Applying the Possibilistic C-Means Algorithm in Kernel-Induced Spaces

Maurizio Filippone, Francesco Masulli, and Stefano Rovetta

M. Filippone is with the Department of Computer Science of the University of Sheffield, 211 Portobello Street, Sheffield, S1 4DP. United Kingdom. email: m.filippone@dcs.shef.ac.uk

F. Masulli and S. Rovetta are with the Department of Computer and Information Science, University of Genova, Via Dodecaneso 35, I-16146 Genova, Italy.

Abstract

In this paper, we study a kernel extension of the classic possibilistic clustering. In the proposed extension, we implicitly map input patterns into a possibly high dimensional space by means of positive semidefinite kernels. In this new space, we model the mapped data by means of the Possibilistic Clustering algorithm. We study in more detail the special case where we model the mapped data using a single cluster only, since it turns out to have many interesting properties. The modeled memberships in kernel-induced spaces, yield a modeling of generic shapes in the input space. We analyze in detail the connections to One-Class SVM and Kernel Density Estimation, thus suggesting that the proposed algorithm can be used in many scenarios of unsupervised learning. In the experimental part, we analyze the stability and the accuracy of the proposed algorithm on some synthetic and real data sets. The results show high stability and good performances in terms of accuracy.

Index Terms

possibilistic clustering, kernel methods, outlier detection, regularization.

I. INTRODUCTION

Unsupervised learning is an important branch of Machine Learning dealing with the problem of analyzing unlabeled data. In this context, learning algorithms can provide useful insights about structures in data and produce results than can help the process of decision making. This situation occurs in several applications; popular tasks belonging to unsupervised learning are density estimation, clustering, and outlier detection. In density estimation, one is interested in modeling the probability density function generating the data. Clustering algorithms, instead, aim to find groups of data points that are similar on the basis of a (dis-)similarity criterion. Outlier detection identifies data points that share few similarities with the others.

Focusing on clustering, central clustering algorithms belonging to the K-means family [1] are widely used. All these algorithms are based on the concept of centroids and memberships, and the solution is obtained by solving an optimization problem. Centroids are also known as prototypes, or codevectors, and are representatives of the clusters. In this paper, data points will be also referred to as patterns. Memberships measure quantitatively the degree of belonging of patterns to clusters. Among the central clustering algorithms, we can find many modifications to the K-means algorithm. Popular fuzzy central clustering algorithms are the fuzzy versions of K-means with the probabilistic and possibilistic description of the memberships: Fuzzy c-means [2]

and Possibilistic c-means [3]. In many applications, the extension of the concept of membership from crisp to fuzzy is particularly useful. Let's consider some scenarios where clusters are overlapped or when data are contaminated by the presence of outliers. In such situations, it is more appropriate to allow pattern memberships to represent the degree of belonging to the clusters.

The main drawback of the possibilistic c-means, as well as of most central clustering methods, is its inability to model in a non-parametric way the density of clusters of generic shape (parametric approaches such as Possibilistic C-Spherical Shells [3], instead, have been proposed for some classes of shapes). This problem can be crucial in several applications, since the shapes of clusters are not hyper-spherical in general. Also, in all the central clustering algorithms belonging to the K-means family, it is needed to specify the number of clusters c in advance. In many cases, there is little information about the number of clusters; some methods have been proposed to find it automatically [2], but often it is required to run the algorithms for different values of c, and select the one that maximizes a suitable score function. In order to overcome these limitations, several modifications of the central clustering algorithms using kernels have been proposed [4].

In this paper, we study an extension of the classic possibilistic clustering by means of kernels[1]. In particular, we introduce the Possibilistic c-means (PCM) algorithm in kernel-induced spaces PCM_Φ, that is an application of the PCM proposed in Ref. [6] in the space induced by positive semidefinite kernels. As we will see shortly, the proposed extension is in the direction of providing a framework where both the shape and the number of clusters do not need to be specified, but only the spatial resolution at which data have to be analyzed. This extends the classes of problems where the possibilistic paradigm for data analysis can be employed.

In the classical PCM, the memberships modeling the data follow a Gaussian function, centered in the centroids, with covariance matrix proportional to the identity matrix. In the proposed extension, we implicitly map input patterns into a possibly high dimensional space by means of kernels. In this new space, also known as *feature space*, we model the mapped data by means of the PCM algorithm. We make use of the theory of positive semidefinite kernels to show how it is possible to obtain an iterative algorithm for the computation of the memberships of the input

[1]The algorithm has been proposed in Ref. [5]; in this paper, we report more theoretical results and experimental validations

data points. Effectively, the resulting algorithm models patterns in feature space by means of memberships that follow a Gaussian distribution centered in the centroids in the feature space.

We note here that another possibilistic clustering algorithm making use of kernels has been proposed [4], [7]. It belongs, however, to the family of methods where kernels are used to compute distances between the centroids and the patterns. This technique is the so called *kernelization of the metric* and differs substantially from the technique we present here. In other words, in those algorithms the centroids lie in the input space and kernels play a role only in the computation of distances. In the proposed method, instead, kernels induce an implicit mapping of the input patterns and the algorithm is applied in such a new space; therefore, the centroids will live in the induced space as well.

Although PCM_Φ is an important extension of the classical PCM algorithm, we realize that in practical applications the lack of competition among clusters leads all the centroids in feature space to collapse into a single one. This property of the PCM algorithm characterizes the possibilistic paradigm and is a direct consequence of the lack of probabilistic constraint on the memberships. Therefore, we propose a more detailed study of PCM_Φ, where we model the mapped data using a single cluster only. The One Cluster PCM in feature space (1-PCM_Φ) turns out to have many interesting properties. Remarkably, we show that the objective function optimized by 1-PCM_Φ is closely related to that of One-Class SVM (1-SVM). Also, we show that the role of the memberships in 1-PCM_Φ is dual with respect to the Lagrange multipliers in 1-SVM, and the objective function contains a further term that works as regularizer; both these facts give good robustness properties to the proposed algorithms, as we will see shortly. 1-PCM_Φ models the memberships of data points in feature space by means of a Gaussian; in the input space, this results in a non-linear modeling of densities. In fact, the resulting density in input space is expressed in terms of memberships and cannot be thought in probabilistic terms, since it is not a proper probability density function. Despite that, we can still make use of the memberships to obtain a quantitative measure on the density of regions in the input space. We provide an approximate result, however, showing the formal connections with Kernel Density Estimation (KDE). The modeling stage by means of the memberships leads naturally to a clustering algorithm in the input space where we connect the regions of the space where the memberships are above a selected threshold. Finally, the analysis of the memberships can be used to obtain an outlier detection algorithm; patterns having low membership with respect

to others lie in low density regions, and can be considered as outliers.

In the experimental part, we analyze the behavior of the proposed algorithm on some applications. We first show an example of density estimation and clustering. Then, we introduce a test of stability for outlier detection based on [8]. We modify such test to compare 1-PCM_Φ, 1-SVM, and KDE for outlier detection, by making use of a score based on the Jaccard coefficient. Finally, we compare stability and accuracy of 1-PCM_Φ, 1-SVM, and KDE on three real data sets.

The paper is organized as follows: In Section III we briefly review the classical PCM, in Section III we introduce the kernel extension of PCM, in Section IV we study the connections of the proposed model with 1-SVM and KDE, and in Section V we report the experimental analysis. Finally, we report the conclusions in Section VI.

II. Possibilistic Clustering

Given a set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ of n patterns $\mathbf{x}_i \in \mathbb{R}^d$, the set of centroids $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_c\}$ and the membership matrix U are defined. The set V contains the prototypes/representatives of the c clusters. U is a $c \times n$ matrix where each element u_{ih} represents the membership of the pattern h to the cluster i. In the PCM, $u_{ih} \in [0, 1]$ and memberships of a pattern to all the c clusters are not constraint to sum up to one. In other words, in the possibilistic clustering, the following constraint:

$$\sum_{i=1}^{c} u_{ih} = 1 \quad \forall k = 1, \ldots, n \tag{1}$$

also known as *Probabilistic Constraint*, is relaxed, leading to an interpretation of the membership as a degree of typicality.

In general, all the K-means family algorithms are based on the minimization of an objective function based on a measure of the distortion (or intra-cluster distance), that can be written as:

$$G(U, V) = \sum_{i=1}^{c} \sum_{h=1}^{n} u_{ih}^{\theta} \|\mathbf{x}_h - \mathbf{v}_i\|^2 \tag{2}$$

with $\theta \geq 1$. Also, an entropy term $H(U)$ can be added to the objective function to avoid trivial solutions where all the memberships are zero or equally shared among the clusters. For the algorithms having a constraint on U, the Lagrange multipliers technique has to be followed in order to perform the optimization, leading to a further term in the objective function that is also called Lagrangian (for a complete derivation of some central clustering algorithms based on this concept see [9]).

The technique used by these methods to perform the minimization is the so called Picard iterations technique [2]. The Lagrangian $L(U,V)$ depends on two groups of variables U and V related to each other, namely $U = U(V)$ and $V = V(U)$. In each iteration one of the two groups of variables is kept fixed, and the minimization is performed with respect to the other group. In other words:

$$\frac{\partial L(U,V)}{\partial \mathbf{v}_i} = 0 \tag{3}$$

with U fixed, gives a formula for the update of the centroids $\mathbf{v}_i$, and:

$$\frac{\partial L(U,V)}{\partial u_{ih}} = 0 \tag{4}$$

with V fixed, gives a formula for the update of the memberships u_{ih}. The algorithms start by randomly initializing U or V, and iteratively update U and V by means of the previous two equations. It can be proved that the value of L does not increase after each iteration [10]. The algorithms stop when a convergence criterion is satisfied on U, V or G. For instance, the following stopping criterion can be considered:

$$\|U - U'\|_p < \varepsilon \tag{5}$$

where U' is the updated version of the memberships and $\|\|_p$ is a p-norm.

The objective function of the PCM does not contain any terms due to the probabilistic constraint, thus becoming [6]:

$$L(U,V) = \sum_{h=1}^{n}\sum_{i=1}^{c} u_{ih}\|\mathbf{x}_h - \mathbf{v}_i\|^2 + \sum_{i=1}^{c}\eta_i \sum_{h=1}^{n}(u_{ih}\ln(u_{ih}) - u_{ih}) \tag{6}$$

The second term in the equation is an entropic term that penalizes small values of the memberships.

Setting to zero the derivatives of $L(U,V)$ with respect to the memberships u_{ih}:

$$\frac{\partial L(U,V)}{\partial u_{ih}} = \|\mathbf{x}_h - \mathbf{v}_i\|^2 + \eta_i \ln(u_{ih}) = 0 \tag{7}$$

we obtain:

$$u_{ih} = \exp\left(-\frac{\|\mathbf{x}_h - \mathbf{v}_i\|^2}{\eta_i}\right) \tag{8}$$

Setting to zero the derivatives of $L(U,V)$ with respect to $\mathbf{v}_i$, we obtain the update formula for the centroids $\mathbf{v}_i$:

$$\mathbf{v}_i = \frac{\sum_{h=1}^{n} u_{ih}\mathbf{x}_h}{\sum_{h=1}^{n} u_{ih}} \tag{9}$$

It has been suggested [6] that the value of η_i can be estimated as:

$$\eta_i = \gamma \frac{\sum_{h=1}^{n} u_{ih} \|\mathbf{x}_h - \mathbf{v}_i\|^2}{\sum_{h=1}^{n} u_{ih}} \tag{10}$$

Intuitively, η_i is an estimate of the spread of the i-th cluster, and γ can be set to have a better control on it.

III. Possibilistic Clustering in Feature Space

In this Section, we extend the possibilistic approach to clustering in kernel induced spaces PCM_Φ. It consists in the application of the PCM in the feature space $\mathcal{F}$ obtained by a mapping Φ from the input space S ($\Phi : S \to \mathcal{F}$). The objective function to minimize is then:

$$L^\Phi(U, V^\Phi) = \sum_{h=1}^{n} \sum_{i=1}^{c} u_{ih} \|\Phi(\mathbf{x}_h) - \mathbf{v}_i^\Phi\|^2 + \sum_{i=1}^{c} \eta_i \sum_{h=1}^{n} (u_{ih} \ln(u_{ih}) - u_{ih}) \tag{11}$$

Note that the centroids $\mathbf{v}_i^\Phi$ of PCM_Φ algorithm lie in the feature space. We can minimize $L^\Phi(U, V^\Phi)$ by setting its derivatives with respect to $\mathbf{v}_i^\Phi$ and u_{ih} equal to zero, obtaining:

$$\mathbf{v}_i^\Phi = b_i \sum_{h=1}^{n} u_{ih} \Phi(\mathbf{x}_h), \qquad b_i \equiv \left(\sum_{h=1}^{n} u_{ih} \right)^{-1} \tag{12}$$

$$u_{ih} = \exp\left(-\frac{\|\Phi(\mathbf{x}_h) - \mathbf{v}_i^\Phi\|^2}{\eta_i} \right). \tag{13}$$

In principle, the necessary conditions in Eq.s 12 and 13 can be used for a Picard iteration minimizing $L^\Phi(U, V^\Phi)$. Let's consider Mercer Kernels [11], i.e. symmetric and positive semidefinite kernels; they can be expressed as:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^{\mathrm{T}} \Phi(\mathbf{x}_j) \tag{14}$$

Note that the choice of K implies Φ; for many kernel functions, the mapping Φ is implicit (and possibly high dimensional). In this case, this means that we cannot compute the centers $\mathbf{v}_i^\Phi$ explicitly. Despite that, we can obtain an optimization scheme by making use of the properties of kernels. Eq. 14 yields the following, also known as *kernel trick* [12]:

$$\|\Phi(\mathbf{x}_i) - \Phi(\mathbf{x}_j)\|^2 = K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j) \tag{15}$$

The last equation shows that it is possible to compute the distance between mapped patterns without knowing explicitly Φ; this is a crucial aspect in algorithms using kernels [13]. Distances

in $\mathcal{F}$ are only function of the kernel function between input data. In our case, the kernel trick allows us to obtain an update rule for the memberships by plugging Eq. 12 into Eq. 13:

$$u_{ih} = \exp\left[-\frac{1}{\eta_i}\left(k_{hh} - 2b_i\sum_{r=1}^{n} u_{ir}k_{hr} + b_i^2\sum_{r=1}^{n}\sum_{s=1}^{n} u_{ir}u_{is}k_{rs}\right)\right]. \tag{16}$$

Note that in Eq. 16 we introduced the notation $k_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. The Picard iteration then reduces to the iterative update of the memberships only by using Eq. 16. We can stop the iterations when an assigned stopping criterion is satisfied (e.g., when memberships change less than an assigned threshold, or when no significant improvements of $L^{\Phi}(U, V^{\Phi})$ are noticed).

For what concerns the parameters η_i, we can apply in feature space the same criterion suggested for the PCM obtaining:

$$\eta_i = \gamma\ b_i\ \sum_{h=1}^{n} u_{ih}\left(k_{hh} - 2b_i\sum_{r=1}^{n} u_{ir}k_{hr} + b_i^2\sum_{r=1}^{n}\sum_{s=1}^{n} u_{ir}u_{is}k_{rs}\right) \tag{17}$$

The parameters η_i can be estimated at each iteration or once at the beginning of the algorithm. In the latter case the initialization of the memberships, that allows to provide a good estimation of the η_i, can be obtained as a result of a Kernel Fuzzy c-Means [14].

Note that if we choose a linear kernel $k_{ij} = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j$, PCM_{Φ} reduces to the standard PCM. Indeed, using a linear kernel is equivalent to set $\Phi \equiv I$, where I is the identity function. In the following, we will consider the Gaussian kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{18}$$

that is characterized by the fact that the induced mapping Φ maps the data space to an infinite dimensional feature space $\mathcal{F}$ [15], and by the following:

$$\|\Phi(\mathbf{x}_i)\|^2 = \Phi(\mathbf{x}_i)^{\mathrm{T}}\Phi(\mathbf{x}_i) = k_{ii} = 1. \tag{19}$$

As a consequence, patterns are mapped by the Gaussian kernel from data space to the surface of a unit hyper-sphere in feature space. Centroids $\mathbf{v}_i^{\Phi}$ in $\mathcal{F}$, instead, are not constrained to the hyper-spherical surface. Therefore, centroids would lie inside this hyper-sphere, and due to the lack of competition among clusters, they often collapse into a single one, with slight dependency on the value of the cluster spreads η_i. This effect is a direct consequence of the lack of probabilistic constraint, and characterizes the possibilistic clustering framework [6], [16]. Such a drawback motivates our analysis of the case where we model data in feature space by means of a single cluster only, namely where we set $c = 1$.

IV. One Cluster PCM in kernel-induced spaces

In this Section, we study the connections between the PCM_Φ with $c = 1$, that we will call the One Cluster PCM in feature space 1-PCM_Φ and the One-Class SVM (1-SVM). In particular, we show the formal analogies between the two objective functions, highlighting the robustness of the proposed method against 1-SVM. We will also show a connection between 1-PCM_Φ and Kernel Density Estimation (KDE).

A. One-Class SVM

One among the approaches using kernels in unsupervised learning, is based on the support vector description of data [13], [17]. We will start by following the presentation given in Ref. [17] based on the Support Vector Domain Description (SVDD). The aim of this approach is to look for an hyper-sphere with center $\mathbf{v}$ containing almost all data, namely allowing some outliers. Such approach leads to possibly non-linear surfaces separating the clusters in the input space.

The optimization problem is the following:

$$\min_{\alpha_1,\ldots,\alpha_n} \left(\sum_{r=1}^{n} \sum_{s=1}^{n} \alpha_r \alpha_s k_{rs} - \sum_{h=1}^{n} \alpha_h k_{hh} \right) \qquad \text{subject to :}$$

$$\sum_{h=1}^{n} \alpha_h = 1 \qquad \text{and} \qquad 0 \leq \alpha_h \leq C$$

The variables α_i are the Lagrange multipliers that are introduced in the constrained optimization problem. The optimization stage is carried out by a quadratic program that yields a sparse solution. In other words, many α_i result to be zero, thus providing a compact representation of the data set. This aspect is very important from the computational point of view [13]. At the end of the optimization, the following facts hold:

- when $\alpha_h = C$, the image of $\mathbf{x}_h$ lies outside the hyper-sphere. These points are called *bounded support vectors* and are considered *outliers*;
- when $0 < \alpha_h < C$, the image of $\mathbf{x}_h$ lies on the surface of the hyper-sphere. These points are called *support vectors*.
- when $\alpha_h = 0$, the image of $\mathbf{x}_h$ is inside the hyper-sphere.

The computation of the center of the sphere is a linear combination of the mapped patterns, weighted by the Lagrange multipliers:

$$\mathbf{v} = \sum_{h=1}^{n} \alpha_h \Phi(\mathbf{x}_h) \tag{20}$$

The last expression, combined with the kernel trick, leads to the computation of the distance between a pattern and the center:

$$d_h = \|\Phi(\mathbf{x}_h) - \mathbf{v}\|^2 = k_{hh} - 2\sum_{r=1}^{n} \alpha_r k_{hr} + \sum_{r=1}^{n}\sum_{s=1}^{n} \alpha_r \alpha_s k_{rs} \tag{21}$$

The radius R is the distance between a support vector and the center $\mathbf{v}$.

In Ref. [18] it has been proposed an SVM-based approach to separate data in feature space from the origin by means of an hyper-plane. Interestingly, in the case of kernels that are functions of difference between patterns (as in the Gaussian case, for example), the two approaches yield the same optimization problem. In Ref. [18], the parameter ν is used in place of C, since it has a more direct interpretation on the fraction of the outliers. In particular, the relation between the two parameters is:

$$C = \frac{1}{n\,\nu}$$

with $\nu \in [0, 1]$. In this parameterization, it can be proved that ν gives the upper bound on the fraction of outliers *and* a lower bound on the fraction of support vectors on the data set [18]. In the remainder of this paper, we will refer to these algorithms as 1-SVM, and we will use ν for the parameterization[2].

B. One-Cluster PCM in Feature Space

We show now an alternative view of the optimization problem of 1-PCM_Φ, starting from a formulation in input space to keep the notation uncluttered. Let's consider PCM_Φ with $c = 1$. We represent the memberships as a vector $\mathbf{u}$, where u_h is the membership of the h-th pattern to the cluster.

The objective function of 1-PCM_Φ becomes:

$$L = \sum_{h=1}^{n} u_h \|\mathbf{x}_h - \mathbf{v}\|^2 + \eta \sum_{h=1}^{n} (u_h \ln(u_h) - u_h) \tag{22}$$

[2]We used the implementation of 1-SVM in the R package e1071, that is based on LIBSVM [19].

The possibilistic constraint on the memberships is the following:

$$0 \leq u_h \leq 1 \tag{23}$$

Setting to zero the derivatives of L with respect to $\mathbf{v}$:

$$\frac{\partial L}{\partial \mathbf{v}} = -\sum_{h=1}^{n} u_{ih} \left(\mathbf{x}_h - \mathbf{v}\right) = 0 \tag{24}$$

we obtain the update formula for the centroid $\mathbf{v}$:

$$\mathbf{v} = \frac{\sum_{h=1}^{n} u_h \mathbf{x}_h}{\sum_{h=1}^{n} u_h} \tag{25}$$

Substituting $\mathbf{v}$ in L, and expanding the norm, we obtain:

$$\begin{aligned} L &= \sum_{h=1}^{n} u_h \|\mathbf{x}_h - \mathbf{v}\|^2 + \eta \sum_{h=1}^{n} \left(u_h \ln(u_h) - u_h\right) \\ &= \sum_{h=1}^{n} u_h \mathbf{x}_h^{\mathrm{T}} \mathbf{x}_h - \frac{\sum_{r=1}^{n} \sum_{s=1}^{n} u_r u_s \mathbf{x}_r^{\mathrm{T}} \mathbf{x}_s}{\sum_{h=1}^{n} u_h} + \eta \sum_{h=1}^{n} \left(u_h \ln(u_h) - u_h\right) \end{aligned}$$

The last equation can be extended by means of positive semidefinite kernels, leading to the following optimization problem:

$$\min \left(\sum_{h=1}^{n} u_h k_{hh} - \frac{\sum_{r=1}^{n} \sum_{s=1}^{n} u_r u_s k_{rs}}{\sum_h u_h} + \eta \sum_{h=1}^{n} \left(u_h \ln(u_h) - u_h\right) \right) \qquad \text{subject to :}$$

$$0 \leq u_k \leq 1$$

With this extension, the proposed algorithm models all data points by means of a single cluster in $\mathcal{F}$. If we add the constraint $\sum_h u_h = 1$, the problem becomes the following:

$$\min \left(\sum_{h=1}^{n} u_h k_{hh} - \sum_{r=1}^{n} \sum_{s=1}^{n} u_r u_s k_{rs} + \eta \sum_{h=1}^{n} u_h \ln(u_h) \right) \qquad \text{subject to :}$$

$$0 \leq u_h \leq 1 \qquad \text{and} \qquad \sum_{h=1}^{n} u_h = 1$$

In the Appendix, we will show that the introduction of the last constraint does not change the results of the optimization procedure, since it just corresponds to scale the values of the memberships (and the position of the centroid is not affected by that). This result shows that the objective function of the 1-PCM_Φ is closely related to that of 1-SVM. The center $\mathbf{v}$ in both cases is a linear combination of the mapped patterns; in 1-SVM the weights of the sum are provided by the Lagrange multipliers α_h, whereas in 1-PCM_Φ by the memberships u_h.

We notice, however, that the role of the α_h is the dual with respect to the u_h. In 1-SVM the values of α_h, and therefore the weights of the sum, are high for the outliers; in 1-PCM$_\Phi$ the memberships are high for patterns in regions of high density. The result is that in 1-SVM the center of the sphere is computed as combination of outliers, whereas in 1-PCM$_\Phi$, the center of the Gaussian modeling the data is computed as a combination of typical patterns. This can lead to a more reliable estimation for the centroid $\mathbf{v}$ in 1-PCM$_\Phi$. Moreover, in 1-PCM$_\Phi$ we can see the presence of a regularization term, which is an entropy based score of the memberships. In the experimental analysis, we will see that these properties give to the proposed method good performances in terms of robustness.

We note that the algorithms we are comparing are based on different ideas. 1-SVM looks for the center $\mathbf{v}$ and the radius R of the enclosing sphere, 1-PCM$_\Phi$ looks for a centroid in feature space and computes the memberships on the basis of $\mathbf{v}$. The parameter η works as the width of the membership function, and corresponds to the square of the radius R^2. 1-PCM$_\Phi$ yields the memberships of the patterns, and it is possible to set a threshold to obtain a decision boundary. This corresponds to select a sphere in feature space that is the intersection between the multivariate Gaussian describing the memberships and the hyper-plane corresponding to a specific threshold on the membership.

We report here the resulting update equation, representing the core part of the 1-PCM$_\Phi$ (in the unconstrained case):

$$u_h = \exp\left[-\frac{1}{\eta}\left(k_{hh} - 2b\sum_{r=1}^{n} u_r k_{hr} + b^2 \sum_{r=1}^{n}\sum_{s=1}^{n} u_r u_s k_{rs}\right)\right], \qquad b \equiv \left(\sum_{h=1}^{n} u_h\right)^{-1} \tag{26}$$

The iterative application of such equation leads to a solution of the optimization problem of the 1-PCM$_\Phi$. The parameter η can be estimated from the data set in the following way:

$$\eta = \gamma\ b\ \sum_{h=1}^{n} u_h \left(k_{hh} - 2b\sum_{r=1}^{n} u_r k_{hr} + b^2 \sum_{r=1}^{n}\sum_{s=1}^{n} u_r u_s k_{rs}\right) \tag{27}$$

As we have just seen, η can be also interpreted as a regularization parameter. Therefore, the value of γ can be set so as to enhance the regularization properties of the algorithm. The whole derivation of the update equation, along with a discussion about the role played by the constraint on the sum of the memberships, can be found in the Appendix. Before closing this section, we

report the equation allowing to compute the membership value for a test point $\mathbf{x}_*$:

$$u(\mathbf{x}_*) = \exp\left[-\frac{1}{\eta}\left(K(\mathbf{x}_*,\mathbf{x}_*) - 2b\sum_{r=1}^{n} u_r K(\mathbf{x}_*,\mathbf{x}_r) + b^2\sum_{r=1}^{n}\sum_{s=1}^{n} u_r u_s K(\mathbf{x}_r,\mathbf{x}_s)\right)\right] \tag{28}$$

The elements of $\mathbf{u}$ in Eq. 28 are the memberships of the training points obtained after the training stage, and b is the inverse of the sum of the u_h (Eq. 26). Eq. 28 can be readily obtained from $u(\mathbf{x}_*) = \exp(-\frac{1}{\eta}\|\Phi(\mathbf{x}_*) - \mathbf{v}^\Phi\|)$, by expanding $\mathbf{v}^\Phi$ in terms of the mapped training data points and using the kernel trick.

C. Connections to Kernel Density Estimation

Kernel Density Estimation (KDE) is a non-parametric method that yields a probability density function (pdf) given a set of observations $\{x_1,\ldots,x_n\}$ [20]. For the sake of presentation, let $x_i \in \mathbb{R}$. The resulting pdf is the sum of kernel functions centered in the data points. In the simplest form of KDE, the weights given to the kernels are equal, as well as the parameters of the kernels $\mathcal{G}$:

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{G}(x,x_i) \tag{29}$$

where $\mathcal{G}(x,x_i)$ is a kernel function such that:

$$\mathcal{G}(x,x_i) \geq 0 \quad \forall x,x_i \in \mathbb{R} \qquad \int_{\mathbb{R}} \mathcal{G}(x,x_i)dx = 1 \tag{30}$$

Despite its simplicity, this form of KDE has nice theoretical properties in terms of consistency [20]. Several modifications have been proposed to KDE, in order to improve the performances in applications; in particular, the weighted KDE assigns a different weight to the kernels:

$$\hat{p}(x) = \sum_{i=1}^{n} w_i \mathcal{G}(x,x_i) \tag{31}$$

where $\sum_i w_i = 1$.

We now give an interesting interpretation of the 1-PCM_Φ, in the context of KDE. Let's rewrite Eq. 26 showing explicitly the dependence from a test point x and considering kernels that are functions of the difference between the arguments:

$$u(x) = \psi \exp\left[\frac{2b}{\eta}\sum_{r=1}^{n} u_r K(x,x_r)\right]. \tag{32}$$

TABLE I

PSEUDO-CODE OF THE CORE PART OF 1-PCM$_\Phi$

1) Initialize the kernel parameter σ, and the parameter γ;
2) Initialize all the memberships $u_h = 1/n$;
3) Compute the regularization parameter η using Eq. 27;
4) Initialize the convergence parameter ε;
5) **repeat**
   a) Update the memberships u_h using Eq. 26
   b) Compute $\delta = \sum_{h=1}^{n} |u_h - u'_h|$;
6) **until** $(\delta < \varepsilon)$

where ψ is a multiplicative term that is independent from x. If we consider a test point x_* that is far away from all the training points, its membership would be $u(x_*) = \psi$, since all the values $K(x, x_r) \simeq 0$. In order to turn the memberships into probabilities, we would need to set the probability of x_* to zero. This suggests to consider:

$$f(x) = u(x) - u(x_*) = \psi \left(\exp \left[\frac{2b}{\eta} \sum_{r=1}^{n} u_r K(x, x_r) \right] - 1 \right) \tag{33}$$

A first order approximation of the exponential gives:

$$f(x) \simeq \sum_{r=1}^{n} w_r K(x, x_r) \tag{34}$$

where we absorbed all the constants and the normalization terms needed to make $f(x)$ integrate to one over $\mathbb{R}$ into the weights w_r. Note also that when η is very large, all the memberships tend to one (see Eq. 26). Therefore, in this limit the weights of the approximation become equal, leading to the KDE solution:

$$f(x) \simeq \frac{1}{n} \sum_{r=1}^{n} K(x, x_r) \tag{35}$$

D. Applications of 1-PCM$_\Phi$

The *Core* part of the algorithm produces a fuzzy-possibilistic model of densities (membership function) in the feature space. It is initialized by selecting a *stop criterion* (e.g., when memberships change less than an assigned threshold, or when no significant improvements of $L^\Phi(U, V^\Phi)$

are noticed), setting the value of σ for the Gaussian kernel (in order to define the spatial resolution of density estimation), and initializing the memberships u_h. Then, after estimating the value of η using Eq. 17, we perform the Picard iterations using Eq. 26. In absence of prior knowledge on the data set, we suggest to set all the memberships to the same value. Note also that the initialization value of the memberships is arbitrary. This can be easily seen by noticing that in fact the first iteration updates the centroid $\mathbf{v}$ and the memberships in one step via Eq. 26. The centroid $\mathbf{v}$ is implicitly computed as a weighted combination of the mapped patterns $\Phi(\mathbf{x}_r)$, where the weights are the memberships divided by their sum. Therefore, if we initialize the memberships to the same value, their sum does not influence the implicit computation of $\mathbf{v}$ that is the used to compute the updated version of the memberships.

Density Estimation: At the end of the *Core* step, we have modeled the density of patterns in feature space. These memberships, back to the input space, represent a density estimation in input space based on a specific kernel. Again, we stress that the density estimation is expressed in terms of memberships, and it cannot be interpreted as a density in a probabilistic sense. The value of the parameter η plays the role of a scaling factor on the range of membership values that can be obtained by the algorithm.

Outlier Detection: Once the memberships are obtained, it is possible to select a threshold $\alpha \in (0, 1)$ and use it to define an α-*cut* (or α-*level set*) on data points:

$$A_\alpha = \{\mathbf{x}_h \in X \mid u_h > \alpha\} \tag{36}$$

This can be considered as a *Defuzzification* step. Note that given the form of u_h (Eq. 13) the threshold α defines a hyper-circle which encloses a hyper-spherical cap. A_α is then the set of data points whose mapping in feature space lies on the cap, whose base radius depends on α. Points outside the α-cut are considered to be outliers. We can set α on the basis of the rejection rate that we are interested in by using the quantiles of the histogram of the memberships.

When we assume that we are dealing with a training set without outliers, the rejection rate can be set as a measure of the false positive rate. This is because some "normal" data points would still fall in the region where their membership is lower than the threshold. This procedure is similar to setting a confidence level in statistical testing.

When training data are contaminated by the presence of outliers, it is necessary to specify their fraction with respect to the size of the training set. From the analysis of the histogram of

the memberships it is possible to obtain a rough estimate on the number of outliers, since they will have far lower memberships than the normal patterns.

Clustering: Once we have the results from the *Core* part of the algorithm, we can perform clustering by applying an idea similar to that in Support Vector Clustering [21]. It uses a convexity criterion derived from the one proposed for 1-SVM [21] assigning the same label to a pair of points only if all elements of the linear segment joining the two points in data space belong to A_α. In order to check that the points of the linear segment belong to A_α, we compute the memberships of a set of them (typically twenty [21]) using Eq. 28. If none of the selected points has membership below the selected threshold α, two points will be considered belonging to the same cluster. In practice, we construct an unweighted undirected graph, where the nodes are the data points, and an arc connects two nodes when the corresponding data points have a joining linear segment in the data space that belongs to A_α. The labeling procedure amounts in finding the connected components of such a graph, assigning the same labels to the nodes, and therefore to the data points, in the same connected component of the graph. This procedure separates the data points belonging to the single cluster in feature space, in a set of non-convex clusters in data space, thus avoiding the need to specify the number of clusters in advance. We will illustrate this procedure with a simple example in the experimental section. The selection of α can follow different approaches. In our experience, we found that α can be set, as in outlier detection, on the basis of how many patterns we intend to reject from the training; the computation can be performed by looking at the quantiles of the memberships of the training points.

We recall here the formal analogy between KDE and 1-PCM$_\Phi$ in the case of kernels that are functions of the difference between the arguments. In such cases, we might as well use KDE for modeling densities, clustering, and outlier detection in the same spirit of 1-PCM$_\Phi$. In KDE, we would have a modeling in terms of probabilities of data points instead of memberships, and we could still mimic the procedures to achieve clustering or outlier detection. We note, however, that the applicability of 1-PCM$_\Phi$ is more general than KDE. As a simple example, we can consider the case of a linear kernel. In such a case, 1-PCM$_\Phi$ is equivalent to modeling the data with a single Gaussian in the data space, whereas there is no corresponding KDE solution. In general, 1-PCM$_\Phi$ requires only that kernel values among training data and kernel values between training and test data are available; this is always the case when pairwise dissimilarities are available among data points [9]. Also, any positive semidefinite kernel can be employed, depending on the

modeling requirements of the system, since the kernel function implies the mapping Φ. KDE is applied to data represented in terms of feature vectors and kernels are functions of the difference (see e.g. [20]) or scalar product (when data are on hyper-spherical surfaces [22]) between data points.

V. Experimental Analysis

In this Section, we report the experimental analysis showing the properties of 1-PCM_Φ. We first show its ability to model densities and to perform clustering on an illustrative example. In the second part, we focus on a comparison of the stability and the accuracy of 1-PCM_Φ with 1-SVM and KDE in the context of outlier detection.

A. Density Estimation and Clustering

As an example of use of 1-PCM_Φ for estimation of densities and clustering, we applied the proposed algorithm to the data set shown in Fig. 1. The data set is composed by six clusters of different shapes and densities, and some outliers. In particular, the spherical, the banana-shaped, ring-shaped clusters contain respectively 30 (each of the four spherical clusters), 60, and 80 points; the number of outliers is 30. We run our algorithm using a Gaussian kernel, and setting $\gamma = 1$. The stop criterion was $\sum_h |\Delta u_h| < \varepsilon$ with $\varepsilon = 0.01$. In Fig. 1, we can see the role played by the parameter σ of the kernel. The first row shows the contour plot of the memberships for $\sigma = 1$ and $\sigma = 2$. The left plot of the second row of Fig. 1 shows the case $\sigma = 0.5$. It is possible to see how σ selects the spatial resolution in the analysis of densities. Selecting a rejection rate of 10%, we computed the corresponding quantiles of the memberships (in the case $\sigma = 0.5$), thus obtaining a decision boundary in the input space. As we can see in the right plot of the second row of Fig. 1, the resulting boundary identifies correctly the shapes of the clusters. The labeling step would yield six clusters corresponding to the six connected regions and the outliers (denoted by crosses).

As shown in this experiment, 1-PCM_Φ shows robustness to outliers and the capability to model clusters of generic shape in the data space (modeling their distributions in terms of fuzzy memberships). Moreover, it is able to find *autonomously* the *natural* number of clusters in the data space. The outliers rejection ability is shared also by the PCM, but is limited to the case of globular clusters.

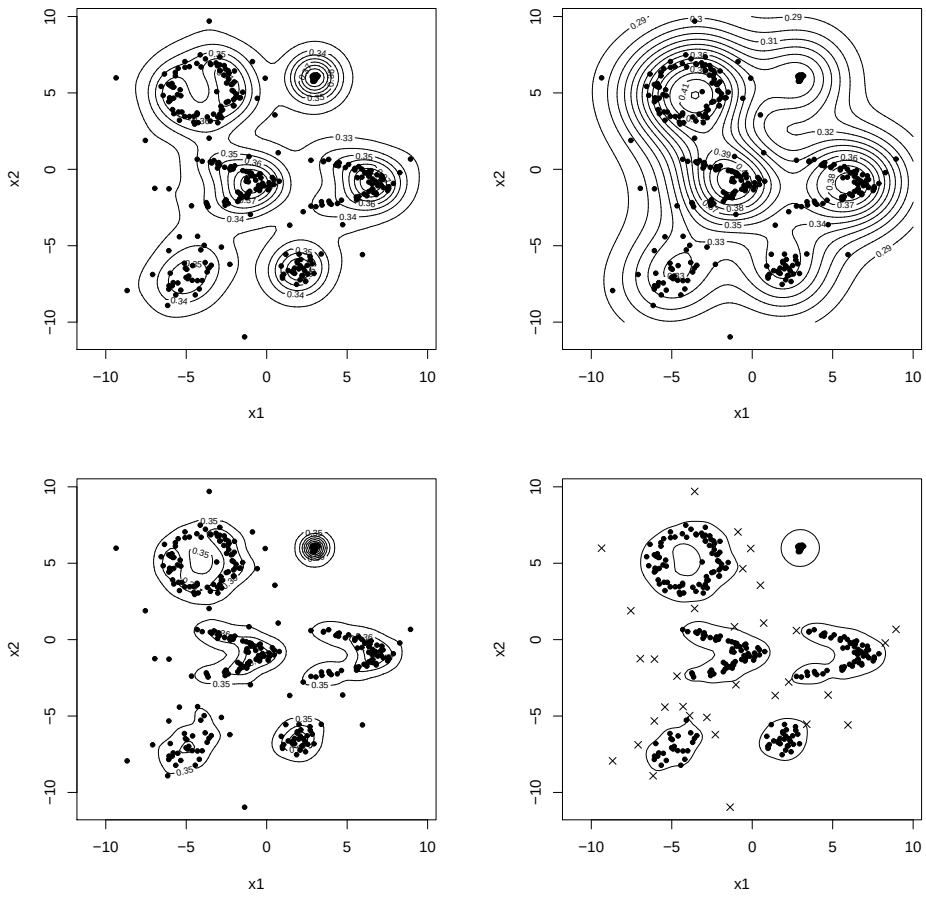

Fig. 1. First row - Contour plot of the memberships for $\sigma = 1$ and $\sigma = 2$. Second row - left - Contour plot of the memberships for $\sigma = 0.5$. right - Cluster boundaries for $\sigma = 0.5$ with a 10% rejection rate.

In all the runs of 1-PCM_Φ the *Core* step, which involves the minimization of $L^\Phi(U, V^\Phi)$ (Eq. 11), resulted to be very fast, since few tenths of iterations of Eq. 16 where enough.

B. Stability Validation for Outlier Detection

We want to compare the stability of the solutions of 1-PCM_Φ, 1-SVM and KDE for outlier detection. In order to do that, we propose a modified version of the method in Ref. [8], where it has been used to estimate the natural number of clusters in a data set. We first report the general ideas underpinning the method, and then we will detail how we intend to modify it to use it in the context of outlier detection.

The general procedure presented in Ref. [8] starts by splitting the original data set in two

disjoint subsets $X_{(1)}$ and $X_{(2)}$. The cardinality of $X_{(1)}$ and $X_{(2)}$ is half the cardinality of the original data set, and data are picked at random to form the two sets. By applying a clustering algorithm on $X_{(1)}$, it is possible to assign the cluster labels to $X_{(2)}$. This mechanism is called *Transfer by Prediction* and can be formalized by a classifier ϕ trained on $X_{(1)}$ that allows to predict the labels of $X_{(2)}$. Here the term classifier denotes the fact that a decision on $X_{(2)}$ can be taken on the basis of the clustering algorithm trained on $X_{(1)}$. On the other hand, it is possible to apply directly the clustering algorithm on $X_{(2)}$ obtaining a set of labels $\mathbf{z}_{(2)}$. The labels $\phi(X_{(2)})$ and $\mathbf{z}_{(2)}$ can then be compared using, for instance, the Hamming distance. Such distance has to take into account the possible permutations of the cluster labels, since the labels $\phi(X_{(2)})$ and $\mathbf{z}_{(2)}$ are not necessarily in a direct correspondence. The expected value of this distance, that in practice is evaluated an the average over several repetitions, can be considered as a stability measure of the clustering solution. This distance requires a normalization dependent from the number of clusters.

Now we present a modified version of that algorithm to deal with outlier detection instead of clustering. Again, we split the data set X in two halves $X_{(1)}$ and $X_{(2)}$ as discussed before. Now we can apply an outlier detection algorithm on $X_{(1)}$ and use this to take a decision on the patterns in $X_{(2)}$; in this way we obtain the labels $\phi(X_{(2)})$. The decision on the data in $X_{(2)}$ is taken by comparing their membership values, as computed through Eq. 28, to the threshold on the memberships of the training patterns (the threshold is selected using their quantiles as explained in Section IV-D). Then, we apply the outlier detection algorithm on $X_{(2)}$ directly, thus obtaining the set of labels $\mathbf{z}_{(2)}$. Note that the labels are of the type $0-1$ meaning "normal" and "outlier" respectively.

To evaluate the stability of an outlier detection algorithm, we propose a matching of the labels $\phi(X_{(2)})$ and $\mathbf{z}_{(2)}$ based on the Jaccard coefficient. For two binary variables ξ and χ, the Jaccard coefficient is a measure of their concordance on positive responses. Given the confusion matrix:

		χ	
		0	1
ξ	0	a_{00}	a_{01}
	1	a_{10}	a_{11}

TABLE II

PSEUDO-CODE OF THE STABILITY VALIDATION PROCEDURE FOR OUTLIER DETECTION

1) Repeat r times:
   a) split the given data set into two halves $X_{(1)}$ and $X_{(2)}$;
   b) apply the outlier detection algorithm on $X_{(1)}$ and predict the labels on $X_{(2)}$ obtaining $\phi(X_{(2)})$;
   c) apply the outlier detection algorithm on $X_{(2)}$ obtaining $\mathbf{z}_{(2)}$;
   d) compute the Jaccard coefficient between $\phi(X_{(2)})$ and $\mathbf{z}_{(2)}$;

The Jaccard coefficient is defined as:

$$J(\xi, \chi) = \frac{a_{11}}{a_{01} + a_{10} + a_{11}} \tag{37}$$

The motivation for the use of the Jaccard coefficient, instead of the simple matching, is that we want to measure the concordance between the solutions $\phi(X_{(2)})$ and $\mathbf{z}_{(2)}$ in the identification of outliers. We want to give more importance to the fact that $\phi(X_{(2)})$ and $\mathbf{z}_{(2)}$ match on the outliers, rather than normal patterns. Also, since we are dealing with two classes (outlier vs non-outliers) we don't need to normalize this score as in the case of clustering [8]. The steps of the stability validation procedure for outlier detection are outlined in Tab. II.

We decided to evaluate the stability for different values of ν in 1-SVM. As we have seen before, ν gives the upper bound on the fraction of outliers that we flag in the data set. For this reason, to compare correctly 1-SVM with 1-PCM_Φ for different values of ν, we decided to set a threshold on the memberships obtained by 1-PCM_Φ and a threshold on the probabilities obtained by KDE, in order to reject exactly the same number of patterns rejected by 1-SVM with that particular value of ν.

C. Results

1) Synthetic data set: The synthetic data set used in our experiments is shown in Fig. 2. It is a two-dimensional data set composed by 400 points. They have been generated using a Gaussian distribution centered in $(0, 0)$ having unit variance along the two axes. Other 20 points have been added sampling uniformly the set $[3, 10] \times [-10, 10]$ and 10 points sampling uniformly the set $[-10, -3] \times [-10, 10]$ thus obtaining a non-symmetric outlier distribution.

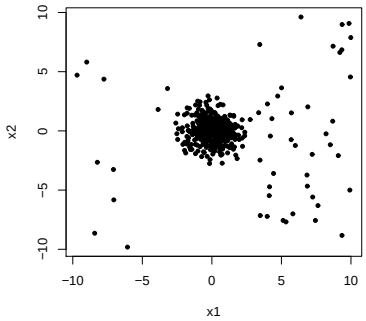

Fig. 2. A two dimensional synthetic data set. Data are generated from a Gaussian and a non-symmetric distribution of outliers.

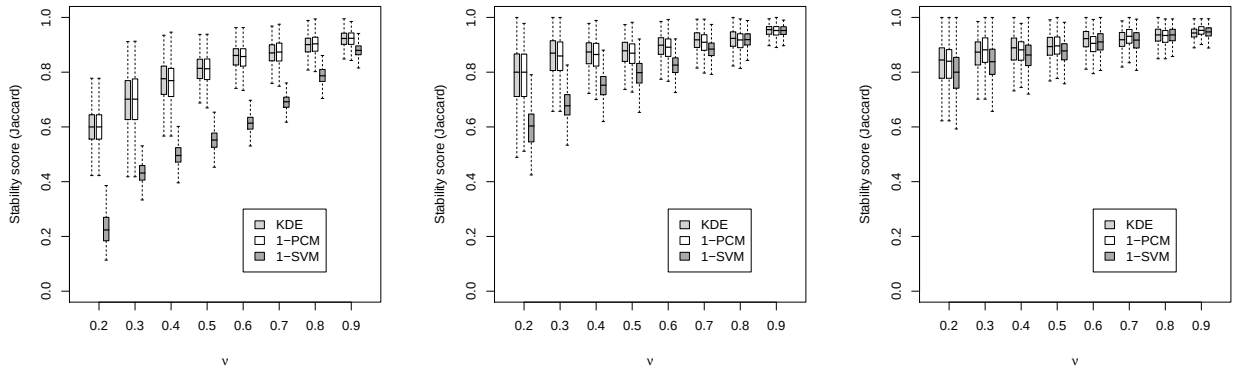

Fig. 3. Synthetic data set - Comparison of 1-SVM, 1-PCM$_\Phi$, and KDE using box-and-whisker plots of the Jaccard coefficient over 500 repetitions. All the methods use a Gaussian kernel; in the three plots the width of the kernel has been set respectively to: $\sigma = 0.5$, $\sigma = 1$, and $\sigma = 5$. The regularization parameter η in 1-PCM$_\Phi$ has been set using Eq. 27 with $\gamma = 1$.

We tested the stability of 1-SVM and 1-PCM$_\Phi$ for outlier detection using the algorithm presented in Tab. II. We used a Gaussian kernel with three different values of σ: 0.5, 1, and 5; the regularization parameter η has been set automatically using Eq. 27, where we set the value of γ to 1. The results are summarized in Fig. 3, where the box-and-whiskers plot of the Jaccard coefficient over 500 repetitions ($r = 500$) for different values of ν. In each plot of Fig. 3, we report a comparison among 1-SVM, 1-PCM$_\Phi$, and KDE.

We can see that the performances of 1-PCM$_\Phi$ and KDE are comparable in terms of stability, as we expect from the analysis on the connection between them. The analogy lies in the the

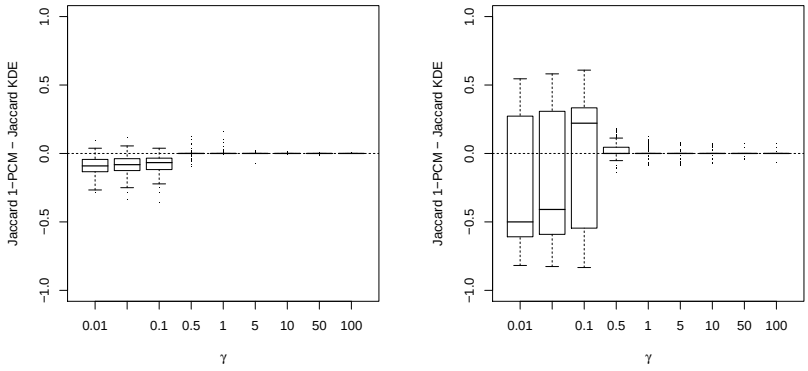

Fig. 4. Synthetic data set - Box-and-whisker plots of the difference between the stability scores for 1-PCM_Φ and KDE with kernel parameter $\sigma = 1$ over 1000 repetitions for different values of γ. The two plots correspond to 1% and 10% rejection rates respectively.

regularization properties parameterized by η that can be computed automatically from the data set. In Eq. 27 we introduced the multiplicative term γ in the computation of η to have a better control on the regularization properties of the algorithm. It is interesting to analyze the behavior of 1-PCM_Φ with respect to KDE for different values of γ. In Fig. 4 we report two box-and-whisker plots of the difference between the stability of 1-PCM_Φ and KDE's (evaluated using the algorithm in Tab. II) over 1000 repetitions. The two plots correspond to 1% and 10% rejection rates respectively. As we can see from the Fig. 4, for high values of γ, the stabilities are comparable, while for very small values of γ 1-PCM_Φ overfits the training data. This is expected from the theoretical analysis, since the regularization term vanishes for small values of η.

2) Real data sets: We compared the stability and the accuracy of 1-PCM_Φ, 1-SVM, and KDE for outlier detection on three real data sets taken from the UCI repository [23]: Breast, Ionosphere, and Iris. The accuracy has been evaluated by considering some of the classes as normal, and the remaining ones as containing the outliers[3]. We considered 500 repetitions where we trained the outlier detection algorithm on a subsets of size n of the normal class. When comparing the stability and the accuracy, we fixed ν in 1-SVM that resulted in a fraction of outliers. As in the synthetic case, in 1-PCM_Φ and KDE we chose to reject the same fraction of outliers as in 1-SVM. The multiplicative term γ in the computation of η in Eq. 27 for 1-PCM_Φ

[3]A similar experimental setting has been proposed in [17], [18].

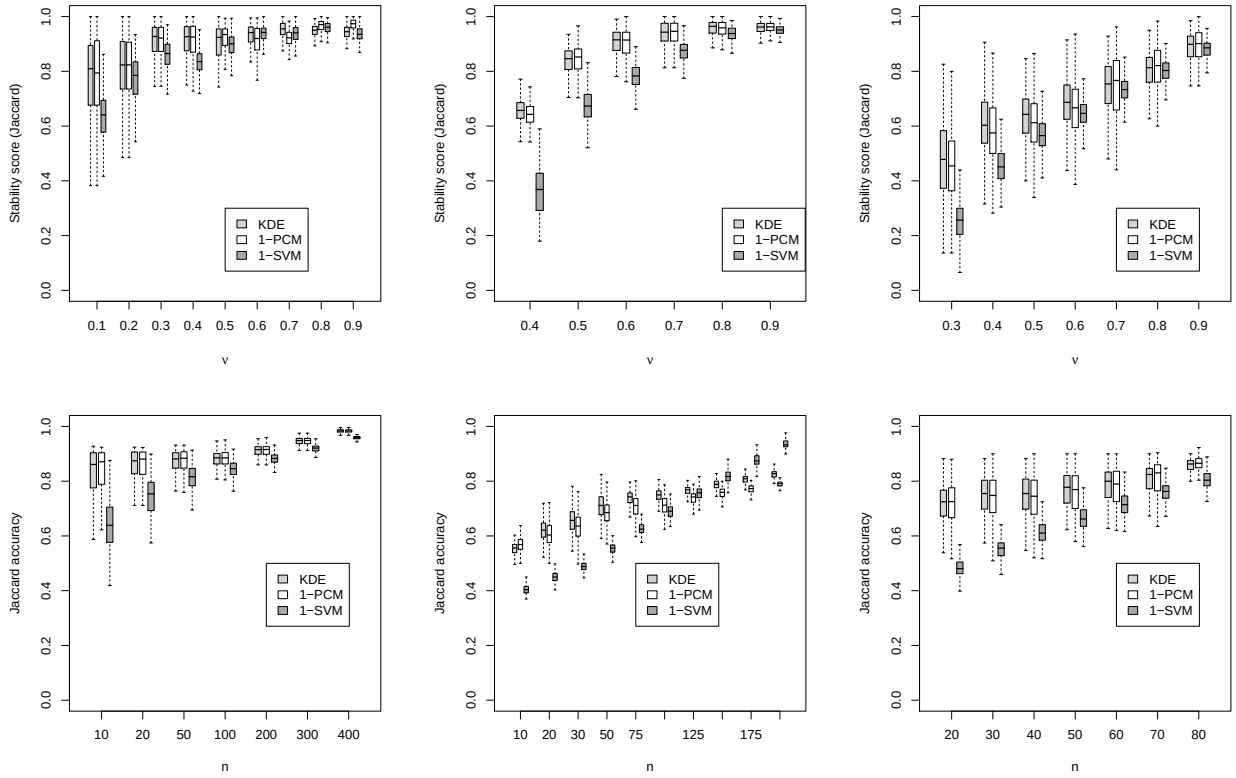

Fig. 5. Stability and accuracy of 1-SVM, 1-PCM$_\Phi$, and KDE; all the methods use a Gaussian kernel. We report the results on Breast, Ionosphere, and Iris in the three columns respectively. The value of the kernel parameter is: Breast $\sigma = 10$, Ionosphere $\sigma = 1$, Iris $\sigma = 0.5$. In all the data sets, the regularization parameter η in 1-PCM$_\Phi$ has been set using Eq. 27 with $\gamma = 1$. The stability is evaluated over 500 repetitions using the method of Tab. II and is shown in the first row. The second row shows the accuracy (evaluated as the Jaccard coefficient between predicted and actual labels) of the three methods over 500 repetitions ($\nu = 0.1$ for Breast and $\nu = 0.2$ for Ionosphere and Iris).

has been set to one in all the experiments.

The study of the stability follows the same steps as in the synthetic data set. The study of the performances has been done in terms of accuracy in identifying outliers for the three algorithms. In this case, we show a comparison of accuracy with respect to the size of the data set. In particular, we train the outlier detection algorithms on a subset $X_{(1)}$ of the entire data set, and we predict the labels on $X_{(2)}$ using the decision function learned on $X_{(1)}$, thus obtaining the labels $\phi(X_{(2)})$. Let $\mathbf{t}_{(2)}$ be the vector of true labels of $X_{(2)}$; we evaluate the accuracy computing the Jaccard coefficient between $\phi(X_{(2)})$ and $\mathbf{t}_{(2)}$:

$$\text{accuracy} = J(\phi(X_{(2)}), \mathbf{t}_{(2)})$$

For each size value of $X_{(1)}$, we resampled 500 times. The results are shown in the bottom row of Fig. 5 for different values of n (the size of the training set $X_{(1)}$).

The Breast Cancer Wisconsin (Original) Data Set was obtained by the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg [24]. The data set is composed by 699 nine-dimensional patterns, labeled as benign or malignant. Since there are some missing values, we decided to remove the corresponding patterns, obtaining 683 patterns. The class distribution is 65% for the benign class and 35% for the malignant class. In the comparison of stability and accuracy, we used a Gaussian kernel with $\sigma = 10$. The stability of the solutions is shown in the top panel of the first column of Fig. 5. The accuracy has been evaluated by considering the benign class as normal and the malignant class as the one containing the outliers. The plot of the accuracy corresponds to $\nu = 0.1$.

Ionosphere is a collection of radar data, collected by a phased array of 16 high-frequency antennas in Goose Bay, Labrador having the free electrons in the ionosphere as target [25]. The class labels are two: "Good" radar returns are those showing evidence of some type of structure in the ionosphere, while "Bad" returns are those that do not; their signals pass through the ionosphere. Received signals were processed using an appropriate autocorrelation function. The system used 17 pulse numbers and the patterns in the data set are described by two features per pulse number. In the comparison of stability and accuracy, we used a Gaussian kernel with $\sigma = 1$. The stability of the solutions is shown in the top panel of the central column of Fig. 5. The accuracy has been evaluated by considering the class "Good" as normal and the class "Bad" as the one containing the outliers. The plot of the accuracy corresponds to $\nu = 0.2$.

The Iris data set is one of the most popular data sets studied by the Machine Learning community [1], [26]. It contains three classes of 50 patterns each; each class refers to a type of iris plant. The class "setosa" is linearly separable from the other two ("versicolor" and "virginica") that are overlapped. The features are four: sepal length, sepal width, petal length, and petal width. In the comparison of stability and accuracy, we used a Gaussian kernel with $\sigma = 0.5$. The stability of the solutions is shown in the top panel of the right column of Fig. 5. The accuracy has been evaluated by considering the classes "setosa" and "versicolor" as normal and the class "virginica" as the one containing the outliers. The plot of the accuracy corresponds $\nu = 0.2$.

As we can see from these results, the proposed method achieves good performances both in

terms of accuracy and in terms of stability of the solutions, compared to 1-SVM. This effect can be seen especially for small values of n and for small rejection rates. This can be particularly useful in some applications where the cardinality of the data set might be small. Stability and accuracy of 1-PCM$_\Phi$ are comparable to those of KDE.

VI. Conclusion

In this paper, we introduced the possibilistic clustering in kernel-induced spaces, and we analyzed some of its theoretical properties. In particular, we highlighted the connections of the 1-PCM$_\Phi$ with 1-SVM and KDE. This suggests that 1-PCM$_\Phi$ can be used to model densities in a non-parametric way, perform clustering, and to detect outliers. In the comparison with KDE, we focused on kernel that are function of the difference between patterns. We showed that in this case, the limit for a large value of the regularization parameter yields an interpretation of 1-PCM$_\Phi$ in terms of a KDE solution. In the comparison with 1-SVM, we noticed the similarity between the optimization problems. The 1-PCM$_\Phi$ objective function, however, contains an additional term that can be interpreted as a regularizer, and is an entropy based score computed on the memberships. Also, we noticed the dual role of the memberships in 1-PCM$_\Phi$ with respect to the Lagrange multipliers in 1-SVM. These differences give to the proposed algorithm the ability to avoid overfitting and to enhance the stability of the found solutions.

All these considerations are fully confirmed by the tests conducted on synthetic and real data sets on the stability and the accuracy in outlier detection problems. Especially for small values of ν, that correspond to the rejection of few outliers, the stability of 1-PCM$_\Phi$ is on average higher than 1-SVM's. In 1-PCM$_\Phi$, the selection of the regularization parameter is not critical, and the stability is achieved for η in a wide range of values. Moreover, the optimization procedure is iterative and very fast, since few iterations are needed.

The performances in terms of accuracy and stability of 1-PCM$_\Phi$ and KDE resulted to be comparable. We discussed, however, that the applicability of 1-PCM$_\Phi$ is more general than KDE. 1-PCM$_\Phi$ can be employed with any positive semidefinite kernel, and in any application where pairwise dissimilarities are available among data points.

It is important to remark the weak points of 1-PCM$_\Phi$ as well. The main drawback is related to the complexity in the testing stage. The representation of the data set in 1-SVM is sparse, thanks to the description of the data in terms of the support vectors only. In many cases, the

reduction given by this compact description leads to a remarkable computational advantage when testing new patterns. In the proposed algorithm, instead, we need to use all the patterns, and hence the full kernel matrix, to compute the membership of a new test pattern. Sparsification schemes could reduce the computational complexity in the testing stage.

APPENDIX I

A. Optimization Algorithm - The Unconstrained Case

Let's analyze the procedure to optimize the objective function:

$$L = \sum_h u_h \|\mathbf{x}_h - \mathbf{v}\|^2 + \eta \sum_h (u_h \ln(u_h) - u_h) \tag{38}$$

The optimization technique that we use is the so called Picard iterations technique. L depends on $\mathbf{u}$ and $\mathbf{v}$ that are related to each other, namely $\mathbf{u} = \mathbf{u}(\mathbf{v})$ and $\mathbf{v} = \mathbf{v}(\mathbf{u})$. In each iteration one of the two groups of variables is kept fixed, and the minimization is performed with respect to the other. The update equation can be obtained by setting the derivatives of L to zero:

$$\frac{\partial L}{\partial \mathbf{v}} = 0, \qquad \frac{\partial L}{\partial u_h} = 0 \tag{39}$$

These equations lead to the following:

$$u_h = \exp\left(-\frac{\|\mathbf{x}_h - \mathbf{v}\|^2}{\eta}\right) \tag{40}$$

$$\mathbf{v} = \frac{\sum_{h=1}^n u_h \mathbf{x}_h}{\sum_{h=1}^n u_h} \tag{41}$$

The constraint $0 \leq u_k \leq 1$ is satisfied, since the form assumed by the update equations.

B. Optimization Algorithm - The Constrained Case

We show now that constraining the sum of the memberships does not affect the behavior of the optimization procedure. In other words, the results of the constrained and unconstrained case differ only in the scaling factor of the memberships. Let's start with the objective function:

$$L = \sum_h u_h \|\mathbf{x}_h - \mathbf{v}\|^2 + \eta \sum_h (u_h \ln(u_h) - u_h) \tag{42}$$

subject to:

$$\sum_h u_h = 1 \tag{43}$$

Following the Lagrange multipliers technique, the optimization of L with the constraint on the memberships requires the optimization of the Lagrangian:

$$L' = \sum_h u_h \|\mathbf{x}_h - \mathbf{v}\|^2 + \eta \sum_h (u_h \ln(u_h) - u_h) + \gamma \left(\sum_h u_h - 1 \right) \tag{44}$$

that is a combination of L and the constraint equation weighted by the Lagrange multiplier γ. Setting the derivatives of L with respect to u_h to zero:

$$\frac{\partial L'}{\partial u_h} = \|\mathbf{x}_h - \mathbf{v}\|^2 + \eta \ln(u_h) + \gamma = 0 \tag{45}$$

we get:

$$u_h = \exp\left(-\frac{\|\mathbf{x}_h - \mathbf{v}\|^2}{\eta} \right) \exp\left(-\frac{\gamma}{\eta} \right) \tag{46}$$

Substituting u_h into the constraint equation, we obtain:

$$\sum_h \exp\left(-\frac{\|\mathbf{x}_h - \mathbf{v}\|^2}{\eta} \right) \exp\left(-\frac{\gamma}{\eta} \right) = 1 \tag{47}$$

that gives:

$$\gamma = \eta \ln\left(\sum_h \exp\left(-\frac{\|\mathbf{x}_h - \mathbf{v}\|^2}{\eta} \right) \right) \tag{48}$$

Finally, substituting Eq. 48 into Eq. 46:

$$u_h = \frac{\exp\left(-\frac{\|\mathbf{x}_h - \mathbf{v}\|^2}{\eta} \right)}{\sum_r \exp\left(-\frac{\|\mathbf{x}_r - \mathbf{v}\|^2}{\eta} \right)} \tag{49}$$

From this result, it is clear that the update of $\mathbf{v}$ is the same as in the unconstrained case, since the normalization in Eq. 49 cancels out in the computation of $\mathbf{v}$. This means that starting from the same memberships, the constrained and unconstrained cases give the same $\mathbf{v}$, and the memberships are only scaled to sum up to one.

REFERENCES

[1] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Wiley, 1973.

[2] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 1981.

[3] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 2, pp. 98–110, 1993. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=227387

[4] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta, "A survey of kernel and spectral methods for clustering," *Pattern Recognition*, vol. 41, no. 1, pp. 176–190, January 2008.

[5] M. Filippone, F. Masulli, and S. Rovetta, "Possibilistic clustering in feature space," in *WILF*, ser. Lecture Notes in Computer Science. Springer, 2007.

[6] R. Krishnapuram and J. M. Keller, "The possibilistic c-means algorithm: insights and recommendations," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 3, pp. 385–393, 1996. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=531779

[7] D. Q. Zhang and S. C. Chen, "Kernel based fuzzy and possibilistic c-means clustering," in *Proceedings of the International Conference Artificial Neural Network*. Turkey, 2003, pp. 122–125.

[8] T. Lange, V. Roth, M. L. Braun, and J. M. Buhmann, "Stability-based validation of clustering solutions," *Neural Computation*, vol. 16, no. 6, pp. 1299–1323, 2004.

[9] M. Filippone, "Dealing with non-metric dissimilarities in fuzzy central clustering algorithms," *International Journal of Approximate Reasoning*, vol. 50, no. 2, pp. 363–384, February 2009. [Online]. Available: http://dx.doi.org/10.1016/j.ijar.2008.08.006

[10] F. Höppner and F. Klawonn, "A contribution to convergence theory of fuzzy c-means and derivatives," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 5, pp. 682–694, 2003. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1235994

[11] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, no. 3, pp. 337–404, 1950.

[12] M. Aizerman, E. Braverman, and L. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning." *Automation and Remote Control*, vol. 25, pp. 821–837, 1964.

[13] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[14] D. Q. Zhang and S. C. Chen, "Fuzzy clustering using kernel method," in *The 2002 International Conference on Control and Automation, 2002. ICCA*, 2002, pp. 162–163. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1229535

[15] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. R. Müller, G. Rätsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1000–1017, 1999.

[16] M. Barni, V. Cappellini, and A. Mecocci, "Comments on a possibilistic approach to clustering," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 3, pp. 393–396, 1996.

[17] D. M. J. Tax and R. P. W. Duin, "Support vector domain description," *Pattern Recognition Letters*, vol. 20, no. 11-13, pp. 1191–1199, 1999.

[18] B. Schölkopf, J. C. Platt, J. S. Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[19] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines (version 2.31)." [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.20.9020

[20] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, April 1986. [Online]. Available: http://www.worldcat.org/isbn/0412246201

[21] A. B. Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering." *Journal of Machine Learning Research*, vol. 2, pp. 125–137, 2001.

[22] P. Hall, G. S. Watson, and J. Cabrera, "Kernel density estimation with spherical data," *Biometrika*, vol. 74, no. 4, pp. 751–762, 1987.

[23] A. Asuncion and D. J. Newman, “UCI machine learning repository,” 2007. [Online]. Available: http://www.ics.uci.edu/simmlearn/MLRepository.html

[24] W. H. Wolberg and O. L. Mangasarian, “Multisurface method of pattern separation for medical diagnosis applied to breast cytology,” *Proceedings of the National Academy of Sciences,U.S.A.*, vol. 87, pp. 9193–9196, 1990.

[25] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker, “Classification of radar returns from the ionosphere using neural networks,” *Johns Hopkins APL Technical Digest*, vol. 10, pp. 262–266, 1989.

[26] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annals Eugenics*, vol. 7, pp. 179–188, 1936.

Maurizio Filippone Maurizio Filippone received a Master’s degree in Physics in 2004 and a PhD in Computer Science in 2008, from the University of Genova. In 2007, he has been Research Scholar at the Information and Software Engineering Department, at George Mason University. In 2008, he joined the Machine Learning group at the University of Sheffield as a Research Associate. His research interests are focused on Machine Learning techniques and their applications to pattern recognition, bioinformatics, and time series analysis and forecasting.

Francesco Masulli Francesco Masulli received the Laurea degree in Physics in 1976 from the University of Genova. He is currently an Associate Professor in Computer Science at the University of Genova. From 1983 to 2001 he has been an Assistant Professor at the University of Genova and from 2001 to 2005 an Associate Professor at University of Pisa. He authored or co-autored more than 120 scientific papers in Machine Learning, Neural Networks, Fuzzy Systems, Pattern Recognition and Bioinformatics.

Stefano Rovetta Stefano Rovetta received a Laurea degree in Electronic Engineering in 1992 and a PhD degree in Models, Methods and Tools for Electronic and Electromagnetic Systems in 1996, both from the University of Genova. He is currently an Assistant Professor in Computer Science at the University of Genova. His current research interests include Machine Learning and Clustering techniques for high-dimensional biomedical data analysis and document analysis.