrix $K$ and the true $M$.

Both approaches lead to the same quantity, known as the alignment [3].

We will use the following definition of the inner product between matrices

$$\langle K_1, K_2 \rangle_F = \sum_{i,j=1}^m K_1(x_i, x_j) K_2(x_i, x_j).$$

The index $F$ refers to the Frobenius norm that corresponds to this inner product.

**Definition 1 Alignment.** The (empirical) alignment of a kernel $k_1$ with a kernel $k_2$ with respect to the sample $S$ is the quantity

$$\bar{A}(S, k_1, k_2) = \frac{\langle K_1, K_2 \rangle_F}{\sqrt{\langle K_1, K_1 \rangle_F \langle K_2, K_2 \rangle_F}}$$

where $K_i$ is the kernel matrix for the sample $S$ using kernel $k_i$.

This can also be viewed as the cosine of the angle between the $m$-dimensional vectors $K_1$ and $K_2$, representing the Gram matrices. If we consider $k_2 = y y'$, where $y$ is the vector of $\{-1, +1\}$ labels for the sample, then with a slight abuse of notation

$$\bar{A}(S, k, y) = \frac{\langle K, y y' \rangle_F}{\sqrt{\langle K, K \rangle_F \langle y y', y y' \rangle_F}} = \frac{\langle K, y y' \rangle_F}{m \| K \|_F},$$

since $\langle y y', y y' \rangle_F = m^2$.

Another measure of separation between classes is the average separation between two points in different classes, again normalised by the matrix norm.

**Definition 2 Cut Cost.** The cut cost of a clustering is defined as

$$C(S, k, y) = \frac{\sum_{i,j} y_i y_j k(x_i, x_j)}{m \| K \|_F}.$$
its mean, as proven in the companion paper [3]. This shows that the expected alignment can be reliably estimated from its empirical estimate $\hat{A}(S)$. As the cut cost can be expressed as the difference of two alignments

$$C(S, k, y) = 0.5(T(S, k) - A(S, k, y)),$$

it will be similarly concentrated around its expected value.

3 Optimising the cost with spectral techniques

In this section we will introduce and test two related methods for clustering, as well as their extensions to transduction. The general problem we want to solve is to assign class-labels to datapoints so as to maximize one of the two cost functions given above. By equation (1) the optimal solution to both problems is identical for a fixed data set and kernel. The difference between the approaches is in the two approximation algorithms developed for the different cost functions. The approximation algorithms are obtained by relaxing the discrete problems of optimising over all possible labelings of a dataset to closely related continuous problems solved by eigenvalue decompositions. See [5] for use of eigenvectors in partitioning sparse matrices.

3.1 Optimising the alignment

To optimise the alignment, the problem is to find the maximally aligned set of labels

$$\hat{A}^*(S, k) = \max_{y \in \{-1, 1\}^m} \hat{A}(S, k, y) = \max_{y \in \{-1, 1\}^m} \frac{\langle K, yy' \rangle_F}{\sqrt{\langle K, K \rangle_F}}.$$

Since in this setting the kernel is fixed maximising the alignment reduces to choosing $y \in \{ -1, 1 \}^m$ to maximise $\langle K, yy' \rangle = y^T Ky$. If we allow $y$ to be chosen from the larger set $\mathbb{R}^m$ subject to the constraint $\|y\|^2 = m$, we obtain an approximate maximum-alignment problem that can be solved efficiently. After solving the relaxed problem, we can obtain an approximate discrete solution by choosing a suitable threshold to the entries in the vector $y$ and applying the sign function. Bounds will be given on the quality of the approximations.

The solution of the approximate problem follows from the following theorem that provides a variational characterization of the spectrum of symmetric matrices.

**Theorem 3 (Courant-Fisher Minmax Theorem)** If $M \in \mathbb{R}^{m \times m}$ is symmetric, then for $k = 1, \ldots, m$,

$$\lambda_k(M) = \min_{\dim(T) = m - k \neq 0 \neq v \in T} \max_{v \in T} \frac{\langle Mv, v \rangle}{\langle v, v \rangle} = \min_{\dim(T) = m - k + 1} \max_{0 \neq v \in T} \frac{\langle Mv, v \rangle}{\langle v, v \rangle}.$$

If we consider the first eigenvector, the first min does not apply and we obtain that the approximate alignment problem is solved by the first eigenvector, so that the maximal alignment is upper bounded by a multiple of the first eigenvalue, $\lambda_{\max} = \max_{y \in \mathbb{R}^m} \frac{\langle Ky, y \rangle}{\langle y, y \rangle}$. One can now transform the vector $v$ into a vector in $\{-1, +1\}^m$ by choosing the threshold $\theta$ that gives maximum alignment of $y = \text{sign}(v^{\text{max}} - \theta)$.

By definition, the value of alignment $A(S, k, y)$ obtained by this $y$ will be a lower bound of the optimal alignment, hence we have

$$A(S, k, y) \leq \hat{A}^*(S, k) \leq \lambda_{\max}/\|K\|_F.$$

One can hence estimate the quality of a dichotomy by comparing its value with the upper bound. The absolute alignment tells us how specialized a kernel is on a given dataset: the higher this quantity, the more committed to a specific dichotomy.