Convex Hull Data Description

Chengqiang Huang*, Yulei Wu*, Geyong Min*, Yiming Ying†
*University of Exeter
†State University of New York at Albany
*{ch544,Y.L.Wu,G.Min}@exeter.ac.uk †yying@albany.edu

***Abstract*—Data description is a key topic under the broad umbrella of data mining and finds application in various domains. In this paper, a novel method called Convex Hull Data Description (CHDD) is proposed to expand the family of data description. Essentially, CHDD is rooted in convex hull approximation which finds the approximated extreme points of the convex hull through Semi-Nonnegative Matrix Factorization (Semi-NMF). With the utilization of the kernel trick, CHDD achieves data description that is readily applicable to general one-class classification and clustering problems. Empirical experiments show that CHDD successfully describes the convex hull with the approximated extreme points and achieves competitive results in one-class classification and clustering.**

I. Introduction

"Data Description" is a general term that typically refers to the description of the characteristics of a dataset. All the related tasks, e.g., data distribution/density analysis and clustering, are generally covered by the term. The data description studied in this paper, however, mainly concerns the problem of one-class classification, i.e., identifying the best contour of a given dataset, which finds extensive applications in novelty/anomaly detection. As a concrete example, a famous method for data description is Support Vector Data Description (SVDD) proposed by Tax and Duin in 1999 [16]. In Fig. 1, it shows the performance of SVDD in describing 25 data instances (blue cross). The contour of the dataset, i.e., the red boundaries, acts as a practical discriminator in distinguishing anomalies from normal data instances. This kind of data description has been broadly applied in intrusion detection [8], time series anomaly detection [12], and many other different fields related to the problem of one-class classification.

Different from SVDD, this paper considers a well-known geometrical concept, i.e., the convex hull, and proposes Convex Hull Data Description (CHDD) to supply additional beneficial capabilities in data description. To be more specific, CHDD approximates the convex hull of a dataset by recognizing the data representatives. The description of the original dataset using the representatives not only results in the criteria for one-class classification, but also reveals the internal relations among data instances which enable data clustering. Therefore, CHDD solves the tasks of one-class classification and clustering at the same time. In addition to the aforementioned capability, the utilization of the approximated convex hull in CHDD sheds light on the intrinsic reasons for the decision-making in related tasks and facilitates incremental data description.

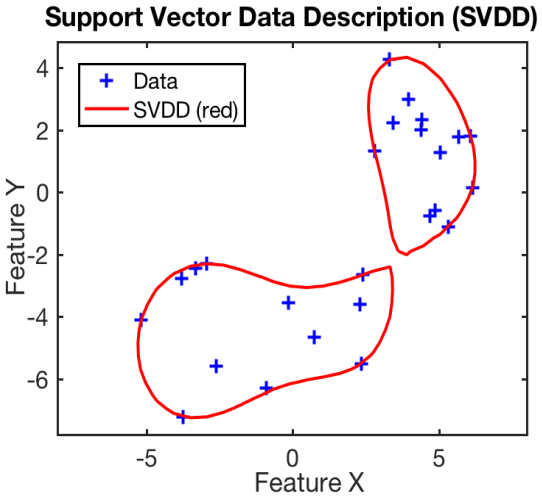

Fig. 1. An exemplary data description using SVDD

To summarize, this paper makes the following contributions:

- A novel method, i.e., Convex Hull Data Description (CHDD), is developed for data description. The method is applicable in both one-class classification and clustering tasks. The underlying mathematical formulation CHDD solved, i.e., convex hull approximation, is itself a hot research problem.
- The Semi-Nonnegative Matrix Factorization (Semi-NMF) is proved to be a practical solver for convex hull approximation. It enables the utilization of the kernel trick that generates competitive results in one-class classification and clustering.
- The applicability of CHDD is illustrated in one-class classification and clustering of a variety of datasets with distinct features. The further advantages of CHDD are also discussed.

The rest of this paper is organized as follows. Related work and background information are presented in Section II. In Section III, the formulation of CHDD as well as the ways to achieve kernelization for the purpose of general one-class classification and data clustering are elaborated. The prove of the applicability of Semi-NMF in convex hull approximation is given in the same section. The details of the experiments and discussions are provided in Section IV and conclusions are drawn in the last section.

II. Related Work

A. Data Description

A significant related work of data description is the comprehensive toolbox provided by Tax [17] for the purpose of one-class classification. In the toolbox, numerous kinds of methods were implemented. Basically, the methods can be categorized into five different classes: 1) **statistical methods** that analyze the statistical properties of the dataset, e.g., data distributions. Gaussian Mixture Model (GMM) is a famous method of this kind; 2) **distance-based methods** that regard distance as the most relevant factor in measuring data similarity to enable one-class classification or clustering. K-Nearest Neighbor (KNN) is an good representative; 3) **density-based methods** which leverage the density information of normal/abnormal data in a specific region of the data space to give the probability of whether a data instance is anomalous. An example of this kind in the toolbox is Parzen density estimation; 4) **model-based methods**, typical examples of which include Self-organization Map (SOM) and Support Vector Data Description (SVDD). Both of these methods use a model to capture the properties of the dataset; and 5) **spectral analysis methods** that investigate the attributes of the space on which the data lie, e.g., Principle Component Analysis (PCA).

In addition to the methods that have been extensively studied and practiced, some up-to-date methods, such as the binary decision diagram-based one-class classifier (BDD) [7], one-class classification with Gaussian Process (one-class GP) [6] and Isolation Forest (iForest) [11], also exhibit their novelty and good performances in data description. Amongst these three methods, one-class GP heavily relies on the underlying assumption of the correctness of the Gaussian Process which is a concept that is relatively difficult to understand and cannot be interpreted geometrically. On the other hand, BDD and iForest both adopt intuitive geometrical explanations for data description. BDD uses hypercubes to define the regions that normal data lie in. As a result, the determination of whether a data instance is normal is converted to checking whether the data instance lies in a normal hypercube. iForest describes a given dataset with several tree models, each of which tries to isolate any data instance in the dataset from all the others through iteratively dividing the data space. All these methods achieve one-class classification. However, unlike other methods, e.g., KNN and GMM, they cannot be immediately applicable in clustering problems because they do not explicitly measure the relationship among data instances.

B. Convex Hull Analysis

Identifying the convex hull of a multivariate dataset has been a research topic for a long time. Over the years, geometricians and many other researchers have developed numerous related methods. The early idea of computing an approximated convex hull was proposed in 1982, when Bentley and Faust [2] introduced a set of algorithms for the problem. Ever since, convex hull identification and approximation have both experienced speedy development due to their broad applications in data mining and machine learning [1] [19].

Besides the conventional methods, most of which find the convex hull from a geometric perspective, a recent work [4] introduced the utilization of spectral analysis for convex hull identification. The method leverages the definition of the convex hull and solves an optimization problem to obtain a sparse coefficient matrix which encodes the identities of the extreme points. A related formulation is also adopted in [14], where the authors proposed a greedy search approach for selecting the extreme points according to some predefined constraints derived from the formulation. It is worth noting that the convex hulls of general datasets are normally not tight enough for data description. As a result, although both of these methods are elegant in convex hull identification, they are not practically applicable in data description tasks.

C. Originality of CHDD

Compared to the above-mentioned methods, the most notable features of CHDD from the perspective of data description lie in two aspects. 1) CHDD origins from the fact that the data instances inside a convex hull can be described by a non-negative linear combination of the extreme points. Consequently, the classification or clustering decisions made for a data instance can be better understood by examining the extreme points that have high weights in describing the data instance. 2) CHDD is among the methods that are applicable in both one-class classification and clustering due to the reason that it measures the relations among all the data instances. One-class classification is achieved through thresholding the relations, while a detailed mining of the relations reveals the densely connected groups of the data instances.

Additionally, CHDD adopts spectral analysis for convex hull approximation. Different from the related work, it considers data description from the perspective of matrix decomposition, which enables the utilization of the kernel trick. For that reason, the data description can be performed in the kernel space instead of the original data space, which makes possible the better performance of CHDD in one-class classification and clustering over general datasets. The formulated matrix decomposition motivates the usage of the multiplicative updating algorithm that is originally exploited in Semi-Nonnegative Matrix Factorization (Semi-NMF) [3]. CHDD, therefore, explores the applications of Semi-NMF in one-class classification and clustering from a different perspective.

III. Convex Hull Data Description

A. Problem Formulation

Let $S = \{x_1, x_2, \cdots, x_N\}$ be a dataset consisting of N D-dimensional data instances $x_i \in \mathbb{R}^D$, $i \in \{1, 2, \cdots, N\}$. From the perspective of simple geometry, there must be some data instances in S that can be used as the representatives, i.e., extreme points, to describe any data instance in the dataset. According to the concept of the convex hull, the process is expressed as:

$$\begin{aligned} & x_i = X_{ep} c_i, \\ \forall i, \quad & c_i \geq 0, \quad \mathbf{1}^T c_i = 1, \end{aligned} \tag{1}$$

where $X_{ep} \triangleq [x_{ep,1} \cdots x_{ep,n}]$ is a matrix that comprises n column vectors representing the extreme points, and $c_i \in \mathbb{R}^n$ denotes the coefficient vector that is employed in reconstructing x_i. Note that $\mathbf{1}$ in this context is a n-dimensional column vector whose elements are all 1. Essentially, the formulation tries to describe x_i using a weighted linear combination of the extreme points. A more completed formulation can be given with $X \triangleq [x_1 \cdots x_N]$ and $C \triangleq [c_1 \cdots c_N]$:

$$\begin{aligned} X &= X_{ep}C, \\ C \geq 0, \quad & \mathbf{1}^T C = \mathbf{1}^T. \end{aligned} \tag{2}$$

In the task of convex hull identification or approximation, the essential problem is to determine the extreme points X_{ep}. Although there are numerous solutions that consider the problem from the geometric point of view, we argue that it is also beneficial to start from the mathematical formulation. A key observation of the problem is that the coefficient matrix C encodes valuable information for the identification of the extreme points while considering the following formulation:

$$\begin{aligned} X &= XC, \\ C \geq 0, \quad & \mathbf{1}^T C = \mathbf{1}^T. \end{aligned} \tag{3}$$

For an extreme point x_j, its corresponding coefficient vector has to be of the form: $c_j = (0, \ldots, 1, \ldots, 0)^T$. The index of the position that 1 appears has to be j. In other words, the diagonal elements of C indicate whether their corresponding data instances are extreme points. As a concrete example, let us consider the situation where the coefficient matrix is calculated for 4 data instances according to Eq. (3):

$$\begin{bmatrix} 0 & 2 & 1 & 1 \\ 0 & 0 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 2 & 1 & 1 \\ 0 & 0 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0.25 \\ 0 & 1 & 0 & 0.25 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4}$$

As the coefficient matrix indicates, the first three data instances, i.e., $(0,0)^T$, $(2,0)^T$, and $(1,2)^T$ are extreme points, because it is not possible to find other linear combinations to reconstruct the data without using themselves. Nevertheless, $(1,1)^T$ can be described by the other data, which reflects that it is not an extreme point. In the next subsection, an optimization method is developed to find an approximated coefficient matrix for the identification of extreme points in general datasets, through which we develop Convex Hull Data Description.

B. Convex Hull Approximation

Based on the problem formulation, the essentials of CHDD rest on the convex hull approximation through solving an optimization problem:

$$\begin{aligned} \min_{C} \quad & \|X - XC\|^2, \\ \text{s.t.} \quad & C \geq 0, \quad \mathbf{1}^T C = \mathbf{1}^T. \end{aligned} \tag{5}$$

This formulation is derived from Eq. (3). It relaxes the constraint of the equality, i.e., $X = XC$, and tries to describe the original dataset using itself in a best-effort way, i.e., minimizing the differences between X and XC.

To solve the formulation and enable wider classes of applicable algorithms, the equality constraint in Eq. (5) is emerged into the target function through appending a constant weight to each data instances. In other words, a data instance x_i is actually constructed as $x_i = [x_i^T \ \omega]^T$, where ω is the weight. It implicitly appends $\min_C \omega^2 \cdot \|\mathbf{1}^T C - \mathbf{1}^T\|^2$ to the original target function and realizes the requirement of $\mathbf{1}^T C = \mathbf{1}^T$ with certain relaxation. Consequently, the actual optimization problem is simpler:

$$\min_{C \geq 0} \quad \|X - XC\|^2. \tag{6}$$

A straightforward way to solve this optimization problem is to decompose it into a sequence of Non-Negative Least Square problems (NNLS), which are Quadratic Programming (QP) problems that can be solved directly by existing QP solvers. However, this method is found to be very slow because the number of the QP problems is positively proportional to that of the data instances, which is typically a large number. Another algorithm that was reported to be efficient is Rank-one Residual Iteration (RRI) [5]. Nevertheless, the result of RRI does not necessarily lead to the identification of the extreme points (will be shown in Section IV). As will be theoretically proved later, a good way to solve this problem is to regard it as a Semi-NMF problem [3] and adopt the corresponding slover, i.e., the multiplicative updating rule:

$$C^{k+1} = C^k \circ \sqrt{\frac{[X^T X]_+ + [X^T X]_- C^k}{[X^T X]_- + [X^T X]_+ C^k}}, \tag{7}$$

where C^k denotes the matrix C after the kth iteration of the updating. The notation $A \circ B$ and $\frac{A}{B}$ represent the element-wise multiplication and division between matrixes A and B respectively. The operations $[\cdot]_+$ and $[\cdot]_-$ are defined as:

$$[A]_+ = \frac{A + |A|}{2}, \quad [A]_- = \frac{A - |A|}{2}, \tag{8}$$

where $|A|$ is a matrix consisting of all the absolut values of the elements in matrix A.

Theorem 1: The multiplicative updating rule in Eq. (7) solves Eq. (6) with the guarantee that the solution c_j of an extreme point x_j is of the form $c_j = (0, \ldots, 1, \ldots, 0)^T$, where the 1 is the jth element.

Proof: Without loss of generality, consider only the solution c_j of $x_j = Xc_j$ and $X \geq 0$. The updating rule is simplified as $c_{ij}^{k+1} = c_{ij}^k \cdot \sqrt{\frac{x_i^T x_j}{x_i^T X c_j^k}}$, where c_{ij} denotes the ith element in the vector c_j. Semi-NMF is guaranteed to converge and when it converges [3], i.e., $c_{ij}^{k+1} = c_{ij}^k$, it must suffice that $c_j \geq 0$, $\sum_i c_{ij} = 1$ and:

$$\begin{aligned} x_j &= Xc_j \\ x_j(1 - c_{jj}) &= \sum_{i \neq j} x_i c_{ij}. \end{aligned} \tag{9}$$

If x_j is an extreme point, it cannot be expressed by a linear combination of other data instances. Thus, $c_{jj} = 1$ and $c_{ij} = 0$ for $i \neq j$. This proves the theorem. □

C. Convex Hull Approximation with Gaussian Kernel

When utilizing the algorithm of Semi-NMF for convex hull approximation, the updating rule only depends on the inner product of the original data matrix, i.e., X^TX. Therefore, the kernel trick can be employed to generalize the algorithm for the kernel convex hull approximation. In this paper, we focus on the use of Gaussian kernel $K(X,X) = \phi(X)^T\phi(X)$, whose elements are defined as:

$$\forall i,j, \quad K(x_i,x_j) = \phi(x_i)^T\phi(x_j) = \exp^{\frac{-\|x_i-x_j\|^2}{\sigma^2}}, \tag{10}$$

where $\phi(\cdot)$ denotes the projection function. Hence, the optimization problem in Eq. (6) is changed as:

$$\min_{C\geq 0} \quad \|\phi(X) - \phi(X)C\|^2. \tag{11}$$

Due to the reasons that $K(x,x) = 1$ and $K(x,y) \geq 0$, the updating rule of Semi-NMF algorithm can be modified accordingly:

$$C^{k+1} = C^k \circ \sqrt{\frac{K(X,X)}{K(X,X)C^k}}. \tag{12}$$

Note that this Semi-NMF updating rule under the utilization of Gaussian kernel has roughly the same form as that in NMF [9]. With a further analysis of the algorithm, it is noted that for each element c_{ij} in C, the updating rule works as:

$$c_{ij}^{k+1} = c_{ij}^k \cdot \sqrt{\frac{\phi(x_i)^T\phi(x_j)}{\phi(x_i)^T\phi(X)c_j^k}}. \tag{13}$$

Theorem 2: The multiplicative updating rule in Eq. (13) solves Eq. (11) with the guarantees that the solution c_j of an extreme point $\phi(x_j)$ is of the form $c_j = (0,\ldots,1,\ldots,0)^T$, where the 1 is the jth element, and the solution of an non-extreme point is of a different form, i.e., $c_{jj} \neq 1$.

Proof: It is straightforward from Theorem 1 that, for an extreme point $\phi(x_j)$, $c_{jj} = 1$ and $c_{ij} = 0$ for $i \neq j$. Therefore, the first part of Theorem 2 is proved. For a normal point $\phi(x_j)$ and its estimation $\phi(X)c_j$, consider two known extreme points $\phi(x_l)$ and $\phi(x_r)$ (see Fig. 2), it holds in Gaussian kernel space that if $\frac{\phi(x_l)\phi(x_j)}{\phi(x_l)\phi(X)c_j} < 1$:

$$\begin{aligned} \|\phi(x_l)\|\|\phi(x_j)\|\cos\theta_{lj} &< \|\phi(x_l)\|\|\phi(X)c_j\|\cos\theta_{l\hat{j}} \\ \cos\theta_{lj} &< \cos\theta_{l\hat{j}} \\ \cos\theta_{jr} = \cos\theta - \theta_{lj} &> \cos\theta - \theta_{l\hat{j}} = \cos\theta_{\hat{j}r} \\ \|\phi(x_r)\|\|\phi(x_j)\|\cos\theta_{jr} &> \|\phi(x_r)\|\|\phi(X)c_j\|\cos\theta_{\hat{j}r}, \end{aligned} \tag{14}$$

then it suffices that $\frac{\phi(x_r)\phi(x_j)}{\phi(x_r)\phi(X)c_j} > 1$. The properties, i.e., $\|\phi(x)\| = 1$ and $\|\phi(X)c_j\| < 1$, of the Gaussian kernel are used. And $\theta = \theta_{lj} + \theta_{jr} = \theta_{l\hat{j}} + \theta_{\hat{j}r}$, where $\theta_{lj}, \theta_{jr}, \theta_{l\hat{j}}, \theta_{\hat{j}r}$ and θ indicate the acute angles between the vectors in the pairs $\big(\phi(x_l),\phi(x_j)\big)$, $\big(\phi(x_j),\phi(x_r)\big)$, $\big(\phi(x_l),\phi(X)c_j\big)$, $\big(\phi(Xc_j),\phi(x_r)\big)$ and $\big(\phi(x_l),\phi(x_r)\big)$, respectively. Therefore, there is at least one weight c_{lj} or c_{rj} that is not 0 when Semi-NMF converges. In other words, $\sum_{i\neq j} c_{ij} \neq 0$ and $c_{jj} \neq 1$. This proves the theorem. □

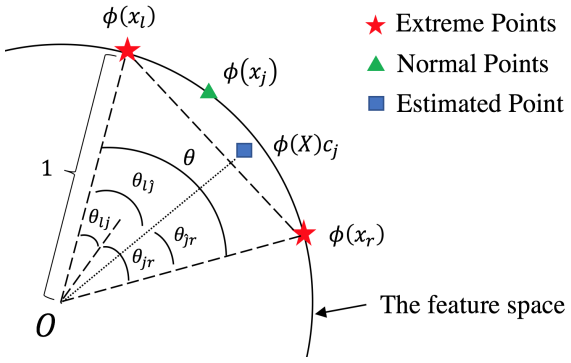

Fig. 2. Gaussian Kernel Space

Algorithm 1 (Kernel) Convex Hull Approximation

Input:
The dataset of D-dimensional N data instances, X;
The parameter of Gaussian kernel, σ;
The expected number of extreme points, n;
The convergence criteria;

Output:
The approximated extreme points, X_{ep};

1: initialize C, where $c_j = \frac{1}{N}\cdot\mathbf{1}$ and $j \in \{1,2,\cdots,N\}$;
2: append $\omega = \sqrt{D}$ to each data instance in X;
3: set $K = K(X,X)$ using Gaussian kernel with σ;
4: **repeat**
5: set $C = C \circ \sqrt{\frac{K}{KC}}$;
6: **until** convergence criteria are met
7: set X_{ep} as the set (a matrix) of n data points whose corresponding values in $diag(C)$ are among the top n.
8: **return** X_{ep}.

Therefore, it is summarized that the multiplicative updating rule of Semi-NMF achieves convex hull approximation through identifying the extreme points based on the importance of a data instance in describing itself, which is reflected by the diagonal elements in C, i.e., $diag(C)$. The algorithm for identifying the extreme points of a dataset is formally presented in Algorithm 1. In practice, we adopt convex hull approximation rather than identification due to the reason that Semi-NMF takes too long to converge and approximated convex hull are sufficiently useful in related tasks.

Practically, the initialization of C in Algorithm 1 is to set every element in C to $\frac{1}{N}$, i.e., $c_j = \frac{1}{N}\cdot\mathbf{1}$. For the assignment of the weight ω (see Section III-B), we empirically adopt $\omega = \sqrt{D}$ to make sure that the constraint $\mathbf{1}^TC = \mathbf{1}^T$ will not be neglected when the dimension of the data is high. Concerning the convergence criteria, the standard stopping criteria of NMF is utilized, i.e., whenever the maximum change of the elements in C or that of the value of the target function in Eq. (6) is below a certain threshold, the updating stops. These settings are utilized in all our experiments for convex hull approximation, which is the basic building block of the Convex Hull Data Description.

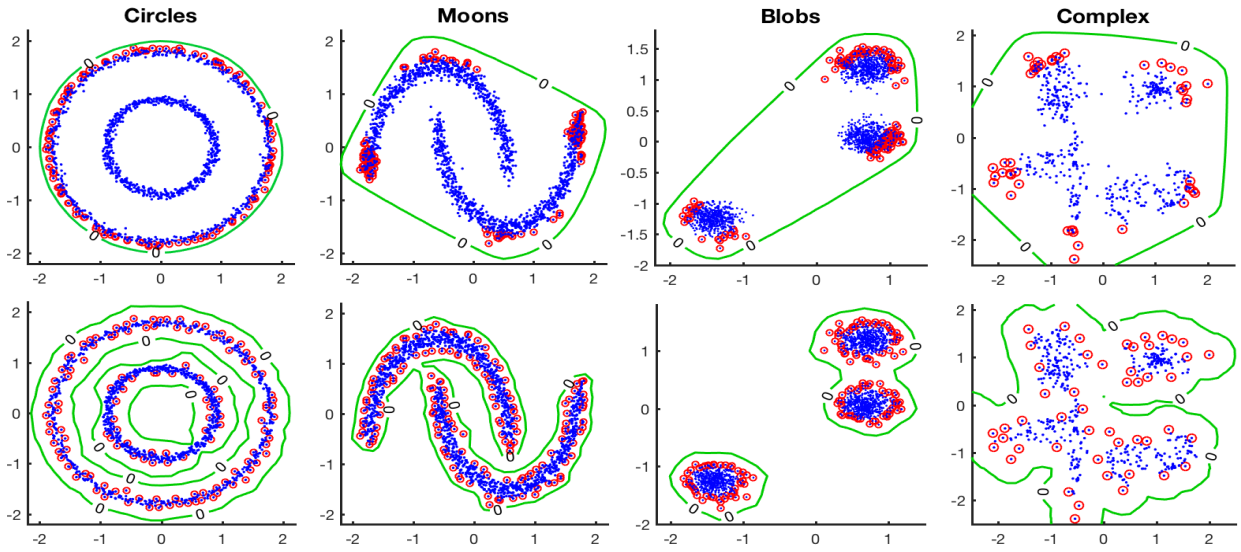

Fig. 3. The performance of convex hull one-class classification in four toy datasets. ($n = 0.1 \times N$, Gaussian kernel $\sigma = 0.3$, better view in color)

Algorithm 2 Reconstruction Coefficient and Error

Input:
The target dataset, X;
The source dataset, X_{ep};
The parameter of Gaussian kernel, σ;
The convergence criteria;
Output:
The reconstruction coefficient, C_{ep};
The reconstruction error, E;

initialize C_{ep};
append a weight ω to each data instance in X and X_{ep};
set $K_1 = K(X, X)$ using Gaussian kernel with σ;
set $K_2 = K(X, X_{ep})$ using Gaussian kernel with σ;
set $K_3 = K(X_{ep}, X_{ep})$ using Gaussian kernel with σ;
repeat
  set $C_{ep} = C_{ep} \circ \sqrt{\frac{K_2}{K_3 C_{ep}}}$;
until convergence criteria are met
set $E = diag(K_1) - 2 \cdot diag(K_2 C_{ep}) + diag(C_{ep}^T K_3 C_{ep})$.
return C_{ep} and E.

Algorithm 3 Convex Hull One-class Classification

Input:
The training dataset, X_{trn};
The testing dataset, X_{tst};
The estimated number of extreme points, n;
The parameter of Gaussian kernel, σ;
The convergence criteria;
Output:
The reconstruction errors of X_{tst}, E_{tst};
The threshold, ϵ;

run Algorithm 1 with X_{trn}, n, σ and the convergence criteria: extract approximated extreme points X_{ep};
run Algorithm 2 with X_{trn}, X_{ep}, σ and the convergence criteria: obtain the threshold $\epsilon = \max E$;
run Algorithm 2 with X_{tst}, X_{ep}, σ and the convergence criteria: measure the anomality of the data instances in X_{tst} using $E_{tst} = E$;
return ϵ and E_{tst}.

D. Convex Hull Data Description (CHDD)

Relying on convex hull approximation, CHDD is to extract the key features of a dataset in order to **(I)** determine whether a new data instance belongs to the dataset, i.e., one-class classification, and **(II)** separate the different groups in the dataset to form clusters, i.e., clustering. To this end, the whole dataset is once again described by only the approximated extreme points using the same optimization formulation:

$$\min_{C_{ep} \geq 0} \|\phi(X) - \phi(X_{ep}) C_{ep}\|^2, \tag{15}$$

where X_{ep} is a matrix obtained by running Algorithm 1 over the original dataset X and contains the approximated extreme points, while C_{ep} is the new coefficient matrix.

1) One-class Classification: To achieve one-class classification, the reconstructed error of each data instance is exploited as the key feature. And it is defined that

$$\epsilon = \max_i \|\phi(x_i) - \phi(X_{ep}) c_{ep,i}\|^2, \tag{16}$$

which is the upper threshold of all the reconstructed errors of the original data instances. Note that, $c_{ep,i}$ is the ith column vector in C_{ep}. Hence, any new data instance whose reconstructed error exceeds the threshold is considered to be an outlier that does not belong to the original dataset. Formally, for a new data instance x_{new}, only if it satifies the following constraint can it be regarded as a member of the original dataset:

$$\min_{c_{new}} \|\phi(x_{new}) - \phi(X_{ep}) c_{new}\|^2 \leq \epsilon. \tag{17}$$

The algorithms used to solve Eqs. (15) - (17) are essentially the same and formally given in Algorithm 2. When the target and source datasets are X and X_{ep}, respectively, the output E maintains the reconstruction errors for all the original data. Therefore, ϵ is the maximum element in E. While the target and source datasets are X_{new} and X_{ep}, respectively, the output E maintains the reconstruction errors of all the new data instances. A data instance with $e \in E$ and $e > \epsilon$ is considered as an anomaly.

It is summarized that the process of one-class classification in CHDD follows three key steps as shown in Algorithm 3. The detailed process starts with the utilization of Algorithm 1 to obtain the approximated extreme points. Afterward, Algorithm 2 is followed in order to extract the feature of the training dataset, i.e., the reconstruction errors of all the known data instances, assuming that they are within the same class. The maximum reconstruction error is chosen as the threshold for identifying anomalies. The second run of Algorithm 2 with a testing dataset will reveal the construction errors of the data in the testing dataset. Therefore, a further comparison between the errors and the threshold determines the result of one-class classification.

In Fig. 3, the performances and characteristics of convex hull one-class classification in four toy datasets are displayed. The blue dots are the original data instances from the datasets whereas the red circles emphasize the approximated extreme points. The green boundaries are the decision boundaries for anomaly detection. Data instances outside the boundaries will be detected as anomalies, while normal data instances should rest inside the boundaries. The four subfigures on the top of Fig. 3 firstly demonstrate convex hull one-class classification without using the kernel trick. The four subfigures on the bottom exhibit the effects of the Gaussian kernel with $\sigma = 0.3$ in the same task. And the parameters n in all the tasks are selected as $n = 0.1 \times N$. It is apparent that the employment of Gaussian kernel significantly enhances the performance of convex hull one-class classification and generalizes the method to be applicable in various datasets. It can also be observed from Fig. 3 that different from other methods such as SVDD the decision boundary of convex hull one-class classification is not tight. This is due to the fact that the maximum reconstruction error is chosen as the threshold and it does not always lead to tight boundaries. Although one-class classification is preferred to describe a dataset with a tight boundary, a relaxed boundary could be more practical in dealing with data noise and has demonstrated better accuracy in some tasks of anomaly detection [13].

2) Clustering: Besides one-class classification, another application of convex hull approximation, i.e., clustering, is made possible with a careful examination of the reconstruction coefficient matrix C_{ep} after the resolution of Eq. (15). Theoretically, C_{ep} encodes the coefficients of the extreme points for constructing the original dataset. Each column of C_{ep} holds the coefficients for constructing a specific data, while each row of C_{ep} reveals how much the data instances are dependent on the corresponding extreme point. According to the definition of the convex hull, data instances that profoundly rely on a same extreme point are expected to be in the same cluster. Therefore, based on this intuition, a thorough investigation of C_{ep} can identify clusters of the original dataset. We call this process convex hull clustering. The details of convex hull clustering are given in Algorithm 4. Specifically, after the executions of Algorithm 1 and 2, convex hull clustering examines each row of C_{ep} to identify new clusters or integrated known clusters that have intense connections. The intensity of the connections is governed by a threshold ε. It is noted that the first two steps of convex hull clustering are exactly the same as that in convex hull one-class classification. The actual work that forms the clusters is inside the loop, which goes through the rows of C_{ep}.

Algorithm 4 Convex Hull Clustering

Input:
The target dataset, X;
The estimated number of extreme points, n;
The parameter of Gaussian kernel, σ;
The convergence criteria;
The threshold of data relationship, ε;
Output:
The clustering label, L;

```
run Algorithm 1 with X, n, σ and the convergence criteria: extract approximated extreme points X_ep;
run Algorithm 2 with X, X_ep, σ and the convergence criteria: obtain the reconstruction coefficient C_ep;
initialize L, set the label of all data to 0;
for each row in C_ep do
    get the label set L_ε of the elements in the row whose value is greater than ε;
    if max L_ε == 0 then
        ∀l ∈ L_ε, set l to a new label;
    else
        ∀l ∈ L whose value appears in L_ε, set l = max L_ε;
    end if
end for
return L.
```

The performance of convex hull clustering in the four same toy datasets used in one-class classification is demonstrated in Fig. 4. The data instances are shown as dots and assigned different colors according to the clusters they belong to. The circles, squares, diamonds and triangles are emphsizing the approximated extreme points in the corresponding clusters. It is shown that it correctly identifies all the clusters of the toy datasets with the utilization of the Gaussian kernel and proper tuning of the parameters. In some situations, due to the fact that CHDD chooses a set of approximated extreme points rather than the real ones, convex hull clustering may generate clusters that have few data instances. These data instances are normally outliers. In this paper, it is assumed that the training dataset does not contain outliers. Therefore, in our experiments, all the clusters with few data are merged into their most related clusters.

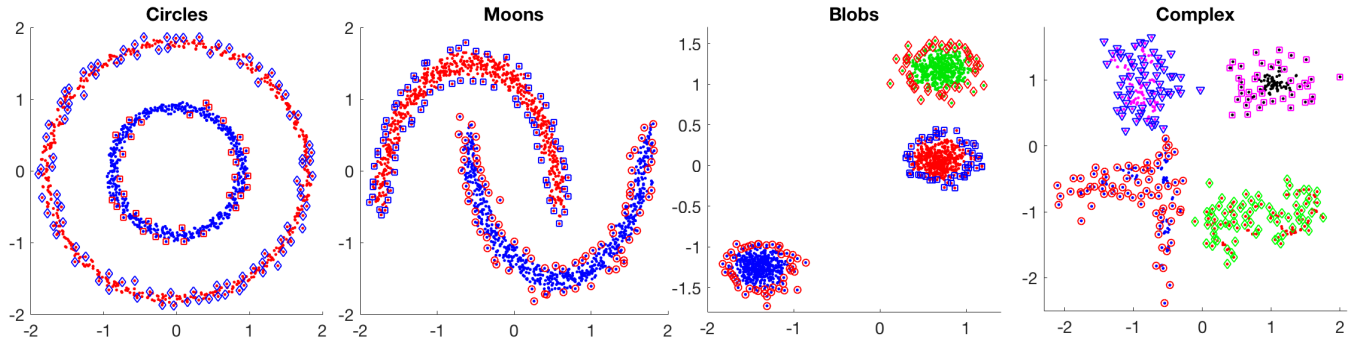

Fig. 4. The performance of convex hull clustering in four toy datasets. The first three subfigures use $n = 0.1 \times N$ and Gaussian kernel $\sigma = 0.2$, while the last subfigure uses $n = 0.5 \times N$ and Guassian kernel $\sigma = 0.18$. (better view in color)

IV. Experiment Results

In this section, experiment results[1] are presented to demonstrate the capability of Convex Hull Data Description (CHDD). The experiments are arranged in three parts, each of which targets at explaining a particular aspect of CHDD. To be more specific, the contents cover:

- The algorithms and results of convex hull approximation;
- The effectiveness of CHDD in one-class classification;
- The effectiveness of CHDD in clustering.

Note that in one-class classification and clustering all the CHDD experiments use the Gaussian kernel for convex hull approximation. The tunable parameters are the parameter σ of the Gaussian kernel, the estimated number of the approximated extreme points n, the convergence criteria and the threshold ε for data clustering. All the parameters of CHDD and other compared methods are selected by grid search from a set of candidates, which will be specified later, to get the best possible results in the tasks.

A. Convex Hull Approximation

As mentioned in Section III, some algorithms, e.g., Semi-NMF and RRI, are considered in this work to solve the problem in Eq. (6). The results of using Semi-NMF and RRI to determine the extreme points of four toy datasets are displayed in Figs. 5 and 6. Fig. 5 shows the different effects of Semi-NMF and RRI in identifying extreme points without the utilization of the Gaussian kernel. The subfigures in Fig. 5 depict the data instances using dots in different colors. Generally, the color of a data instance indicates how important it is in describing itself. The importance is measured by the diagonal elements $diag(C)$ in the coefficient matrix C, which is obtained after the execution of Algorithm 1. The data instances whose values of the corresponding diagonal elements in C are among the top 10% are marked as red dots. Those with lower value are represented as green (between 10%-40%), blue (between 40%-80%) and purple (between 80%-100%) dots, respectively. It is shown that the solutions of Semi-NMF (subfigures in the first row) correctly reflect the importance of the data instances. Namely, the data that are marked as red dots are more prone to be extreme points than the data marked with other colors. It is also interesting to discover that the values in $diag(C)$ reveal the relative positions of the data. The red dots are the data with the top 10% values in $diag(C)$. They are always located on and near the edge of the dataset. Compared to the red dots, the green ones are closer to the center of the dataset. The purple circles are nearest to the center. This result confirms that the extreme points of the given dataset, which typically lie on the edge of the dataset, tend to have higher values in $diag(C)$. Hence, the utilization of Semi-NMF in identifying the extreme points nicely approximates the convex hull given the approximated number of extreme points n.

Besides Semi-NMF, we also investigate RRI in solving Eq. (6) due to its reported efficiency in related work and its capability of leveraging the kernel trick. Nonetheless, the solutions of RRI depicted in the subfigures in the second row of Fig. 5 show that it does not provide satisfactory results in convex hull approximation. The red dots, which should pinpoint extreme points, do not capture the edge of the dataset. Additionally, the distribution of the colored dots does not have a clear pattern, which shows that $diag(C)$ obtained by RRI does not provide valuable information in identifying the extreme points. Specifically, the utilization of RRI in solving Eq. (6) is essentially an updating rule for each row vector c_i' in C, i.e., $C \triangleq [c_1';\ \cdots\ ;\ c_N']$:

$$c_i'^{,k+1} = \frac{[x_i^T R_i^k]_+}{x_i^T x_i}, \tag{18}$$

where $R_i^k = X - \sum_{j \neq i} x_j c_j'$ is called the residue. The kernelized version with the Gaussian kernel is of the form:

$$c_i'^{,k+1} = K(x_i, X) - K(x_i, X)C^k + c_i'^{,k}. \tag{19}$$

Through the operation $[\cdot]_+$, RRI maintains the non-negativity of the coefficient matrix C in each iteration and ensures convergence [5]. Although the updating process always leads to a smaller value of the target function, the updating of the row vector c_i' does not necessarily lead to a satisfactory solution of the column vectors c_i, which are much more important in convex hull approximation considering $x_i = Xc_i$.

[1]The source codes of the experiments are available in https://github.com/ (Not detailed due to the triple blind submission guidelines of ICDM).

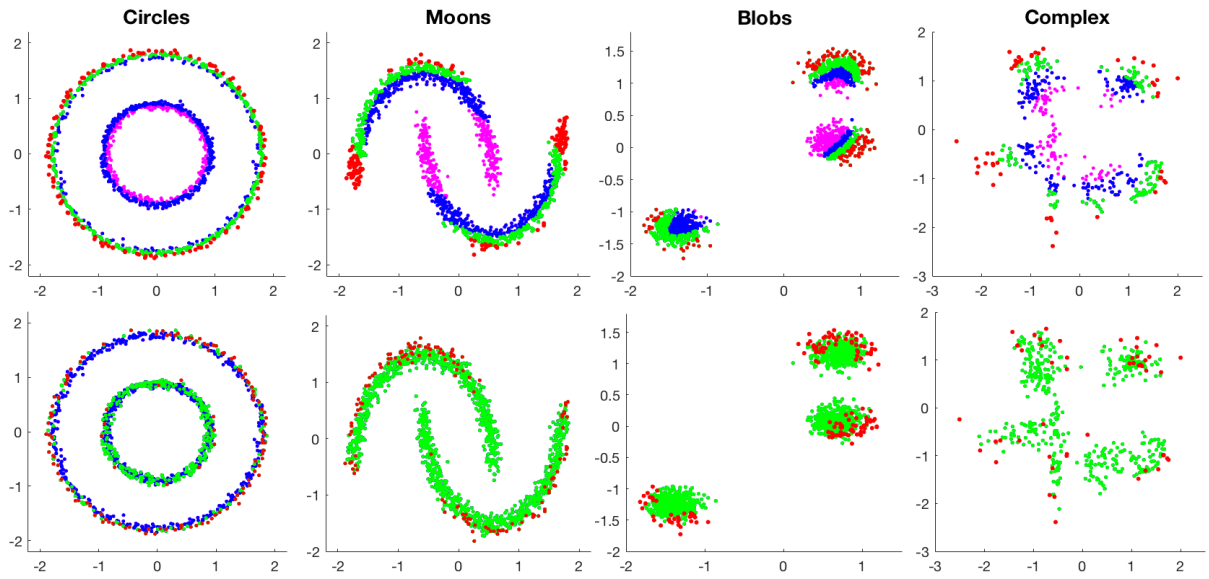

Fig. 5. The performance of Semi-NMF (1st row) and RRI (2nd row) in convex hull approximation of four toy datasets. (better view in color)

Consequently, the extreme points identified by RRI do not grasp the concept of the convex hull and are distinct from those found by Semi-NMF.

In addition to Fig. 5, Fig. 6 depicts the results of convex hull approximation with the utilization of the Gaussian kernel in Semi-NMF (first row) and RRI (second row). The difference between the results obtained through Semi-NMF and RRI is clear in the datasets, especially in "Blobs", where Semi-NMF produces a decent set of potential extreme points, while RRI points out many erroneous ones. To conclude with, it is shown that Semi-NMF demonstrates better performance in both linear and nonlinear cases of convex hull approximation. Therefore, it is employed as the main algorithm in CHDD.

TABLE I
THE DATASETS FOR ONE-CLASS CLASSIFICATION

Name	#Dimension	#Class	#Target	#Outlier
iris	4	3	50	100
wine	13	3	59	119
breast	9	2	458	241
car	6	4	384	1344
biomed	5	2	67	127
diabetes	8	2	268	500
sonar	60	2	111	97
breastdiag	30	2	357	212
glass	9	4	70	214
liver	6	2	145	345
ionosphere	34	2	225	351
imox	8	4	48	192
auto_mpg	6	2	229	398
chromo	8	24	42	1143
ecoli	7	8	143	336

B. One-class Classification

1) Datasets and Methods: To examine the effectiveness of CHDD in one-class classification, 15 different datasets from the UCI data repository [10] are tested. The details of the selected datasets are illustrated in Table I. For each dataset, we firstly normalize the data and then pick the first class as the target class. All the data in other classes are considered as anomalies. In each experiment, we randomly sample 90% of the data instances from the target class for model training. The remaining 10% of the target data and all the anomalies are used as the testing dataset. The detailed list of the tested methods is provided in Table II. Apart from CHDD, 8 different methods from different categories are tested: (1) Support Vector Data Description (SVDD); (2) Gaussian Mixture Model (GMM); (3) Parzen-Window Density Estimation (Parzen); (4) Principle Component Analysis (PCA) for anomaly detection; (5) K-means; (6) K-Nearest Neighbour (KNN); (7) Local Outlier Factor (LOF); and (8) Linear Programming Data Description (LPDD). In each of the methods, there is one or several parameters. We prepare a set of candidates for each of the parameters and employ a grid search to find the best model parameter for each method. During the grid search, for each parameter setting, a 5-folds cross-validation is performed upon the training dataset to gain the average performance. More specifically, for the methods that use kernels, e.g., CHDD, SVDD and LPDD, we employ Gaussian kernel and adopt the same parameter $\sigma^2 \in [0.1, 0.3, \cdots, 1.9] \cdot D$, where D indicates the dimension of the training dataset. The parameter σ^2 is also adopted as the width parameter in Parzen. For GMM, K-means, KNN and LOF, $k \in [1, 2, \cdots, 10]$ is utilized as the number of clusters or the number of neighbors. PCA picks the number of primary component from the set $c \in [1, 2, \cdots, \sqrt{D}]$. The rejection fraction (outlier fraction in training dataset) of all the methods are set to 0 because the training datasets do not contain outliers. Additionally, CHDD considers the fraction of the extreme points from the set $n \in [0.05, 0.1, \cdots, 0.5]$ and uses the same convergence criteria as K-means.

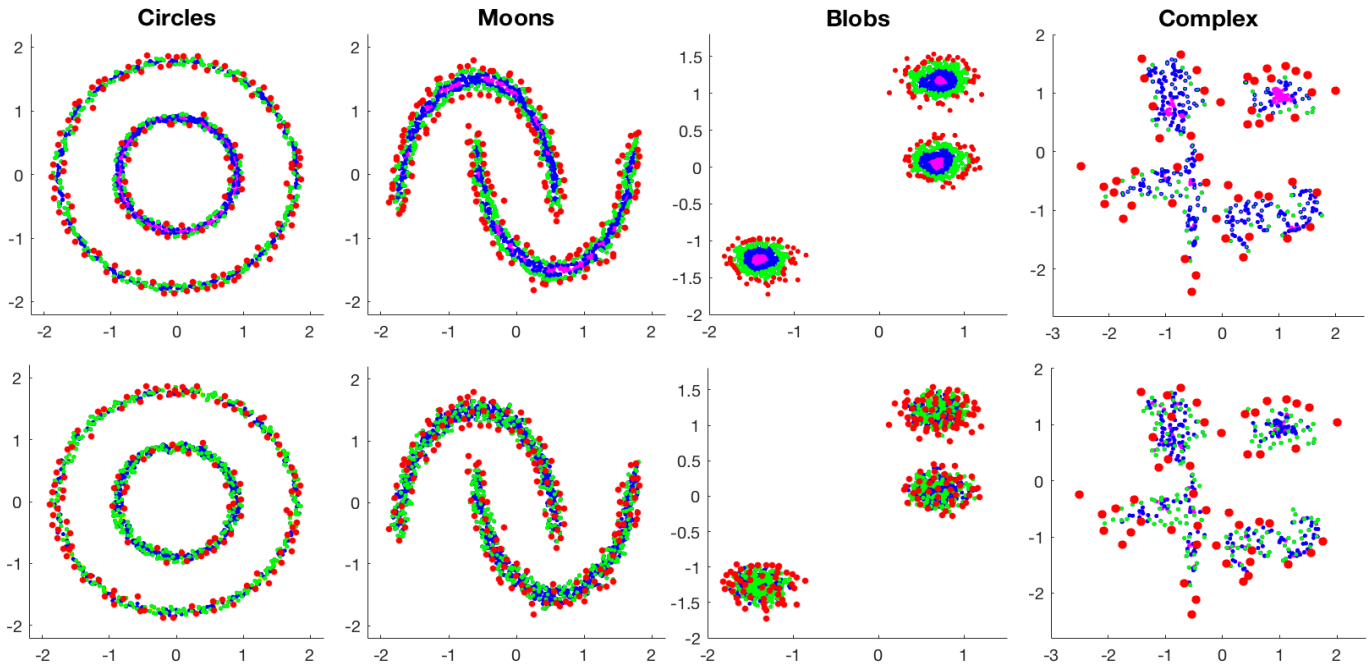

Fig. 6. The performance of Semi-NMF (1st row) and RRI (2nd row) in kernel convex hull approximation of four toy datasets. All the experiments use Gaussian kernel with $\sigma = 0.3$ and $n = 0.1 \times N$. (better view in color)

TABLE II
THE RESULTS (AUC) OF ONE-CLASS CLASSIFICATION IN UCI DATASETS

	CHDD	SVDD	GMM	Parzen	PCA	K-means	KNN	LOF	LPDD
iris	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
wine	**0.99**	**0.99**	**0.99**	**0.99**	0.98	**0.99**	**0.99**	**0.99**	**0.99**
breast	**0.99**	**0.99**	0.98	**0.99**	0.98	**0.99**	**0.99**	0.70	**0.99**
car	**0.99**	**0.99**	0.98	0.98	0.92	0.96	**0.99**	0.97	**0.99**
biomed	**0.73**	0.71	0.65	0.70	0.65	0.71	0.56	0.62	0.71
diabetes	**0.72**	0.63	0.53	0.53	0.55	0.55	0.46	0.54	0.62
sonar	0.73	0.73	0.71	0.65	0.66	0.68	0.72	**0.81**	0.72
breastdiag	0.92	**0.94**	0.93	0.90	**0.94**	0.92	0.90	0.93	0.93
glass	0.81	0.82	0.83	0.72	0.80	0.79	0.81	**0.87**	0.79
liver	0.55	**0.60**	**0.60**	0.59	**0.60**	0.58	**0.60**	0.59	0.57
ionosphere	0.97	0.97	0.96	0.89	**0.98**	0.95	0.97	0.94	0.97
imox	0.91	0.97	**0.98**	0.96	0.88	0.94	0.97	0.90	0.97
auto_mpg	0.84	0.88	0.92	0.92	0.81	0.91	**0.93**	0.64	0.86
chromo	0.95	**0.96**	0.94	**0.96**	0.94	0.95	0.95	0.95	**0.96**
ecoli	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**

2) Results: The performances of all the one-class classification methods in the selected UCI datasets are presented in Table II. Each result is the average performance of AUC, i.e. area under the curve, over 10 runs of the corresponding method. It is indicated by the results that there is no "best" method in one-class classification, because no method outperforms all the other methods in all the datasets. For the datasets of "iris", "wine", "breast", "car" and "ecoli", CHDD is among the methods that demonstrate the best performance. For the datasets of "biomed" and "diabetes", CHDD outperforms all the other methods. While, in "sonar", "ionosphere" and "chromo", CHDD ranks the 2nd place in the performance. For the other datasets, CHDD ranks 3rd to 5th places. Generally speaking, CHDD is competitive in the effectiveness of one-class classification compared to other methods.

C. Clustering

1) Datasets and Methods: Similar to one-class classification, 8 datasets are selected for the purpose of measuring the effectiveness of CHDD in clustering tasks. The datasets are from UCI [10] and Comprehensive R Archive Network (CRAN) repositories [20]. The first two rows in Table III present the names of the datasets as well as their basic information, i.e., the number and dimension of the data instances and the number of classes in each dataset. The compared methods contain Kernel K-means, DBSCAN, and the linkage clustering method. All these methods are experimented after the normalization of the datasets. A number of parameter settings are tested for each method to find the best clustering results, which are measured by Adjusted Mutual Information (AMI) [18].

TABLE III
THE RESULTS (AMI) OF CLUSTERING IN UCI AND CRAN DATASETS

(num,dim,#class)	**Motor** (94,3,3)	**Prest.** (98,5,3)	**Maps** (429,3,10)	**Image Seg.** (210,19,7)	**Pen** (1000,16,10)	**User** (403,5,5)	**DrivFaceD** (606,6400,3)	**Move. Libras** (360,90,15)
CHDD	**1**	**0.55**	**0.84**	0.57	0.62	0.26	**0.44**	0.53
Kernel K-means	0.70	0.53	0.76	**0.69**	**0.76**	**0.27**	0.08	**0.64**
DBSCAN	**1**	0.52	0.82	0.52	0.53	0.24	0.00	0.20
Linkage Clustering	**1**	0.52	0.82	0.12	0.02	0.04	0.10	0.06

To be more specific, CHDD selects the fraction of the extreme points from the set $[0.05, 1, \cdots, 1]$ and adopts Gaussian kernel with $\sigma^2 \in [0.1, 0.3, \cdots, 1.9] \cdot D \cup [0.05, 0.1, \cdots, 1]$, where D is the dimension of the dataset. Kernel K-means has the same kernel setting and sets the number of clusters $K \in [1, 2, \cdots, 15]$. DBSCAN picks the distance parameter from $[0.05, 0.1, \cdots, 1]$ and the minimum number of data points from the set $[2, 3, 7, 10, 15, 20, 25, 50, 75]$. And, linkage clustering utilizes default distance settings and selects cluster numbers also from $[1, 2, \cdots, 15]$.

2) Results: As indicated in Table III, the clustering performance of CHDD is among the best in the dataset "Motor" and it outperforms all the compared methods in 3 datasets, i.e., "Prest.", "Maps" and "DrivFaceD". For datasets "Image Seg.", "Pen", "User" and "Move. Libra", Kernel K-means shows outstanding performance. The performances of CHDD rank the second places for these four datasets. To conclude with, the results demonstrate that CHDD has a good capability in data clustering.

D. Potential Advantages and Future Work

1) Interpretability: The interpretability of CHDD comes from the fact that the coefficient matrix in CHDD explains each data instance with a linear combination of the approxiated extreme points. As a result, the clustering decision made for each data instance is interpretable, i.e., a data instance is assigned to a specific cluster because it can be described by a linear combination of the approximated extreme points in the cluster. This property is preferred especially when the results of data description are to be checked.

2) Incremental CHDD: As a product of CHDD, the set of approximated extreme points is the key for one-class classification as well as clustering. When considering a streaming dataset, the essential of utilizing CHDD is to maintain an adequate evolving set of the approximated extreme points. For instance, if a new data instance is considered as an outlier, it can be added into the set so that similar data are considered as normal afterwards. On the other hand, if the new data is normal, the set shall stay unchanged. Therefore, incremental CHDD can be achieved through these operations. We leave the detailed design, analysis and application of incremental CHDD as our future work.

V. CONCLUSION

In this paper, a novel method called Convex Hull Data Description (CHDD) is proposed. Three aspects of CHDD, i.e., convex hull approximation, one-class classification and clustering are elaborated. The convex hull approximation is innovatively achieved through Semi-Nonnegative Matrix Factorization (Semi-NMF), which enables the utilization of the kernel trick to support one-class classification and clustering. Our experiment results have demonstrated that CHDD successfully pinpoints the approximated extreme points and with the Gaussian kernel it is highly competitive in both one-class classification and clustering tasks in terms of the effectiveness.

REFERENCES

[1] K. P. Bennett, E. J. Bredensteiner, Duality and Geometry in SVM Classifiers, *ICML*, pp. 57-64, 2000.

[2] J. L. Bentley, F. P. Preparata, M. G. Faust, Approximation algorithms for convex hulls, *Commun. ACM*, vol. 25, no. 1, pp. 64-68, 1982.

[3] C. Ding, T. Li, M. I. Jordan, Convex and Semi-Nonnegative Matrix Factorizations, *TPAMI*, vol. 32, no. 1, pp. 45-55, 2010.

[4] E. Elhamifar, G. Sapiro, R. Vidal, See all by looking at a few: Sparse modeling for finding representative objects, *CVPR*, pp. 1600-1607, 2012.

[5] N. D. Ho, Nonnegative matrix factorization algorithms and applications, Doctoral dissertation, Ecole Polytechnique, 2008.

[6] M. Kemmler, E. Rodner, E. S. Wacker, J. Denzler, One-class classification with Gaussian processes, *Pattern Recognition*, vol. 46, no. 12, pp. 3507-3518, 2013.

[7] T. Kutsuna, A Binary Decision Diagram-Based One-Class Classifier, *ICDM*, pp. 284-293, 2010.

[8] P. Laskov, C. Schfer, I. V. Kotenko, K. R. Mller, Intrusion Detection in Unlabeled Data with Quarter-sphere Support Vector Machines, *Praxis Der Informationsverarbeitung Und Kommunikation*, vol. 27, no. 4, pp. 228-236, 2004.

[9] D. D. Lee, H. S. Seung, Algorithms for Non-negative Matrix Factorization, *NIPS*, pp. 556-562, 2001.

[10] M. Lichman, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml, University of California, Irvine, 2013.

[11] F. T. Liu, K. M. Ting, Z. H. Zhou, Isolation-Based Anomaly Detection, *TKDD*, vol. 6, no. 1, pp. 3-39, 2012.

[12] J. Ma, S. Perkins, Time-series novelty detection using one-class support vector machines, IJCNN, pp. 17411745, 2003.

[13] T. Mu, A. K. Nandi, Multiclass Classification Based on Extended Support Vector Data Description. *IEEE T SYST MAN CY B*, vol. 39, no. 5, 2009.

[14] H. Sartipizadeh, T. L. Vincent, Computing the Approximate Convex Hull in High Dimensions, arXiv:1603.04422, 2016.

[15] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the Support of a High-Dimensional Distribution, *Neural Computation*, vol. 13, no. 7, pp. 1443-1471, 2006.

[16] D. M. J. Tax, R. P. W. Duin, Support Vector Domain Description. *Pattern Recognition Letters*, vol. 20, no. 11-13, pp. 1191-1199, 1999.

[17] D. M. J. Tax, DDtools - the Data Description Toolbox for Matlab, *http://prlab.tudelft.nldavid-taxdd_tools.html*, version 2.1.2, 2015.

[18] N. X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: is a correction for chance necessary?, ICML, pp. 1073-1080, 2009.

[19] D. Wang, H. Qiao, B. Zhang, M. Wang, Online Support Vector Machine Based on Convex Hull Vertices Selection, *TNNLS*, vol. 24, no. 4, pp. 593-609, 2013.

[20] Comprehensive R Archive Network (CRAN), https://cran.r-project.org/.