

A NOVEL KERNEL METHOD for CLUSTERING ^{*}

Francesco Camastra and Alessandro Verri

INFM - DISI, Università di Genova, Via Dodecaneso 35, 16146 Genova, Italy
{camastra,verri}@disi.unige.it

Abstract. Kernel Methods are algorithms that implicitly perform a nonlinear mapping of the input data to a high dimensional Feature Space. In this paper, we present a novel Kernel Method, *Kernel K-Means* for clustering problems. Unlike other popular clustering algorithms that yield piecewise linear borders among data, Kernel K-Means allows to get nonlinear separation surfaces in the data. Kernel K-Means compares better with popular clustering algorithms, on a synthetic dataset and two UCI real data benchmarks.

1 Introduction

Kernel Methods [3] are algorithms that implicitly perform, by replacing the inner product with an appropriate Mercer Kernel, a nonlinear mapping of the input data to a high dimensional Feature Space. Powerful supervised Kernel Methods have been developed to solve classification and regression problems. As far as we know no effective Kernel Methods to solve clustering problems have been developed. In this paper, we present an effective Kernel Method, *Kernel K-Means* for clustering problems. Kernel K-Means maps data from the Input Space to a high dimensional Feature Space using a Mercer Kernel. Then it considers K centers and computes for each center the smallest ball that encloses the data that are closest. Kernel K-Means uses a K-Means-like strategy, i.e. it moves repeatedly the centers, computing for each center the smallest ball, until any center changes. Unlike other popular clustering algorithms that yield piecewise linear borders among data, Kernel K-Means allows to get nonlinear separation surfaces in the data. This is the main quality of the algorithm. The plan of the paper is as follows. In Section 2 we review K-Means that inspired Kernel K-Means and the Support Vector Clustering, the basic step of the Kernel K-Means; in Section 3 we present the Kernel K-Means algorithm; in Section 4 some experimental results are reported; finally some conclusions are drawn in Section 5.

2 Preliminaries

Let $D = (x_1, x_2, \dots, x_m)$ be a data set with vectors $x_i \in \mathbb{R}^N$.

We call *codebook* the set $W = (w_1, w_2, \dots, w_{k-1}, w_k)$ where each element $w_c \in$

^{*} This research has been partially funded by the FIRB Project ASTA

\mathbb{R}^N and $k \ll m$.

The *Voronoi Set* (V_c) of the codevector w_c is the set of all vectors in D for which w_c is the *nearest vector*. The most popular clustering technique is K-Means [7]. K-Means works by repeatedly moving all codevectors to the arithmetic mean of their Voronoi sets.

K-Means consists of the following steps:

1. Initialize the codebook W to contain K with vectors chosen *randomly* from the training set D .
2. Compute for each codevector $w_i \in W$ its Voronoi Set V_i .
3. Move each codevector w_i to the mean of its Voronoi Set.
4. Go to step 2 if any codevector, in the step 3, w_i has been changed.

K-Means is an *Expectation-Maximization* (EM) algorithm [4].

Since each EM algorithm is convergent, the convergence of the K-Means algorithm is guaranteed.

Support Vector Clustering (SVC) [1], called also *one-class SVM*, is a supervised Kernel Method based on support vector description of a data set. In Feature Space, SVC computes the smallest sphere that encloses the image of the input data.

Let $D = (x_i \in \mathbb{R}^N, i = 1, 2, \dots, m) \subseteq \mathcal{X}$, with $\mathcal{X} \subseteq \mathbb{R}^N$. We project the data x_i in some Feature Space \mathcal{F} using a nonlinear transformation $\Phi : \mathcal{X} \rightarrow \mathcal{F}$. Then we look for the smallest sphere of radius R , in the Feature Space, that encloses the data projections $\Phi(x_i)$. This is described by the constraints:

$$\|\Phi(x_j) - a\|^2 \leq R^2 \quad j = 1 \dots m,$$

where $\|\cdot\|$ is the Euclidean norm and a is the center of the sphere.

The constraints can be relaxed by using *slack* variables ξ_j :

$$\|\Phi(x_j) - a\|^2 \leq R^2 + \xi_j$$

with $\xi_j \geq 0$.

We solve the problem of finding the smallest sphere introducing the *Lagrangian*

$$L = R^2 - \sum_j^m (R^2 + \xi_j - \|\Phi(x_j) - a\|^2) \beta_j - \sum_j^m \xi_j \mu_j + C \sum_j^m \xi_j$$

where $\beta_j \geq 0$ and $\mu_j \geq 0$ are Lagrange multipliers, C is a constant and $C \sum_j^m \xi_j$ is a penalty term.

After differentiating w.r.t R , a , ξ_j we may eliminate the variables R , a and μ_j , turning the Lagrangian into a function \mathcal{W} of the variables β_j :

$$\mathcal{W} = \sum_j^m \Phi(x_j)^2 \beta_j - \sum_i^m \sum_j^m \beta_i \beta_j \Phi(x_i) \cdot \Phi(x_j).$$

The point $\Phi(x_j)$ can be classified as follows: if $\beta_j = 0$ it lies inside the surface; if $\beta_j = C$ it lies outside the sphere; if $0 < \beta_j < C$ it lies on the surface of the

Feature Space sphere. Such a point will be referred to as a *support vector (SV)*. Now we compute the inner products $\Phi(x_i) \cdot \Phi(x_j)$ by an appropriate Mercer kernel $G(x_i, x_j)$ (*kernel trick*). The usual choice is to use the Gaussian kernel. We have adopted this choice in the experimentations described in the Section 4. After using the kernel trick, the function \mathcal{W} , becomes

$$\mathcal{W} = \sum_j^m G(x_j, x_j)\beta_j - \sum_i^m \sum_j^m \beta_i\beta_j G(x_i, x_j).$$

It can be shown that the position of the center a can be unknown. Nevertheless, for each point x the distance $R(x)$ between the center a and its image in Feature Space $\Phi(x)$ can be computed:

$$R^2(x) = \|\Phi(x) - a\|^2 = G(x, x) - 2 \sum_j^m \beta_j G(x_j, x) + \sum_i^m \sum_j^m \beta_i\beta_j G(x_i, x_j).$$

3 Kernel K-Means

Since SVC can detect only one cluster, the goal of our research is to formulate a Kernel Method, based on the support vector description of the data set. We propose an algorithm, *Kernel K-Means* for clustering.

Given a data set D , we map our data in some Feature Space \mathcal{F} , by means a non-linear map Φ . Unlike SVC, we consider K , whose value is apriori fixed, centers in Feature Space ($a_i \in \mathcal{F} \quad i = 1, \dots, K$).

We call the set $A = (a_1, \dots, a_K)$ *Feature Space Codebook* since in our representation the centers in the Feature Space play the same role of the codevectors in the Input Space.

In analogy with the codevectors in the Input Space, we define for each center a_c its *Voronoi Set* in Feature Space.

The *Voronoi Set in Feature Space (FV_c)* of the center a_c is the set of all vectors x_i in D such that a_c is the *closest vector* for their images $\Phi(x_i)$ in the Feature Space

$$FV_c = \{x_i \in D \mid c = \arg \min_j \|\Phi(x_i) - a_j\|\}$$

Now we describe the *Kernel K-Means* algorithm. Kernel K-Means uses a K-Means-like strategy, i.e. moves repeatedly all centers a_c in the Feature Space, computing SVC on their FV_c , until any center changes.

In order to make more robust Kernel K-Means than K-Means with respect to the outliers SVC is computed on $FV_c(\rho)$ of each center a_c .

$FV_c(\rho)$ is defined as

$$FV_c(\rho) = \{x_i \in FV_c \text{ and } \|\Phi(x_i) - a_c\| < \rho\}.$$

$FV_c(\rho)$ is the Voronoi set in the Feature Space of the center a_c without outliers, that is the images of data points whose distance from the center is larger than

ρ .

The parameter ρ can be set up using model selection techniques [2].

Kernel K-Means has the following steps:

1. Project the data Set D in a Feature Space \mathcal{F} , by means a nonlinear mapping Φ . Initialize the centers $a_c \quad c = 1, \dots, K \quad a_c \in \mathcal{F}$
2. Compute for each center a_c $FV_c(\rho)$
3. Apply SVC to each $FV_c(\rho)$ and assign to a_c the center yielded, i.e.

$$a_c = SVC(FV_c(\rho))$$

4. Go to step 2 until any a_c changes, otherwise return the Feature Space codebook.

Kernel K-Means is an EM algorithm since its second and third step are respectively the expectation and maximization stage of an EM algorithm. Hence Kernel K-Means convergence is guaranteed, since each EM algorithm is convergent.

4 Experimental Results

Kernel K-Means has been tried on a synthetic data set (*Delta Set*) and on two UCI data sets, that is the *IRIS Data* [5] and the *Wisconsin's breast cancer* database [9].

Delta Set is a bidimensional set formed by 424 points of two classes nonlinearly separated. Therefore the two classes cannot be separated by clustering algorithms that use only two codevectors in the Input Space, since two codevectors permit only linear separation of the data.

To confirm that, we tried K-Means, using two codevectors, on Delta Set. As shown in the figure 1, K-Means cannot separate the clusters. K-Means shares this limitation with other not-kernel-based clustering algorithms, e.g. SOM [6] and Neural Gas [8]. Then we tried Kernel K-Means on Delta Set using only two centers. As shown in figure 1, Kernel K-Means can separate the two clusters, unlike other clustering algorithms.

Iris Data is formed by 150 points, that belong to three different classes. One

model	IRIS Data Points Classified Correctly	Wisconsin Database Points Classified Correctly
SOM	121.5 \pm 1.5 (81.0%)	660.5 \pm 0.5 (96.7%)
K-Means	133.5 \pm 0.5 (89.0%)	656.5 \pm 0.5 (96.1%)
Neural Gas	137.5 \pm 1.5 (91.7%)	656.5 \pm 0.5 (96.1%)
Kernel K-Means	142 \pm 1 (94.7%)	662.5 \pm 0.5 (97.0)%

Table 1. Average Kernel K-Means, SOM, K-Means and Neural Gas performances on IRIS Data and Wisconsin's breast cancer database. The results have been obtained using twenty different runs for each algorithm.

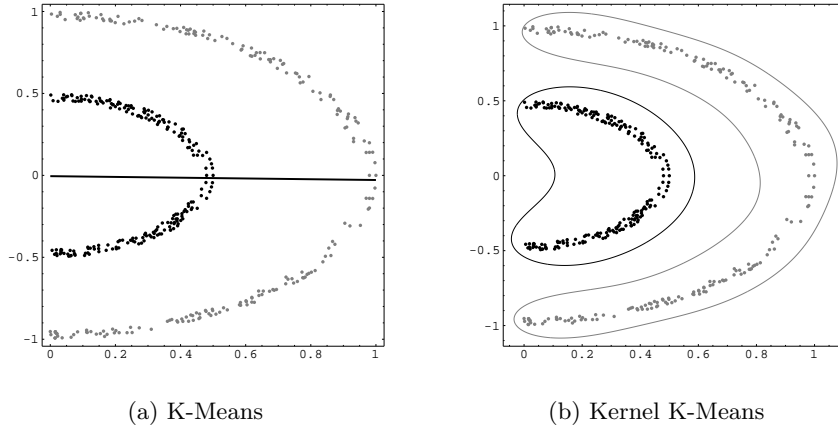


Fig. 1. (a) K-Means on Delta Set. The solid line indicates the separation line determined by K-Means. (b) Kernel K-Means on Delta Set. The region delimited by the black line identifies the input data whose images in the Feature Space have distance from the center a_1 less than 0.75. The region delimited by the gray line identifies the input data whose images in the Feature Space have distance from the center a_2 less than 0.84.

class is linearly separable from the other two, but the other two are not linearly separable from each other.

Wisconsin's breast cancer database collects 699 cases for such diagnostic samples. We have removed 16 database samples with missing values, therefore the database considered in the experiments has 683 patterns. The patterns belong to two different classes, the former has 444 samples, the latter has 239 samples. We tried Kernel K-Means, K-Means, Neural Gas and SOM on IRIS data and Wisconsin database, using respectively two and three centers. The table 1 shows the average performances of the algorithms on 20 runs, obtained changing algorithm initializations and parameters. As shown in the table, Kernel K-Means performances are better than other clustering algorithms on both datasets.

5 Conclusion

In this paper we have presented a novel clustering algorithm, Kernel K-Means. Under this aspect Kernel K-Means compares favourably with clustering algorithms such as *Self Organizing Maps* and *Neural Gas*, whose convergence is not guaranteed. Kernel Grower is a batch clustering algorithm, therefore its performance is not affected by the pattern ordering in the training set, unlike on-line clustering algorithms.

The main Kernel K-Means quality consists, unlike most clustering algorithms published in the literature, in producing nonlinear separation surfaces among

data. Kernel K-Means compares better with K-Means, Neural Gas and SOM, on a synthetic dataset and two UCI benchmarks. These results encourage the use of Kernel K-Means for the solution of real world problems.

References

1. A. Ben-Hur, D. Horn, H.T. Siegelmann, & V. Vapnik, "Support Vector Clustering", *Journal of Machine Learning Research*, 2, 125-137, 2001.
2. C. Bishop, *Neural Networks for Pattern Recognition*, Cambridge University Press, Cambridge (UK), 1995.
3. N. Cristianini & J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, Cambridge (UK), 2000.
4. A.P. Dempster, N.M. Laird & D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM algorithm", *Journal Royal Statistical Society*, 39(1), 1-38, 1977.
5. R.A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems", *Annals of Eugenics*, 7, 179-188, 1936.
6. T. Kohonen, "Self-Organized formation of topologically correct feature maps", *Biological Cybernetics*, 43, 59-69, (1982).
7. S.P. Lloyd, "An algorithm for vector quantizer design", *IEEE Transaction on Communications*, 28(1), 84-95, 1982.
8. T.E. Martinetz, & K.J. Schulten, "Neural-gas network for vector quantization and its application to time-series prediction", *IEEE Transaction on Neural Networks*, 4(4), 558-569, 1993.
9. W.H. Wolberg & O.L. Mangasarian, "Multisurface method of pattern separation for medical diagnosis applied to breast cytology", *Proceedings of the National Academy of Sciences, USA*, 87, 9193-9196, 1990.