World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:7, No:4, 2013

Grid–SVC: An improvement in SVC algorithm, based on grid based clustering

Farhad Hadinejad, Hasan Saberi, Saeed Kazem

Abstract—Support vector clustering (SVC) is an important kernel-based clustering algorithm in multi applications. It has got two main bottle necks, the high computation price and labeling piece. In this paper, we presented a modified SVC method, named Grid–SVC, to improve the original algorithm computationally. First we normalized and then we parted the interval, where the SVC is processing, using a novel Grid–based clustering algorithm. The algorithm parts the intervals, based on the density function of the data set and then applying the cartesian multiply makes multi-dimensional grids. Eliminating many outliers and noise in the preprocess, we apply an improved SVC method to each parted grid in a parallel way. The experimental results show both improvement in time complexity order and the accuracy.

Keywords—Grid–based clustering, SVC, Density function, Radial basis function.

I. Introduction

THe process of grouping a set of data points into classes of similar data is called clustering. Clustering of high dimensional data points is a famous concept in data mining. Lately lots of methods are introduced due to various goals and ideas. Existing clustering algorithms can be classified into five kinds [1]. The main idea of *partitioning* methods for a given database of N nodes are to construct k partitions of the data, where each partition represents a cluster. K-Means [2] and P-Median [3], the kernel based clustering methods have such scheme [1]. That is, it classifies the data into k groups, and then to make the clusters as compact and separated as possible. *Hierarchical* algorithms, consider each data point as a cluster, and then group them sequentially by similarity measures. And also the *Grid–based* methods quantize the object space into a finite number of cells that form a grid structure. All of the clustering operations are performed on the grid structure and model based hypothesize a model for each of the clusters and find the best fit of the data to the given model. The *density based* methods for example DBSCAN [4] are using the idea of detecting the highest dense areas upon the space the clusters are relatively belongs [1].

In 2001, Ben-Hur et al. [5] introduced a kernel based clustering algorithm named SVC. The SVC method used a nonlinear transform of the data points into Hilbert space [6]. Although the accuracy, SVCs popularity is degraded by its pricy computation and poor labeling performance. Different from existing modifications that only resolve the two bottle necks [7], [8], [9].

In this paper, we introduced a novel method named Grid–SVC which improves the original SVC and makes it more efficient in both time computing and labeling piece. The algorithm first applies a novel Grid–based clustering based on density function on normalized data points in $[-0.5, 0.5]$, then initials some inaccurate clusters (grid), which are mostly noise–free. In the initial process it is tried to eliminate the outlier points and scale down the interval where SVC wants to run. The other reason is to parallelize the algorithm. We first obtain some grids (trivial cluster) where have the potential of containing the dense areas in the environment, Then by applying the **SA** algorithm to the labeling piece [7] for the grids separately and in parallel way, the algorithm optimize the original SVC algorithm. The experimental results show both time efficiency and accuracy for the novel algorithm.

In the novel Grid–based clustering algorithm, we introduced a density function based on radial basis function (RBF). The radial basis function method for multivariate approximation is one of the most often applied approaches in modern approximation theory when the task is to approximate irregularly positioned points in several dimensions [10]. Consequently, it is no longer a surprise that in many applications, radial basis functions have been shown to be most useful. There are many applications especially in the sciences and in mathematics, they include. For example, mappings of two- or three-dimensional images such as portraits or underwater sonar scans into other images for comparison [10]. In this important application, interpolation comes into play because some special features of an image may have to be preserved while others need not be mapped exactly.

In this paper we construct the density function using the RBF, then approximate it with a polynomial function using famous Taylor expansion around RBF's center nodes. The reason is, the RBFs have a high ability to preform a distribution function from random initial points.

The structure of this paper is as follows: First we introduced a novel Grid–based clustering algorithm, based on density function for each data dimension separately. Then we introduced SVC and a modification on it. Next we introduce a novel SVC algorithm (Grid–SVC) which applies the grid based clustering to improve SVC efficiency and at the end the experimental result are discussed.

II. Related Works

In 2005, Lee et al. [9] introduced an improved SVC, a novel algorithm to improve original SVC. They applied the gradient

F. Hadinejad is with the Department of Industrial Engineering, Shomal University, Amol, Iran (e-mail: farhad_hdng@yahoo.com).

H. Saberi is with the Department of Computer Science, Shahid Beheshti University of Tehran, Iran (e-mail: dz.saberi@gmail.com).

S. Kazem is with the Department of Mathematics, Imam Khomeini International University, Ghazvin 34149-16818, Iran (corresponding author to provide phone:+989354592405; e-mail: saeedkazem@gmail.com).

vector of each SV, and produced a algorithm based on that they improved the labeling piece of SVC. And also in 2009, Ping et al. [7] introduced improvedSVC (iSVC). The algorithm reduced the total point involved the process of obtaining parameter β for the points, using schrödinger equation. The algorithm divided the interval of data points, based on the potential, the points had, calculating the schrödinger equation. Then from each intervals, picks some candidate points, then applied the SVC for the mentioned points. At the labeling piece, it used a algorithm named **SA** (see section 4) which is applied in our method too.
In 2008, Sun et al. [8] introduced a novel SVC based on k-means [2]. In the algorithm, firstly, SVC algorithm is employed to identify some samples as outliers and some others as intra-cluster points, then the method uses Minimum Spanning Tree Pruning (MSTP) strategy on those intra-cluster points to initialize the number of clusters and finally, it runs k-means on subset without outliers.

III. Grid–based clustering

The grid–based clustering approach uses a multi resolution grid data structure. It quantizes the object space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. Recently some typical examples of the Grid–based approach (i.e. PROCLUS [11], and CLIQUE [12]) are introduced. The CLIQUE algorithm represents a Grid– and density–based approach for clustering in high–dimensional data space. The clustering process starts at single-dimensional subspaces and grows upward to higher-dimensional ones. CLIQUE parts each dimension, like a grid structure (non overlap units) and determines whether a cell is dense based on the number of points it contains, then a cluster is defined as a maximal set of connected dense grid. In PROCLUS algorithm, instead of starting from single-dimensional spaces, it starts by finding an initial approximation of the clusters in the high-dimensional attribute space. Here, we introduced a novel algorithm based on density function.

A. Grid–based clustering based on density function

To find the best d-dimensional rectangles (grid), contain the most aggregated areas of data points. We introduced a novel method base on approximating the distribution of density function, for each data dimension separately. At first, the Parallel environment programming technique is discussed and at the second we obtain the best grids using density function.

1) Parallel environment programming technique: The main foundation of a parallel environment is the agents. An agent is a program, which can act independently and autonomously. A micro agent is an agent, which can perform in a simplified subset of a whole environment. It can cohabitant, transfer, or co-allocate the resources with other micro agents. When a Micro agent's task is done, it can release its resources or accept other arrangement. The structures of the micro agent are distinct for the different system [13].
However, there are two parts generally speaking, which are MA (Micro Agent) and MAE (Micro Agent Environment). MAE realizes the performance of the MA among the others, using agent transfer protocol and distributes the executing environment and service interface to them. It also controls the security, communication, basic service and so on. MA is above the MAE and it can be applied into another MAE. MA can communicate with other MA using agent communication language.

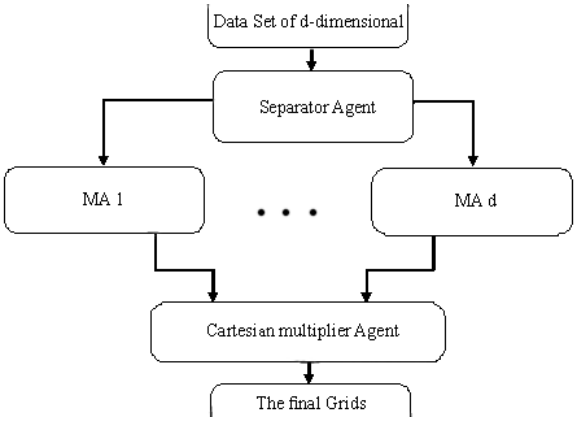

Fig. 1. Using d agents we can cluster a d-dimensional data set in a parallel algorithm.

Fig1 shows a simple MAE of d MAs. The agents work in parallel. The separator agent normalized the data and separate the data point of d dimensions into d data sets of one dimension. For each MA, there is a clustering algorithm which is applied only for 1-dimensional data set. The separator agent departed a d-dimensional data set into d times 1-dimensional, and gives each one to one MA. The cartesian multiplier agent, consider the output of each agent as an interval and produces the final grids, using cartesian multiply.

2) Obtaining the best grids using density function: In previous section, we discussed about d times 1-dimensional clustering algorithm in parallel, now we introduced an 1-dimensional clustering algorithm that can be applied by each MA.
Consider a data set of N data points of d-dimensions as $x_i, i = 1..N$, for each dimension $j, j = 1...d$, we approximate the distribution of density function for dimension j as

$$F_j(x) = \sum_i f(r_{ij}, \epsilon), \tag{1}$$

where $f(r, \epsilon)$ is a RBF and $r_{ij} = \|x_j - x_{ij}\|$ (see Appendix). Variant forms of RBF are introduced in TabIII. Because of simplicity, the Gaussian from (GA) is applied, thus $f(r) = exp(-\epsilon r^2)$. Other RBFs can be applied by little differences. In order to obtain the aggregated areas for each dimension, we must obtain the intervals where function

$$F_j(x) - M_{N_j}, \tag{2}$$

is non–negative.
M_{N_j} is a positive value and a function of the number of data points (N). To obtain the solutions of Eq.(2)≥ 0, we do as follows, if $t_n(f(x), x_0)$ be the Taylor expansion of order n, for function $f(x)$, around point x_0, then we can write Eq.(1)

as

$$F_j(x) \approx P_{n_j}(x) = \sum_{i}^{N} t_n(f(r_{ij}, \epsilon), x_i). \quad (3)$$

where $P_{n_j}(x)$ is a polynomial of order n, in j^{th} dimension. Identifying the two parameters n and ϵ, is the most important issue in the approximation process. The ϵ affects on sharpness of density function. Thus, decreasing of ϵ, we have better approximation with the same n. As n is the order of Taylor expansion, the higher n leads to less error. To obtain the solution of Eq.(2), the time complexity order is strongly depends on n.

In order to improve the approximation process using the Taylor expansion we normalized the data points in to $[-0.5, 0.5]$. It leads the approximation more accurate with lower value of n. Thus we have

$$x_i = \frac{I_i - min_i(I_i)}{max_i(I_i) - min_i(I_i)} - 0.5, \quad (4)$$

where I_i is i^{th} input value.

Remark 1: Using the famous Newton-Raphson method, the roots of Eq.(2)=0 in $[-0.5, 0.5]$, by subsaturation of $F(x)$ with $P(x)$, with m iterations for α roots, can be obtained in $O(\alpha.m)$.

Remark 2: Considering the trade off between accuracy and speed, we can obtain an upper bound for n, as $\alpha.m \leq n \leq N$. Figure 2 shows the process of obtaining the solution of Eq.(2)$\geq$ 0. The data set which is used, is the famous Iris data set by Fisher [14] (In the Experimental results section the attributes of the data set is discussed in details). The value of M_{N_j} in this Figure is

$$M_{N_j} = N \frac{\int_{-0.5}^{0.5} f(\|x\|^2, \epsilon) + f(\|x - 0.5\|^2, \epsilon) dx}{2}.$$

Two followed observations can introduced the whole algorithm:

Observation 1

Given a normalized data set of N data points of d-dimensions, the areas of the most aggregated data points in j^{th} dimension, are of the intervals of form $[a_{i_j}, a_{i+1_j}]$ where a_{i_j}s are the roots of Eq.(2)=0 in $[-0.5, 0.5]$ and $\forall x \in [a_{i_j}, a_{i+1_j}], Eq.(2) \geq 0$.

Observation 2

For a given data set, described in previous observation, the d-dimensional areas (grids) of most aggregated data points are the cartesian multiply of the intervals obtained from previous observation (see Figure 3). To obtain the aggregated areas which are not specifically rectangular we can define hybrid grids as join of some neighbors which contains many data points.

IV. SVC Algorithm

Support Vector Clustering algorithm looks for the smallest sphere in the Hilbert space that encloses the image of the data [6], [5]. This sphere is mapped back to data space, where it forms a set of contours which enclose the data points. These contours are interpreted as cluster boundaries.

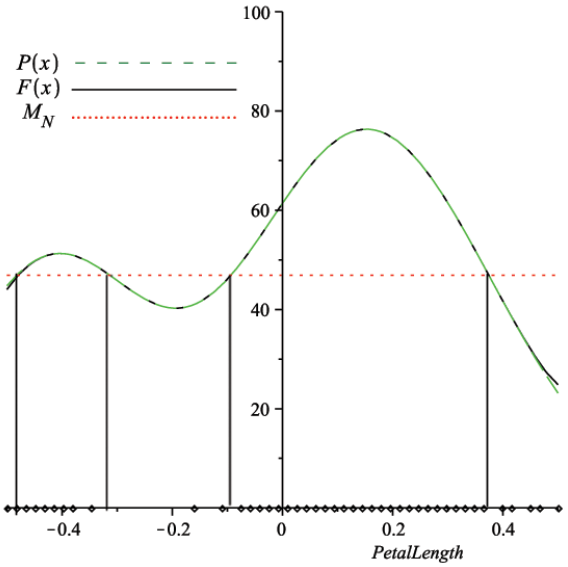

Fig. 2. Approximating of function $F_j(x)$ by $P_j(x)$, using Taylor expansion. The data set is the set of PetalWidth of Iris data set, normalized in interval $[-0.5, 0.5]$. The solution of $P_j(x) - M$ identifies the aggregated intervals. We initial the value of $\epsilon = 18$ for obtaining actual $F(x)$. For the Taylor expansion we used $n = 20$, to approximate $F(x)$ and obtain $P(x)$.

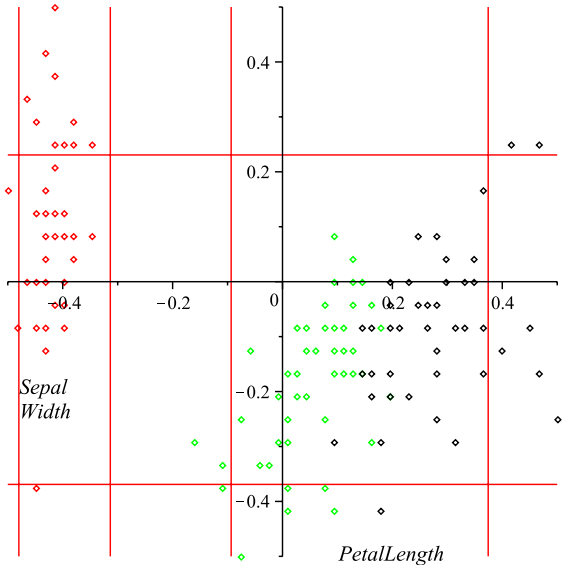

Fig. 3. The most aggregated areas for 2 dimensions of Iris data set, obtained by the Second observation for two of four dimensions of the Iris data set.

A. Description

Given a nonlinear transformation ϕ for a d-dimensional data point $x \in \mathcal{R}^d$ as $\phi(x)$, the distance between the transformed data point and the center of the sphere at the feature space is defined by:

$$\| \phi(X_j) - a \|^2 \leq R^2 + \xi_i, \;\; j = 1...N, \;\; \forall i, \xi_i \geq 0. \quad (5)$$

where where $\| . \|$ is the Euclidean norm, R is the radius, a is the center of the feature space mapped by the data points and ξ_i are the slack variables. To solve the Eq.(5) we apply

Lagrangian [5]

$$L = R^2 - \sum_j (R^2 + \xi_j - \| \phi(X_j) - a \|^2)\beta_j - \sum_j \xi_j\mu_j + C\sum_j \xi_j$$

where $\beta_j \geq 0$ and $\mu_j \geq 0$ are Lagrange multipliers, C is a constant, and $C\sum \xi_j$ is a penalty term. Setting to zero the derivative of L with respect to R, a and ξ_j, respectively, leads to

$$\sum_j \beta_j = 1 \tag{6}$$
$$a = \sum_j \beta_j \phi(X_j)$$
$$\beta_j = C - \mu_j \quad so \quad 0 \leq \beta_j \leq C,$$

for $j = 1...N$.
The KKT complementarity conditions of Fletcher [15] result in

$$\xi_i\mu_i = 0 \tag{7}$$
$$(R^2 + \xi_j - \| \phi - a \|^2)\beta_j = 0 \ .$$

To obtain the β_j we eliminate the the variables R, a and μ_j , turning the Lagrangian into the Wolfe dual form that is a function of the variables β_j [5]

$$W = 1 - \sum_i \sum_j \beta_i\beta_j K(x_i, x_j). \tag{8}$$

where $K(x, y) = \exp(-q \| x - y \|^2)$. Derivation with respect to β_j and considering the condition of Eq.(6) leads to

$$\beta_{n\times 1} = [A]^{-1}_{n\times n} B_{n\times 1}, \ \beta = [\beta_1..\beta_n]^T. \tag{9}$$

At each point X, we define the distance of its image in feature space from the center of the sphere

$$R^2(X) = \| \phi(X) - a \|^2 \ . \tag{10}$$

In view of quadratic equation and the definition of the kernel [5] the following is got

$$R^2(X) = 1 - 2\sum_j \beta_j K(X_j, X) + \sum_i \sum_j \beta_i\beta_j K(X_i, X_j). \tag{11}$$

The radius of the sphere is $R = \{R(x_j)\}$, x_j is support vector. The contours the enclose the points in data space are defined by the set $\{x|R(x) = R\}$.

B. Cluster Analysis

The number of **outlier** points is controlled by the parameter C, $N_{BSV} < 1/C$, where N_{BSV} is the number of Bounded Support Vectors (BSVs). Thus $1/(CN)$ is an upper bound on the fraction of BSVs, then let $1/(CN) \in (0, 1]$. The value of the parameter C is relation to the number of the data points, the q of the $K(x, y)$ is width parameter of Gaussian kernel function, and the q and C influences tightness and number of clustering each other.

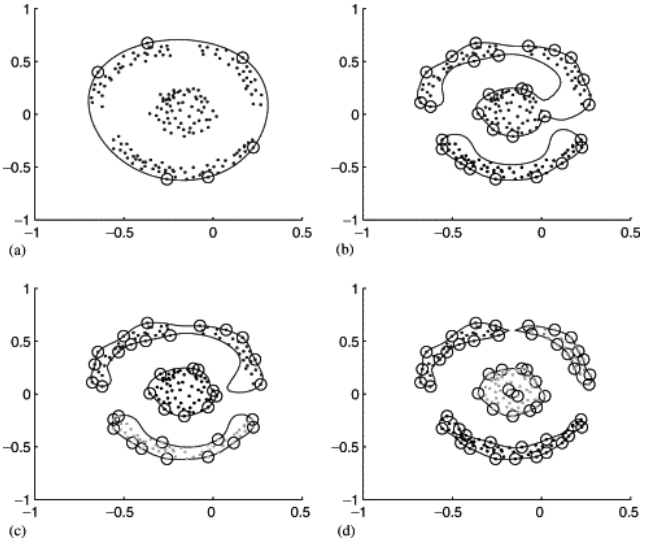

Fig. 4. Clustering of a data set containing 183 points using SVC with C =1. Support vectors are designated by small circles, and cluster assignments are represented by different gray scales of the data points. (a) q=1, (b) q=20, (c) q=24, (d) q=48.

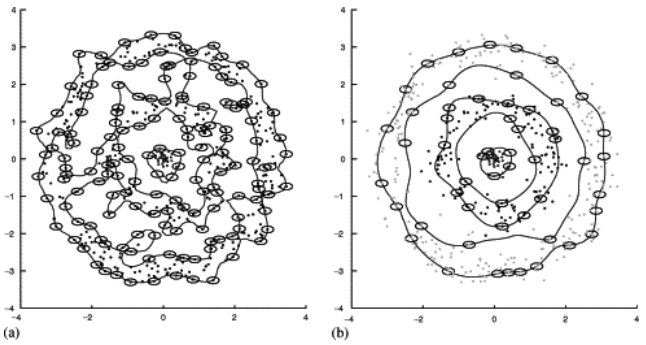

Fig. 5. Clustering with and without BSVs. The inner cluster is composed of 50 points generated from a Gaussian distribution. The two concentric rings contain 150/300 points, generated from a uniform angular distribution and radial Gaussian distribution. (a) The rings cannot be distinguished when C =1.0 Shown here is q=3.5, the lowest q value that leads to separation of the inner cluster. (b) Outliers allow easy clustering. The parameters are p=0.3 and q=1.0.

Figure 4 shows some examples of data points clustering with different q and p without BSVs ($C = 1$). The contour of cluster is blur, while q increases, and is fine while q decreases, but it makes the contour of the cluster affix mutually or break up if q is over-small or over-large. In Figure 5-a without BSVs contour separation does not occur for the two outer rings for any value of q. When some BSVs are present, the clusters are separated easily Figure 5-b. So the two parameters q and C are the identifier of the cluster's accuracy and tightness.

C. Modified SVC

In SVC method, the labeling piece is a bottleneck. iSVC [7] introduced a novel approach, whose idea is to cluster BSVs firstly, then construct a classifier based on labeled BSVs, finally label other data using the classifier. This algorithm is named as **SA** algorithm in some books. The steps are as follows

1) Create affinity matrix H with respect to BSVs where is a $V \times V$ matrix with $H_{i,j} = K(v_i, v_j)$, with v_i and v_j being BSVs.
2) Normalize H, using cholesky decomposition [16], into $H_c = D^{-1/2}HD^{-1/2}$, with $D_{ii} = \sum_j H_{i,j}$.
3) Find $S_1..S_\kappa$, the κ largest eigenvectors (κ is specified by the number of eigenvalues that are larger than 1 [17]) and form Matrix $S_{V\times\kappa} = [S_{1_{V\times 1}}...S_{\kappa_{V\times 1}}]$ then normalize it: $\overline{S_{i,j}} = S^2_{i,j}/(\sum_j S_{i,j})^{1/2}$.
4) Treating each row of $\overline{S}$ as a point in $\mathcal{R}^\kappa$ and cluster it into κ clusters (using k-means [2]).
5) Label v_i as the i^{th} row's cluster membership.
6) Label other data in terms of its nearest BSV's label.

The time complexity bottleneck is occurring in the step **(4)**, where the algorithm clusters the BSVs using k-means. In the next section we introduce a algorithm which modifies the SVC and reduced both time complexity and accuracy of the method.

V. Algorithm

In this section we introduced the algorithm of Grid–SVC. The algorithm has two phases, a preprocess which forms an approximated of clusters, based on a novel grid based clustering, introduced in section 3, then applies modified SVC algorithm for each obtained grid. First we introduce 2 kind of clusters.

Trivial cluster
Cluster ζ_T is a trivial cluster, if there is no way to separate it into clusters ζ_1 and ζ_2, using the algorithm introduced in section 3 (A d–dimensional grid or hybrid grid).

Non-Trivial cluster
Cluster $\zeta_{(q,C)}$ is a non-trivial cluster, if the SVC algorithm by initial parameters q and C, can divide a Trivial cluster ζ_T into clusters $\zeta_{(q,c)_1}$,...,$\zeta_{(q,C)}$,...,$\zeta_{(q,C)_k}$.

In Figure 6 the grids 1 and 2, are the trivial clusters and the contours are non-trivial. In a top-down hierarchical clustering scheme [1], as Figure 7, a trivial cluster can be septated into several non-trivial clusters.
The reduction strategy goes as follows: If a data set of N data points, would be cluster in $O(f(N))$, and the data set contains g, trivial clusters, then using Grid–SVC, we can cluster the data set in $O(f(N/g))$.
For a data set of N data points of d-dimensions, we can develop the algorithm as

(1)form a MAE of d, MAs (Figure 1);
(2)foreach MA, construct the distribution function $F_j(x)$ of Eq.(1) for j^{th} dimension;
(3)using Taylor expansion, obtain $P_j(x)$ Eq.(1);
(4)obtains the intervals which Eq.(2)$\geq$0;
(5)using the cartesian multiply agent, obtain the trivial clusters;

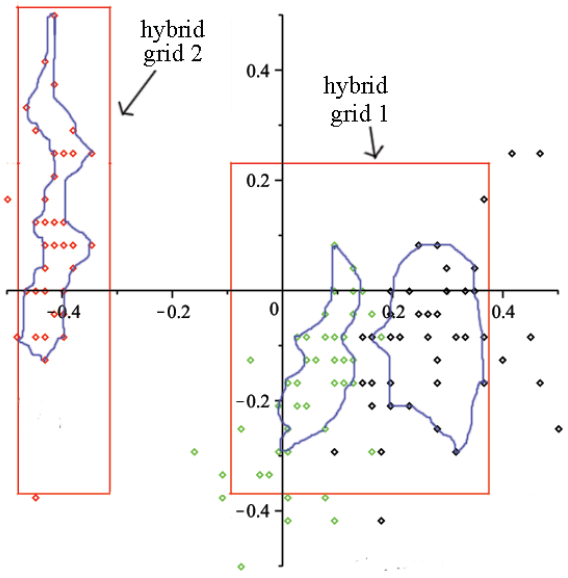

Fig. 6. Clustering of iris data set with three different classes(Setosa,Versicolor,Virginica). Identifying the 2 trivial clusters, using novel Grid–based clustering.

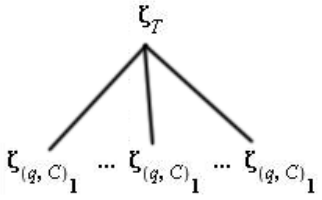

Fig. 7. Trivial cluster ζ_T is separated into clusters $\zeta_{(q,C)_1}...\zeta_{(q,C)_k}$ using SVC method with initial parameters C and q.

(6)using modified SVC obtain the non-trivial (final) clusters;

As mention previously the labeling piece is a bottleneck in SVC. In the step **(6)**, we used the modified SVC with labeling piece, applying **SA** algorithm. As we apply the SVC for each trivial cluster separately, the 4^{th} step in **SA**, changes to g separated k-means, thus the order of **SA** reduced to $O(f(N/g, k))$ where $f(N, k)$ is the time complexity order of k-means and g is the number of trivial clusters.

VI. Experimental results

The software we used to test the result is microsoft.maple.13 [18], with hardware configuration: CPU 1.6 atom dual core, 1 G of ram.
The data set *star* is showed in Figure 8. The data set contains of 300 data points of 2 dimensions. Figure 9 shows the density function of star data set for each dimension. The reason we tested the algorithm on such data set is to show the high performance of the algorithm for the data sets which follow some special patterns. Table I shows the results of Grid–SVC and original SVC for data set *star*. The values of q and C are set separately for the 4 trivial clusters of around the central circle (the first value in Table I) and the cluster in the central position (the second value in Table I). The results shows

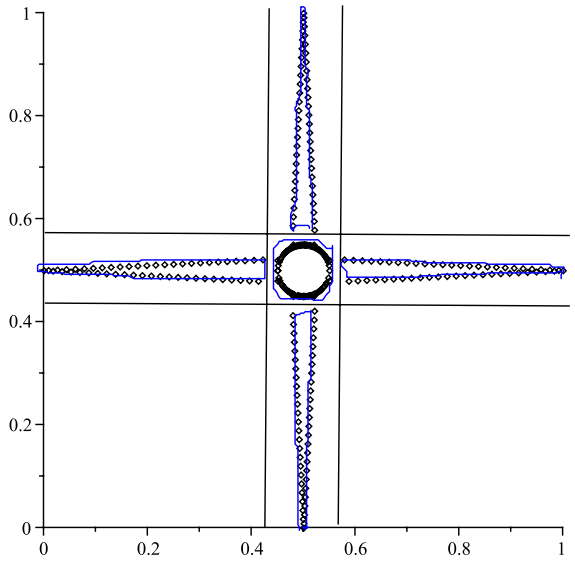

Fig. 8. Clustering the data set star, with 5 trivial and 5 non-trivial clusters.

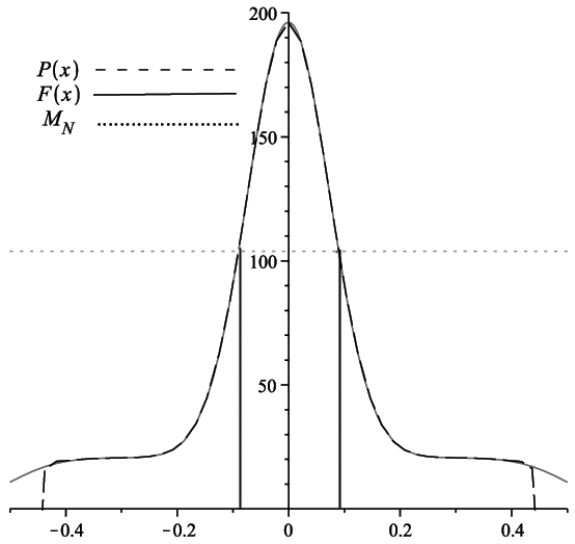

Fig. 9. The density function of data set star for one of its dimensions. In this Experience we set $\epsilon = 100$ and $n = 90$.

improvements both in accuracy and time. SVC misclassifies 2 data points, while the purity of Grid–SVC output clusters is 100%. In this case we used 5 parallel SVC with the labeling algorithm of section 4. The times in second unit for the 4 clusters in around is 9.3, and for the central cluster is 10.4, and 2.4 for the initial process. thus we report the maximum value in Table I). Here, we can eliminate the boundary issue, thus $C = 1.0$ for every grid, while for the original SVC we are forced to consider the BSVs, unless the number of the clusters would be incorrect.

Figure 6 shows the clustering results of normalized iris data set. The iris data set introduced by fisher (1936) [14] and is a standard benchmark in the pattern recognition literature. The data set contains 150 instances each composed of four

TABLE I
COMPARING THE RUNTIME BETWEEN SVC AND G-SVC APPLYING *star* DATA SET. THE UNIT OF TIME IS SECOND. FOR GRID–SVC THERE ARE FIVE TRIVIAL CLUSTERS. WE ASSUME SAME q AND C FOR THE FOUR AROUND CLUSTERS AND ONE FOR CENTRAL CLUSTER. SO WE HAVE TWO VALUES FOR q AND C.

method	q	C	time(sec)	misclassification(s)
SVC	10.0	0.004	28.1	2
Grid–SVC	3.2,2.4	1.0,1.0	12.8	0

TABLE II
COMPARING THE RUNTIME BETWEEN SVC AND GRID–SVC APPLYING IRIS DATA SET. THE UNIT OF TIME IS SECOND. FOR GRID–SVC THERE ARE TWO TRIVIAL CLUSTERS, SO WE HAVE TWO VALUES FOR q AND C.

method	q	C	time(sec)	misclassification(s)
SVC	6.0	0.011	31.3	2
Grid–SVC	4.7,4.2	0.021,1.00	15.1	1

measurements of an iris flower. There are three types of flowers, represented by 50 instances each. One of the clusters is linearly separable from the other two by a clear gap in the probability distribution. The remaining two clusters have significant overlap. There are 2 trivial clusters and 3 non-trivial clusters. To obtain the result, we run the algorithm separately for the two grid by this initial values, q_1, C_1 for the grid number 1, and q_2, C_2 for the grid number 2. In Table II the result are compared with original SVC. The most important issue in Iris is the number of misclassifications. Ben-Hur et al. [19] reported 2 misclassifications, while here we catch more accuracy in final clusters we less time spending (1 misclassification). The time we obtained the final clusters is almost half of the time the original SVC costs. Here we applied 2 MAs for each hybrid grids. the time of initialize process is 2.9 seconds. Same as *star*, we eliminate the boundary issue for the hybrid grid number 2 (C=1.0). For the hybrid grid number 1, we assume the BSVs, as (C=0.021). For parameter q, we decreased the value, in compare with original SVC. The reasons are first we have less points to cluster in each hybrid grid, and second, the accuracy issue is relaxed by eliminating the outlier points.

VII. CONCLUSION

In this paper we introduce a novel algorithm for clustering high dimensional data sets. The algorithm has two phases. At first, using d parallel programs, applies a density function for each dimension based on RBF, then using Taylor approximation, the algorithm obtain a polynomial form of the density function. based on that it identifies the intervals which contain the most data points. Using cartesian multiply, the algorithm finds the grid (trivial clusters) which are the most aggregated areas. Then the algorithm applied the SVC algorithm with a modification on labeling piece and again in parallel way find the final (non-trivial clusters). The experimental results show a fine time reduction and also higher accuracy for the real data sets.

VIII. ACKNOWLEDGEMENT

With Special thanks to Dr. leila Sharif who help us through writing this paper.Paper ID: G202 Paper Password: p2024139

TABLE III
SOME WELL–KNOWN FUNCTIONS THAT GENERATE RBFS
$(r = \|x - x_i\| = r_i),\ \epsilon > 0$

Name of functions	Definition
Inverse quadric(IQ)	$1/((r/\epsilon)^2 + 1)$
Multiquadrics (MQ)	$\sqrt{(r/\epsilon)^2 + 1}$
Inverse multiquadrics (IMQ)	$1/(\sqrt{(r/\epsilon)^2 + 1})$
Gaussian (GA)	$exp(-\epsilon^2 r^2)$

APPENDIX

Radial basis functions: Let $\mathbb{R}^+ = \{x \in \mathbb{R}, x \geq 0\}$ be the non-negative half-line and let $\phi : \mathbb{R}^+ \to \mathbb{R}$ be a continuous function with $\phi(0) \geq 0$. A radial basis function on $\mathbb{R}^d$ is a function of the form

$$\phi(\|X - X_i\|), \tag{12}$$

where $X,\ X_i \in \mathbb{R}^d$ and $\|.\|$ denotes the Euclidean distance between $X,\ X_i$. If one chooses N points $\{X_i\}_{i=1}^N$ in $\mathbb{R}$ then by custom

$$S(X) = \sum_{i=1}^{N} \lambda_i \phi(\|X - X_i\|); \quad X = (x_1 ... x_d) \quad \lambda_i \in \mathbb{R},$$

is called a radial basis function as well. Some of the infinitely smooth RBF choices listed in TabIII.

The standard radial basis functions are categorized into two major classes [20]:

Class 1. Infinitely smooth RBFs [20], [21]:

These basis functions are infinitely differentiable and heavily depend on the shape parameter ϵ e.g. Hardy multiquadric (MQ), Gaussian(GA), inverse multiquadric (IMQ), and inverse quadric(IQ)(See Table III).

Class 2. Infinitely smooth (except at centers) RBFs [20], [21]:

The basis functions of this category are not infinitely differentiable. These basis functions are shape parameter free and have comparatively less accuracy than the basis functions discussed in the Class 1. For example, thin plate spline, etc [20].

All of the radial basis functions have global support, and in fact many of them, such as multiquadrics (MQ), do not even have isolated zeros [22], [23]. The RBFs can be compactly and globally supported, infinitely differentiable, and contain a free parameter ϵ, called the shape parameter [23], [24], [25]. For more basic details about RBFs the interested readers can refer to the recent books and paper by Buhmann [24], [10] and Wendland [26], compactly and globally supported; and convergence rate of the radial basis functions. Too large or too small shape parameter ϵ make the RBFs too flat and too peaked.

In this paper RBFs were applied as density function by choosing good parameter ϵ. These functions allow scattered data in d-dimensions to be easily used in computations.

REFERENCES

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2006, ch. 7, pp. 308–466.
[2] J. A. Hartigan and M. A. Wong, "A K-Means Clustering Algorithm," *J. ROY. STAT. SOC. C-APP.*, pp. 100–108, 1979.
[3] M. J. Brusco and H. F. Köhn, "Optimal partitioning of a data set based on the P-median model," *Psychometrika.*, vol. 73, pp. 89–105, Murch 2008.
[4] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proceedings of Second International Conference on Knowledge Discovery and Data Mining, Portland*, pp. 226–231, 1996.
[5] D. Horn, A. BenHur, H. Siegelmann, and V. Vapnik, "A support vector method for clustering," in *Advances in Neural Information ProcessingSystems*, ser. Proceedings of the 2000 Conference, T. Leen, T. Dietterich, and V. Tresp, Eds., vol. 13. MIT Press, 2001, pp. 367–373.
[6] D. Horn, "Clustering via hilbert space," *Physica A*, vol. 302, pp. 70–79, 2001.
[7] L. Ping, Z. Chun-Guang, and Z. Xu, "Improved support vector clustering," *ENG. APPL. ARTIF. INTEL.*, vol. 23, pp. 552–559, 2010.
[8] Y. Sun, Y. Wang, J. Wang, W. Du, and C. Zhou, "A novel svc method based on k-means," in *Second International Conference on Future Generation Communication and Networking*, 2008, pp. 55–58.
[9] J. Lee and D. Lee, "An improved cluster labeling method for support vector clustering," *IEEE. T. PATTERN. ANAL.*, pp. 461–464, 2005.
[10] M. D. Buhmann, *Radial Basis Functions: Theory and Implementations*. New York: Cambridge University Press, 2004.
[11] C. Aggarwal, C. Procopiuc, J. Wolf, P. S. Yu, and J. S. Park, "Fast algorithms for projected clustering," in *In Proc. 1999 ACM-SIGMOD, Int. Conf. Management of Data (SIGMOD99)*, philadelphia, PA, June 1999, pp. 61–72.
[12] R. Aggrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in *Management of Data (SIGMOD98)*.
[13] S. Russel and P. Norving, *Artificial Intelligence, A modern approach (pages 32-57)*, 2nd ed. Prentice Hall, 2003.
[14] R. A. Fisher, "The use of multiple measurments in taxonomic problems," *Annals of Eugenics*, pp. 179–188, 1936.
[15] R. Fletcher, *Practical Methods of Optimization*. Chichester: Wiley-Interscience, 1987.
[16] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*. New York: Dover Publications, 1972.
[17] T. Zheng, L. Xiaobin, and J. Yanwei, "Disturbing analysis on spectrum clustering," *Science in China(SeriesE)*, vol. 37, no. 4, pp. 527–543, 2007.
[18] www.maplesoft.com.
[19] A. BenHur, D. Horn, H. Siegelmann, and V. Vapnik, "Support vector clustering," *Machine Learning Research*, vol. 2, pp. 125–137, 2001.
[20] A. J. Khattak, S. I. A.Tirmizi, and S. U. Islam, "Application of meshfree collocation method to a class of nonlinear partial differential equations," *ENG. ANAL. BOUND. ELEM.*, vol. 33, pp. 661–667, 2009.
[21] M. Dehghan and A. Shokri, "A meshless method for numerical solution of the one-dimensional wave equation with an integral condition using radial basis functions," *Numer. Algorithms.*, vol. 52, pp. 461–477, 2009.
[22] ——, "Numerical solution of the nonlinear Klein-Gordon equation using radial basis functions," *J. Comput. Appl. Math.*, vol. 230, pp. 400–410, 2009.
[23] S. U. Islam, S. Haqb, and A. Ali, "A meshfree method for the numerical solution of the RLW equation," *J. Comput. Appl. Math.*, vol. 223, pp. 997–1012, 2009.
[24] M. D. Buhmann, "Radial basis functions," *Acta Numerica*, vol. 16, pp. 1–38, 2000.
[25] S. Sarra, "Adaptive radial basis function method for time dependent partial differential equations," *Appl. Numer. Math.*, vol. 54, pp. 79–94, 2005.
[26] H. Wendland, *Scattered Data Approximation*. New York: Cambridge University Press, 2005.