Discovering Local Outliers using Dynamic Minimum Spanning Tree with Self-Detection of Best Number of Clusters

S. John peter
Department of Computer Science and Research Center
St. Xavier's College, Palayamkottai
Tamil Nadu, India.

ABSTRACT
Detecting outliers in database (as unusual objects) using Clustering and Distance-based approach is a big desire. Minimum spanning tree based clustering algorithm is capable of detecting clusters with irregular boundaries. In this paper we propose a new algorithm to detect outliers based on minimum spanning tree clustering and distance-based approach. Outlier detection is an extremely important task in a wide variety of application. The algorithm partition the dataset into optimal number of clusters. Small clusters are then determined and considered as outliers. The rest of the outliers (if any) are then detected in the clusters using Distance-based method. The algorithm uses a new cluster validation criterion based on the geometric property of data partition of the dataset in order to find the proper number of clusters. The algorithm works in two phases. The first phase of the algorithm creates optimal number of clusters, where as the second phase of the algorithm detect outliers in the clusters. The key feature of our approach is it combines the best features of Distance-based and Clustering-based outlier detection to find noise-free/error-free clusters for a given dataset without using any input parameters.

General Terms: Graph Based Algorithm; Information retrieval;

Keywords: Euclidean minimum spanning tree, Subtree, Clustering, Eccentricity, Cluster validity, Cluster Separation, Small Clusters, Outliers

1. INTRODUCTION

An outlier is an observation of data that deviates from other observations so much that it arouses suspicious that was generated by a different mechanism from the most part of data [15]. Outlier may be erroneous or real in the following sense. Real outliers are observations whose actual values are very different than those observed for rest of the data and violate plausible relationship among variables.

Erroneous outliers are observations that are distorted due to misreporting or misrecording errors in the data collection process. Outliers of either type may exert undue influence on the result of data analysis. So they should be identified using reliable detection methods prior to performing data analysis [15].

Outliers can often be individual or groups of clients exhibiting behavior outside the range of what is considered normal. Outliers can be removed or considered separately in *regression modeling* to improve accuracy which can be considered as benefit of outliers. Identifying them prior to modeling and analysis is important [44].
The regression modeling consists in finding a dependence of one random variable or a group of variables on another variables or a group of variables. All most all studies that consider outlier identification as their primary objective are in statistics. The test depends on distribution; whether or not the distribution parameters are known; the number of expected outliers; the type of expected outliers.

The importance of outlier detection is due to the fact that outliers in the data translate to significant (and often critical) information in a wide variety of application domains. For example, an anomalous traffic pattern in a computer network could mean that a hacked computer is sending out sensitive data to an unauthorized destination. In public health data, outlier detection techniques are widely used to detect anomalous pattern in patient medical records which could be symptoms of new diseases. Similarly, outliers in credit card transaction data could indicate credit card theft or misuse. Outliers can also translate to critical entities such as in military surveillance, whereas the presence of unusual region in a satellite image of enemy are could indicate enemy troop movement. Or anomalous readings from space craft would signify a fault in some of the craft. Outlier detection has been found to be directly applicable in large number of domains.

Many data-mining algorithms find outliers as a side-product of clustering algorithms. However these techniques define outlier as points, which do not lie in clusters. Thus, the techniques implicitly define outliers as the background noise in which the clusters are embedded. Another class of techniques defines outlier as points, which are neither a part of a cluster nor part of background noise; rather they are specifically points which behave very differently from the norm [1].

Given a distance measure on a feature space, there are many definitions for the distance-based outliers. Knorr and Ng [24] define the outliers as a point *P* in a data set is outlier with respect to parameters *k and d*, if at least *k* points in the data set lie greater than distance *d* from *P*.

Ramasamy *et al* [37], proposes a new definition for distance based outliers, based on the point *P*, to its *k* th nearest neighbor, denoted with $D^k(P)$. Given a k and n, a point P is outlier if no more than n-1 other points in the data set have a higher value for D^k than *P*. This means that the top n points, having the maximum D^k values, are considered as outliers.

Angiulli and Pizzuti [4] proposed a new definition for outliers. In this definition, for each point, *P*, the sum of the distances from its *k* nearest neighbor's considered. This sum is called

the weight of *P*, W_k *(P)* and is used to rank the points of data set. Outliers are those points having the largest values of W_k.

Distance based approaches are simple to implement. However, they suffer exponential computational growth as they are founded on the calculation of the distances between all objects in the data set. The computational complexity is directly proportional to both the dimensionality of the data and the number of objects. Hence, the technique for calculating the distance with a lower runtime is required.

The outlier detection problem in some cases is similar to the classification problem. Clustering is a popular technique used to group similar data points or objects in groups or clusters [3]. Clustering is an important tool for outlier analysis. Several clustering–based outlier deduction techniques have been developed. Most of these techniques rely on the key assumption that normal objects belong to large and dense clusters, while outliers form very small clusters [28, 32]. The main concern of *clustering–based* outlier detection algorithms is to find clusters and outliers, which are often regarded as noise that should be removed in order to make more reliable clustering [18]. Some noisy points may be far away from the data points, whereas the others may be close. The far away noisy points would affect the result more significantly because they are more different from the data points. It is desirable to identify and remove the outliers, which are far away from all the other points in cluster [20]. So, to improve the clustering such algorithm use the same process and functionality to solve both clustering and outliers discovery [10].

The problem of determining the correct number of clusters in a data set is perhaps the most difficult and ambiguous part of cluster analysis. Hardy [16] recommends that the determination of optimal number of clusters should be made by using several different clustering methods that together produce more information about the data. By forcing a structure to a data set, the important and surprising facts about the data will likely remain uncovered.

In some applications the number of clusters is not a problem, because it is predetermined by the context [17]. Then the goal is to obtain a mechanical partition for a particular data using a fixed number of clusters. Such a process is not intended for inspecting new and unexpected facts arising from the data. Hence, splitting up a homogeneous data set in a "fair" way is much more straightforward problem when compared to the analysis of hidden structures from heterogeneous data set. The clustering algorithms [21, 33] partitioning the data set in to *k* clusters without knowing the homogeneity of groups. Hence the principal goal of these clustering problems is not to uncover novel or interesting facts about data.

Given a connected, undirected graph $G = (V, E)$, where V is the set of nodes, E is the set of edges between pairs of nodes, and a weight $w(u, v)$ specifying weight of the edge (u, v) for each edge $(u, v) \in E$. A spanning tree is an acyclic subgraph of a graph *G*, which contains all vertices from *G*. The Minimum Spanning Tree (**MST**) of a weighted graph is minimum weight spanning tree of that graph. Several well established **MST** algorithms exist to solve minimum spanning tree problem [36, 26, 31]. The cost of constructing a minimum spanning tree is $O(m \log n)$, where *m* is the number of edges in the graph and *n* is the number of vertices. More efficient algorithm for constructing **MST**s have also been extensively researched [22, 13, 19]. These algorithms promise close to linear time complexity under different assumptions. A Euclidean minimum spanning tree (**EMST**) is a spanning tree of a set of *n* points in a metric space ($\mathbf{E^n}$), where the length of an edge is the Euclidean distance between a pair of points in the point set.

Clustering algorithms using minimal spanning tree takes the advantage of **MST**. The **MST** ignores many possible connections between the data patterns, so the cost of clustering can be decreased. The **MST** based clustering algorithm is known to be capable of detecting clusters with various shapes and size [47]. Unlike traditional clustering algorithms, the **MST** clustering algorithm does not assume a spherical shapes structure of the underlying data. The **EMST** clustering algorithm [35, 47] uses the Euclidean minimum spanning tree of a graph to produce the structure of point clusters in the *n*-dimensional Euclidean space. Clusters are detected to achieve some measures of optimality, such as minimum intra-cluster distance or maximum inter-cluster distance [5]. The **EMST** algorithm has been widely used in practice.

Clustering by minimal spanning tree can be viewed as a hierarchical clustering algorithm which follows a divisive approach. Using this method firstly **MST** is constructed for a given input. There are different methods to produce group of clusters. If the number of clusters *k* is given in advance, the simplest way to obtain *k* clusters is to sort the edges of minimum spanning tree in descending order of their weights and remove edges with first *k*-1 heaviest weights [5, 45].

Geometric notion of centrality are closely linked to facility location problem. The distance matrix *D* can computed rather efficiently using Dijkstra's algorithm with time complexity $O(|V|^2 \ln |V|)$ [41].

The *eccentricity* of a vertex *x* in *G* and radius ρ (G), respectively are defined as

$$e(x) = \max_{y \in V} d(x, y) \quad \text{and} \quad \rho(G) = \min_{x \in V} e(x)$$

The *center* of *G* is the set

$$C(G) = \{x \in V \mid e(x) = \rho(G)\}$$

C (G) is the center to the "*emergency facility location problem*" which is always contain single block of *G*. The length of the longest path in the graph is called *diameter* of the graph *G*. we can define diameter D (G) as

$$D(G) = \max_{x \in V} e(x)$$

The *diameter* set of *G* is

$$Dia(G) = \{x \in V \mid e(x) = D(G)\}$$

In this paper, we consider outliers as points, which are far from the most of other data. The proposed approach is first partitioned the dataset into optimal number of clusters and then find outliers from the resulting clusters using distance – based approach. Our approach will detect outliers from clusters with less computational complexity.

Our **DGMSTLOD** algorithm is based on Minimum Spanning Tree does not require a predefined cluster number. The algorithm constructs an **EMST** of a point set and removes the inconsistent edges that satisfy the inconsistence measure. The process is repeated to create a hierarchy of clusters until optimal numbers of clusters (regions) are obtained. Using the

optimal number of clusters outliers can be easily detected. In section 2 we review some of the existing works on distance-based outlier detection, clustering-based outlier detection, cluster validity, and minimum Spanning tree based clustering algorithms. In Section 3 we propose **DGMSTLOD** algorithm which produces optimal number of clusters with outliers. Finally in conclusion we summarize the strength of our methods and possible improvements.

2. RELATED WORK

There is no single universally applicable or generic outlier detection approach [28, 32]. Therefore there is many approaches have been proposed to deduct outliers. These approaches are classified into four major categories as distribution-based, distance-based, density-based and clustering-based [48].

In the distance-based approach [23,24,25,37], outliers are detected using a given distance measure on feature space, A point *q* in a data set is an outlier with respect to the parameters *M* and *d*, if there are less than *M* points with in the distance *d* from *q*, where the values of *M* and *d* are determined by the user. The problem in distance–based approach is that it is difficult to determine the *M* and *d* values. Angiulli [4] propose a new definition of outliers. In this definition, for each point, *P*, the sum of the distances from its *k* nearest neighbor's considered. This sum is called the weight of *P*, W_k *(P)* and is used to rank the points of data set. Outliers are those points having the largest values of W_k. The method proposed by Angiulli [4] needs expected number of outlier *n* and application dependent *k* as input parameter. It is difficult to predict correct values for *n* and *k*. The problem with distance based approach is its high computational complexity. Bay and Schwabcher [7] present an algorithm which is based on the nearest-loop algorithm, using randomization and pruning rule, with linear time performance. However, the algorithm depends on the data ordering which can lead to poor performance.

Clustering-based approaches [28, 14, 18, 20], consider clusters of small sizes as outliers. In these approaches, small clusters (clusters containing significantly less points than other clusters) are considered as outliers. The advantage of clustering based approaches is that they do not have to be supervised.

Jiang et. al. [20] proposed a two-phase method to detect outliers. In the first phase, clusters are produced using modified K-means algorithm, and then in the second phase, an Outlier-Finding Process **(OFP)** is proposed. The small clusters are selected and regarded as outliers. Small cluster is defined as a cluster with fewer points than half the average number of points in the *k* number of clusters. Loureio [28] proposed a method for detecting outlier. Hierarchical clustering technique is used for detecting outliers. The key idea is to use the size of the resulting clusters as indicators of the presence of outliers. Almedia [2] is also used similar approach for detecting outliers. Using the K-means clustering algorithm Yoon [46] proposed a method to detect outliers. The K-means algorithm is sensitive to outliers, and hence it may not give accurate result.

Moh'd Belal Al-Zoubi [29] proposed a method based on clustering approaches for outlier detection using Partitioning Around Medoid (**PAM**). **PAM** attempts to determine *k* partition for *n* objects. The algorithm uses most centrally located object in a cluster (called medoid) instead of cluter mean. **PAM** is more robust than the k-means algorithm in the presence of noise and outlier. This **PAM** based approach suffer from proper cluster Structure. Cluster with irregular boundaries can not be detected using both k-means and **PAM** algorithms.

The Partial Distance (**PD**) algorithm [8, 15, 43] has been proposed to reduce computation complexity of **LGB** algorithm of [27, 34] with in the area of Vector Quantization (**VQ**). The **PD** method first calculates the distance between a query point, *P* and arbitrary data point and takes this distance as the current initial minimum distance. Then it continuously campers the accumulative partial distance between the query point and each candidate data point with the current minimum distance. If the accumulative partial distance exceeds the current minimum distance, the candidate data point is eliminated (rejected) before completing the total distance calculation. If a total distance is obtained, then the current minimum distance is updated by choosing the minimum of the current minimum distance.

The performance of the **PD** algorithm is sensitive to the choice of the initial minimum distance, d_{mim}
[9], this will degrade the performance of the **PD** algorithm. Instead of choosing an arbitrary data point, which is the case in the **PD** algorithm, one might think of choosing the mean value of the data set. However, choosing the mean value might lead to wrong result in cases where the mean value is the closest to nearest neighbor to a given query point. This is because the mean value might not be one of the points in the data set. The Improved partial Distance (**IPD)** approach [11] based on finding the data point nearest to the mean value $\mathbf{N_{mean}}$ and then computing the distance $\mathbf{d_{Nmean}}$ between each data point and the $\mathbf{N_{mean}}$. The resulting distance is used as initial minimum distance. This approach has improved performance over the **PD** algorithm.

Clustering Algorithm based on minimum and maximum spanning tree were extensively studied. Avis [6] found an $O(n^2 \log^2 n)$ algorithm for the min-max diameter-2 clustering problem. Asano, Bhattacharya, Keil and Yao [5] later gave optimal $O(n \log n)$ algorithm using maximum spanning trees for minimizing the maximum diameter of a bipartition. Asano, Bhattacharya, Keil and Yao also considered the clustering problem in which the goal to maximize the minimum inter-cluster distance. They gave a *k*-partition of point set removing the *k*-1 longest edges from the minimum spanning tree constructed from that point set [5]. The identification of inconsistent edges causes problem in the **MST** clustering algorithm. There exist numerous ways to divide clusters successively, but there is not suitable a suitable choice for all cases.

Zahn [47] proposes to construct **MST** of point set and delete inconsistent edges – the edges, whose weights are significantly larger than the average weight of the nearby edges in the tree. Zahn's inconsistent measure is defined as follows. Let *e* denote an edge in the **MST** of the point set, v_1 and v_2 be the end nodes of *e*, *w* be the weight of *e*. A *depth neighborhood* ***N*** of an end node *v* of an edge *e* defined as a set of all edges that belong to all the path of length *d* originating from *v*, excluding the path that include the edge *e*. Let N_1 and N_2 be the depth *d* neighborhood of the node v_1 and v_2. Let $\hat{W}_{N1}$ be the average weight of edges in N_1 and σN_1 be its standard deviation. Similarly, let $\hat{W}_{N2}$ be the average weight of edges in N_2 and σN_2 be its standard deviation. The inconsistency measure requires one of the three conditions hold:

1. $w > \hat{W}N_1 + c \times \sigma N_1$ or $w > \hat{W}N_2 + c \times \sigma N_2$

2. $w > max(\hat{W}N_1 + c \times \sigma N_1 , \hat{W}N_2 + c \times \sigma N_2)$

3. $$\frac{w}{max\,(c \times \sigma N_1 , c \times \sigma N_2)} > f$$

where *c* and *f* are preset constants. All the edges of a tree that satisfy the inconsistency measure are considered inconsistent and are removed from the tree. This result in set of disjoint subtrees each represents a separate cluster.

The **MST** clustering algorithm has been widely used in practice. Xu (Ying), Olman and Xu (Dong) [45] use MST as multidimensional gene expression data. They point out that **MST**-based clustering algorithm does not assume that data points are grouped around centers or separated by regular geometric curve. Thus the shape of the cluster boundary has little impact on the performance of the algorithm. They described three objective functions and the corresponding cluster algorithm for computing *k*-partition of spanning tree for predefined $k > 0$. The algorithm simply removes *k*-1 longest edges so that the weight of the subtrees is minimized. The second objective function is defined to minimize the total distance between the center and each data point in the cluster. The algorithm removes first *k*-1 edges from the tree, which creates a *k*-partitions.

The selection of the correct number of clusters is actually a kind of validation problem. A large number of clusters provides a more complex "model" where as a small number may approximate data too much. Hence, several methods and indices have been developed for the problem of cluster validation and selection of the number of clusters [39, 16, 38, 40, 42]. Many of them based on the within and between-group distance.

3. ALGORITHM FOR LOCAL OUTLIERS

A tree is a simple structure for representing binary relationship, and any connected components of tree is called *subtree*. Through this **MST** representation, we can convert a multi-dimensional clustering problem to a tree partitioning problem, ie., finding particular set of tree edges and then cutting them. Representing a set of multi-dimensional data points as simple tree structure will clearly lose some of the inter data relationship. However many clustering algorithm proved that no essential information is lost for the purpose of clustering. This is achieved through rigorous proof that each cluster corresponds to one subtree, which does not overlap the representing subtree of any other cluster. Clustering problem is equivalent to a problem of identifying these subtrees through solving a tree partitioning problem. In this section we present distance based clustering algorithm which produce optimal number of clusters with outliers.

3.1. DGMSTLOD ALGORITHM

Given a point set *S* in $\mathbf{E}^n$, the hierarchical method starts by constructing a Minimum Spanning Tree (**MST)** from the points in *S*. The weight of the edge in the tree is Euclidean distance between the two end points. So we named this **MST** as **EMST1.** Next the average weight $\hat{W}$ of the edges in the entire **EMST1** and its standard deviation σ are computed; any edge with $W > \hat{W} + \sigma$ or *current longest edge* is removed from the tree. This leads to a set of disjoint subtrees $S_T = \{T_1, T_2 \ldots\}$. Each of these subtrees T_i is treated as cluster having *X* data points of size N. We propose a new algorithm named, *Dynamically Growing Minimum Spanning Tree Clustering for Local Outlier Detection* algorithm *(***DGMSTLOD**), which does not require a predefined cluster number. The algorithm works in two phases. The first phase of the algorithm partitioned the **EMST1** into sub trees (clusters/regions). The centers of clusters or regions are identified using eccentricity of points. These points are a representative point for the each subtree S_T. A point c_i is assigned to a cluster *i* if $c_i \in T_i$. The group of center points is represented as $C = \{c_1, c_2 \ldots\ldots c_k\}$. These center points $c_1, c_2 \ldots c_k$ are connected and again minimum spanning tree **EMST2** is constructed is shown in the Figure 4. From the **EMST2** optimal number of clusters is generated. Based on the definition of *small clusters* as defined in [28], we define *small cluster* as *a cluster with fewer points than half the average number of points in the optimal number of clusters.* We first detect small clusters (outliers) from optimal number of clusters. To detect the outliers from the rest of the clusters (if any), we use the fast distance-based approach [30].

Given a query point, *q*, circle centered at *q* is drawn with radius *d*, and then we count the number of data points in side the circle (ICcount). If the count is less than *M*, then the point is considered as outliers, otherwise, the point is normal (not outliers). If ICcount is greater than *M*, then *q* is not an outlier, then there is no need to test the rest of the data points in the cluster. Otherwise we count the number of points outside the circle (OCcount). We store these data points in an array for further processing. The value of ICcount and OCcount are added. If the sum is less than *M*, then *q* is an outlier. Otherwise, we perform distance calculations for the points that are stored in the array only [30].

Here, we use a cluster validation criterion based on the geometric characteristics of the clusters, in which only the inter-cluster metric is used. The **DGMSTLOD** algorithm is a nearest centroid-based clustering algorithm, which creates region or subtrees (clusters/regions) of the data space. The algorithm partitions a set *S* of data in to *X* data points of size *N*, in data space in to *n* regions (clusters). Each region is represented by a centroid reference vector. If we let *c* be the centroid representing a region (cluster), all data within the region (cluster) are closer to the centroid *c* of the region than to any other centroid *q*:

$$R(c) = \{x \in X \mid dist(x, c) \leq dist(x, q) \ \forall q\} \quad (2)$$

Thus, the problem of finding the proper number of clusters of a dataset can be transformed into problem of finding the proper region (clusters) of the dataset. Here, we use the **MST** as a criterion to test the inter-cluster property. Based on this observation, we use a cluster validation criterion, called Cluster Separation (CS) in **DGMSTLOD** algorithm [12].

Cluster separation (CS) is defined as the ratio between minimum and maximum edge of MST. ie

$$CS = E_{min} / E_{max} \quad (3)$$

where *Emax* is the maximum length edge of **MST**, which represents two centroids that are at maximum separation, and E_{min} is the minimum length edge in the **MST**, which represents two centroids that are nearest to each other. Then, the *CS* represents the relative separation of centroids. The value of *CS* ranges from 0 to 1. A low value of *CS* means that

the two centroids are too close to each other and the corresponding partition is not valid. A high *CS* value means the partitions of the data is even and valid. In practice, we predefine a threshold to test the *CS*. If the *CS* is greater than the threshold, the partition of the dataset is valid. Then again partitions the data set by creating subtree (cluster/region). This process continues until the *CS* is smaller than the threshold. At that point, the proper number of clusters will be the number of cluster minus one. The *CS* criterion finds the proper binary relationship among clusters in the data space. The value setting of the threshold for the *CS* will be practical and is dependent on the dataset. The higher the value of the threshold the smaller the number of clusters would be. Generally, the value of the threshold will be > 0.8[12].

Figure 3 shows the *CS* value versus the number of clusters in hierarchical clustering. The *CS* value < 0.8 when the number of clusters is 5. Thus, the proper number of clusters for the data set is 4. Further more, the computational cost of *CS* is much lighter because the number of subclusters is small. This makes the *CS* criterion practical for the **DGMSTLOD** algorithm when it is used for clustering large dataset to detect outliers. Our approach combines the best features of distance-based outlier detection and Clustering based outlier detection to detect outliers more effectively.

Algorithm: DGMSTLOD ( )
Input : *S* the point set and query point *q, d, M*
Output : optimal number of clusters with *O*
the set of outliers

Let *e* be an edge in the **EMST1** constructed from *S*
Let W_e be the weight of *e*
Let σ be the standard deviation of the edge weights in **EMST1**
Let S_T be the set of disjoint subtrees of **EMST1**
Let n_c be the number of clusters
Let *O* be set of outliers
Let *NewArray* be an array of type *X*

1. Construct an **EMST1** from *S*
2. Compute the average weight of Ŵ of all the Edges from **EMST1**
3. Compute standard deviation σ of the edges from **EMST1**
4. $S_T = ø$; $n_c = 1$; $C = ø$; $O = ø$;
5. **Repeat**
6. **For** each $e \in$ **EMST1**
7. **If** $(W_e > \hat{W} + \sigma)$ or (current longest edge *e1)*
8. Remove *e* from **EMST1**
9. $S_T = S_T \cup \{ T' \}$ // *T*'' is new disjoint Subtree (regions)
10. $n_c = n_c + 1$
11. Compute the center C_i of T_i using eccentricity of points
12. $C = U_{Ti} \in S_T \{C_i\}$
13. Construct an **EMST2** *T* from *C*
14. E_{min} = get-min-edge (*T*)
15. E_{max} = get-max-edge (*T*)
16. $CS = E_{min} / E_{max}$
17. **Until** $CS < 0.8$
18. **For** each T_i (cluster having *X* data points) do
19. Détermine *Small clusters* as outliers
$O = O \cup \{small\ clustesrs\}$
20. **For** each T_i (cluster having *X* data points) do
21. **For** each query point *q* do
22. *ICcount = 0.*
23. *OCcount = 0*
24. **For** *m* = 1 to N // for each point x_m
25. **If** *x* is inside the circle **then** *ICcount = ICcount + 1*
26. **If** *ICcount* > *M* **Next** *m* (*q* is not outlier)
27. **Else**
28. **If** *x* is outside the circle **then**
29. *OCcount = OCcount + 1*
30. **Store** *x* in *NewArray*
31. **Next** *m*
32. **If** *(ICcount + OCcount)* < *M* **then** $O = O \cup \{q\}$
33. **Else**
34. **For** each point in *NewArray, do*
35. **If** *dist (q, x)* < = *d* **then** *ICcount=ICcount + 1*
36. **If** *ICcount* < *M* **then**
37. $O = O \cup \{q\}$
38. Return optimal number of clusters with *O*

Figure 1 shows a typical example of **EMST1** constructed from point set *S*, in which inconsistent edges are removed to create subtree (clusters/regions). Our algorithm finds the center of the each cluster, which will be useful in many applications. Our algorithm will find optimal number of clusters or cluster structures. Figure 2 shows the possible distribution of the points in the two cluster structures with their center vertex as 5 and 3.

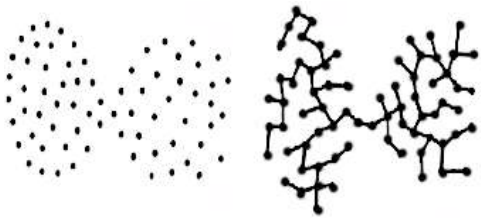

Figure 1: Clusters connected through points -EMST1

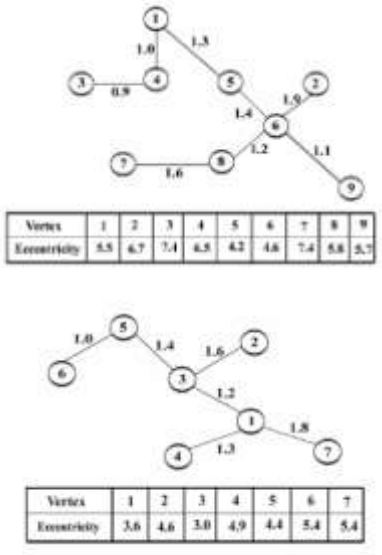

Figure 2: Two Clusters/regions with Center points 5 and 3

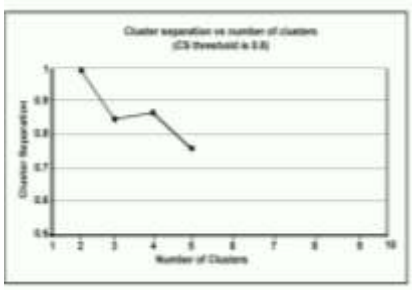

Figure 3: Number of Clusters vs. Cluster Separation

Our **DGMSTLOD** algorithm works in two phases. The outcome of the first phase (lines 1-17) of the algorithm consists of optimal number clusters with their center. It first constructs **EMST1** from set of point S (line 1). Average weight of edges and standard deviation are computed (lines 2-3). Inconsistent edges are identified and removed from **EMST1** to generate subtree *T'* (lines 7-9). The center for each subtree (cluster/region) is computed at line 11. Using the cluster/region center point again another minimum spanning tree **EMST2** is constructed (line 13). Using the new evaluation criteria, optimal number of clusters/regions is identified (lines 14-16). Lines 6-16 in the algorithm are repeated until optimal number of clusters are obtained. The clusters are well separated, shown in Figure 4.
We use the graph of Figure 4 as example to illustrate the second phase (lines 18-38) of the algorithm. The second phase of the **DGMSTLOD** algorithm finds outliers in the optimal number of clusters, which are generated from first phase of the algorithm. It finds small clusters. These small clusters are identified as outliers (line 19). If any outliers present in the cluster are identified using fast Distance-based approach (line 20-38) is shown in Figure 5.

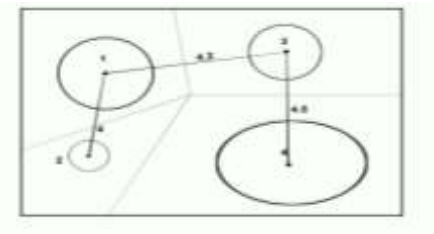

Figure 4**.** EMST2 From 4 region/cluster center points

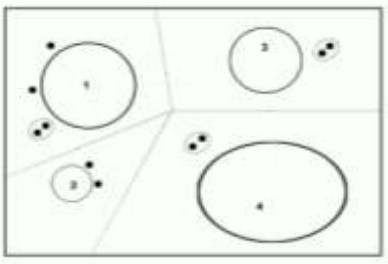

Figure 5: Four Clusters with outliers as black spots

4. CONCLUSION

Our **DGMSTLOD** clustering algorithm finds outliers without using any predefined cluster number as input parameter. The algorithm gradually finds clusters with center for each cluster. Our algorithm does not require the users to select and try various parameters combinations in order to get the desired output. Our **DGMSTLOD** clustering algorithm uses a new cluster validation criterion based on the geometric property of partitioned regions/clusters to produce optimal number of "true" clusters with outliers for each of them. All of these look nice from theoretical point of view. However from practical point of view, there is still some room for improvement for running time of the algorithm. This could perhaps be accomplished by using some appropriate data structure. In the future we will explore and test our proposed clustering algorithm in various domains. We will further study the rich properties of EMST-based clustering methods in solving different clustering problems for detecting outliers in dynamic dataset.

REFERENCES

[1] C. Aggarwal and P. Yu, "Outlier Detection for High Dimensional Data", *In Proceedings of the ACM SIGMOD International Conference on Management of Data,* Volume 30, Issue 2,pages 37 – 46, May 2001.

[2] J.Almeida, L.Barbosa, A.Pais and S.Formosinho, "Improving Hierarchical Cluster Analysis: A new method with OutlierDetection and Automatic Clustering", *Chemometrics and Intelligent Laboratory Systems* 87:208-217, 2007.

[3] Anil K. Jain, Richard C. Dubes "Algorithm for Clustering Data", *Michigan State University, Prentice Hall, Englewood Cliffs, New Jersey* 07632.1988.

[4] F.Angiulli, and C.Pizzuti, "Outlier Mining in Large High-Dimensional Data sets", *IEEETransactions on Kaoeledge and Data Engineering,* 17(2): 203-215, 2005

[5] T. Asano, B. Bhattacharya, M.Keil and F.Yao,"Clustering Algorithms based on minimum and maximum spanning trees". *In Proceedings of the 4th Annual Symposium on ComputationalGeometry,*Pages252-257,1988.

[6] D. Avis "Diameter partitioning", *Discrete and Computational Geometry*, 1:265-276, 1986.

[7] S. Bay and M. Schwabacher, "Mining distance-based outliers in near Linear Time with Randomization and a Simple Pruning Rule". *SIGKDD '03, Washington, DC, USA* 2003.

[8] C. Bei and R. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization", *IEEE Trans. Commun*, 33:1132-1133, 1985.

[9] S. Chen and W. Hsieh, "Fast algorithm for VQ codebook Design", *IEEE Proc.*, 138: 357-362, 1991.

[10] B.Custem and I.Gath, "Detection of Outliers and Robust Estimation using Fuzzy clustering", *Computational Statistics & data Analyses* 15,pp.47-61, 1993.

[11] Fawaz A.M. Masoud, Moh'd Belal Al-Zoubi, Imad Salah and Ali AL-Dahoud, "Fast Algorithms for Outlier Detection", *Journal of Computer Science* 4(2): 129-132, 2008.

[12] Feng Luo,Latifur Kahn, Farokh Bastani, I-Ling Yen, and Jizhong Zhou, "A dynamically growing self-organizing tree(DGOST) for hierarchical gene expression profile" ,*Bioinformatics*,Vol 20,no 16, pp 2605-2617, 2004.

[13] M. Fredman and D. Willard, "Trans-dichotomous algorithms for minimum spanning trees and shortest paths", *In Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 719-725, 1990.

[14] Gath and A.Geva, "Fuzzy Clustering for the estimation of the Parameters of the components of Mixtures of Normal distribution", Pattern Recognition letters,9,pp.77-86, 1989.

[15] B. Ghosh-Dastidar and J.L. Schafer," Outlier Detection and Editing Procedures for Continuous Multivariate Data", *ORP Working Papers*, September 2003. (http://www.opr.princeton.edu/papers/), visited 20.09.2004.

[16] A. Hardy, "On the number of clusters", *Computational Statistics and Data Analysis*, 23, pp. 83–96, 1996.

[17] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: Data mining, inference and prediction", *Springer-Verlag*, 2001.

[18] Z. He, X. Xu and S. Deng, "Discovering cluster-based Local Outliers". *Pattern Recognition Letters*, Volume 24, Issue 9-10, pp 1641 – 1650, June 2003.

[19] H.Gabow, T.Spencer and R.Rarjan. "Efficient algorithms for finding minimum spanning trees in undirected and directed graphs", *Combinatorica*, 6(2):pp 109-122, 1986.

[20] M. Jaing, S. Tseng and C. Su, "Two-phase Clustering Process for Outlier Detection ", *Pattern Recognition Letters*, Volume 22, Issue 6 – 7, pp 691 – 700, May 2001.

[21] S. John Peter, S.P. Victor, "A Novel Algorithm for Meta similarity clusters using Minimum spanning tree", *International Journal of computer science and Network Security*. Vol.10 No.2 pp. 254 – 259, 2010

[22] D. Karger, P. Klein and R. Tarjan, "A randomized linear-time algorithm to find minimum spanning trees". *Journal of the ACM*, 42(2):321-328, 1995.

[23] E. Knorr and R. Ng, "A Unified Notion of Outliers: Properties and Computation". *In*

Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, pages 219 – 222, August 1997.

[24] E.Knorr and R.Ng, "Algorithms for Mining Distance-based Outliers in Large Data sets", *Proc.the 24th International Conference on Very Large Databases(VLDB)*,pp.392-403, 1998.

[25] E.Knorr, R.Ng and V.Tucakov, "Distance- Based Outliers: Algorithms and Applications", *VLDB Journal*, 8(3-4):237-253, 2000.

[26] J. Kruskal," On the shortest spanning subtree and the travelling salesman problem", *In Proceedings of the American Mathematical Society*, pp 48-50, 1956.

[27] Y. Linde, A. Buzo and R. Gray, 1980. "An algorithm for vector quantizer design". *IEEE Trans. Commun.*, 28: 89-95.

[28] A.Loureiro, L.Torgo and C.Soares, "Outlier detection using Clustering methods: A data cleaning Application", *In Proceedings of KDNet Symposium on Knowledge-based systems for the Public Sector*. Bonn, Germany, 2004.

[29] Moh'd Belal Al-Zoubi, "An Effective Clustering-Based Approach for Outlier Detection", *European Journal of Scientific Research*, Vol.28 No.2 ,pp.310-316, 2009

[30] Moh'd Belal Al-Zoubi and Nadim Obeid, "A Fast Distance-Based Approach to Detect Outliers", *Journal Of Computer Science* 3 (12) ,pp.944-3947, 2007

[31] J. Nesetril, E.Milkova and H.Nesetrilova. Otakar boruvka "On Minimum spanning tree problem: Translation of both the 1926 papers, comments, history. DMATH:", *Discrete Mathematics*, 233, 2001.

[32] K.Niu, C.Huang, S.Zhang and J.Chen, "ODDC: Outlier Detection Using Distance Distribution Clustering", T.Washio et al. (Eds.): *PAKDD 2007 Workshops, Lecture Notes in Artificial Intelligence (LNAI)* 4819,pp.332-343,*Springer-Verlag*, 2007.

[33] Oleksandr Grygorash, Yan Zhou, Zach Jorgensen, "Minimum spanning Tree Based Clustering Algorithms", *Proceedings of the 18th IEEE International conference on tools with Artificial Intelligence (ICTAI'06)*, 2006.

[34] K. Paliwal and V. Ramasubramanian, "Effect of ordering the codebook on the efficiency of the partial distance search algorithm for vector quantization", *IEEE Trans. Commun*., 37:538-540, 1989.

[35] F. Preparata and M.Shamos, "Computational Geometry: An Introduction". *Springer-Verlag*, Newyr, NY ,USA, 1985.

[36] R. Prim, "Shortest connection networks and some generalization". *Bell systems Technical Journal*,36:1389-1401, 1957.

[37] S. Ramaswamy, R. Rastogi and K. Shim, "Efficient Algorithms for Mining Outliers from Large Data Sets", *In Proceedings of the 2000 ACM SIGMOD InternationalConference on Management of Data*, Volume 29, Issue 2, pages 427 – 438, May 2000.

[38] D. M. Rocke and J. J. Dai, "Sampling and subsampling for cluster analysis in data mining: With applications to sky survey data", *Data Mining and Knowledge Discovery*, 7, pp. 215–232, 2003.

[39] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms", *in Proceedings Sixteenth IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2004, Los Alamitos, CA, USA, IEEE Computer Society*, pp. 576–584, 2004.

[40] S. Still and W. Bialek, "How many clusters?, An information-theoretic perspective", *Neural Computation*, 16, pp. 2483–2506, 2004.

[41] Stefan Wuchty and Peter F. Stadler, "Centers of Complex Networks", 2006

[42] C. Sugar and G. James, "Finding the number of clusters in a data set, An information theoretic approach", *Journal of the American Statistical Association*, 98 pp. 750–763, 2003.

[43] N. Venkateswarlu and P. Raju, "A new fast classifier for remotely sensed images". *Int.J. Remote Sens.*, 14:383-390, 1993.

[44] G. Williams, R. Baxter, H. He, S. Hawkins and L. Gu, "A Comparative Study for RNN for Outlier Detection in Data Mining", *In Proceedings of the 2nd IEEE International Conference on Data Mining*, page 709, Maebashi City, Japan, December 2002.

[45] Y.Xu, V.Olman and D.Xu, "Minimum spanning trees for gene expression data clustering", *Genome Informatics*, 12:24-33, 2001.

[46] K.Yoon, O.Kwon and D.Bae, "An approach to outlier Detection of Software Measurement Data using the K-means Clustering Method", *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, Madrid.pp.443-445, 2007.

[47] C. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters", *IEEE Transactions on Computers*, C-20:68-86, 1971

[48] J.Zhang and N.Wang, "Detecting outlying subspaces for high-dimensional data: the new task, Algorithms and Performance", *Knowledge and Information Systems*, 10(3):333-555, 2006.