# A Hierarchical ST-DBSCAN with Three Neighborhood Boundary Clustering Algorithm for Clustering Spatio–temporal Data

Amalia Mabrina Masbar Rus[1], Zulaiha Ali Othman[2], Azuraliza Abu Bakar[3], Suhaila Zainudin[4]
Department of Informatics, Universitas Syiah Kuala, Banda Aceh 23114, Indonesia[1]
Center for Artificial Intelligence Technology (CAIT), Universiti Kebangsaan Malaysia, Bangi 43600, Malaysia[2, 3, 4]

***Abstract*—Clustering Spatio-temporal data is challenging because of the complexity of processing the spatial and temporal aspects. Various enhanced clustering approaches, such as partition-based and hierarchical-based algorithms have been proposed. However, the ST-DBSCAN density-based algorithm is commonly used to process irregularly shaped clusters. Moreover, ST-DBSCAN considers neighborhood parameters as spatial and non-spatial. The preliminary results from our experiments indicate that the ST-DBSCAN algorithm addresses temporal elements less effectively. Therefore, an improvement to the ST-DBSCAN algorithm was proposed by considering three neighborhood boundaries in neighborhood function. This experiment used the El Niño dataset from the UCI repository. The experimental results show that the proposed algorithm increased the performance indices by 27% compared to existing approaches. Further improvement using the hierarchical Ward's method (with thresholds of 0.3 and 0.1) reduced the number of clusters from 240 to 6 and increased performance indices by up to 73%. It can be concluded that ST-HDBSCAN is a suitable clustering algorithm for Spatio-temporal data.**

***Keywords*—Data mining; hierarchical clustering; density-based clustering; spatio–temporal clustering**

## I. Introduction

Clustering is a process for grouping data based on similarity distance. It is an effective data mining technique for data segmentation, feature selection, pattern recognition, and anomaly detection. Clustering is an unsupervised method that does not require a prior definition of the input data classes. Clustering techniques help uncover hidden patterns in the examined data [1]. Unlike conventional clustering algorithms that process nonspatial or non-temporal data, clustering spatio–temporal data is challenging. Spatio-temporal data is different from relational data, in which computational approaches are developed for both spatial and temporal attributes [2]. Because of spatio–temporal data structure features, which record the general variables and the corresponding location and time [3], it became challenging to cluster the spatial and temporal data together.

Spatio–temporal clustering is an emerging research area. It gains actual location and time information from the enormous amount of geographical data provided by GPS, satellite, wireless technology, sensor networks, and other devices that could transmit location and time-stamped data. For instance, an organization has invested more resources in obtaining hidden knowledge and information from spatio–temporal data making research in this field more critical. The work by Bogorny and Shekar [4], Mazimpaka and Timpf [5], and Atluri *et al.* [6] have reviewed spatio–temporal data mining and its applications in surface ozone (O3) variations [7], forest fire [8], groundwater potential zone [9], citizens security [10], traffic congestion [11], healthcare, and social media.

Kisilevich *et al.* [12] categorized spatio–temporal data based on temporal extension, spatial extension, and spatial location. Spatial extension expands the spatial shape from points into lines and then areas. The expansion starts from a single snapshot to an updated snapshot and the completed time series in temporal extension. The spatial location is divided into two categories depending on the data collection location (fixed or dynamic). Based on these extensions, Kisilevich *et al.* [12] defined the following five types of spatio–temporal data: ST-events, geo-referenced variable, geo-referenced time series, moving objects, and trajectories.

M. Y. Ansari *et al.* [13] presents six categories of spatiotemporal clustering algorithms: event clustering, geo-referenced data item clustering, geo-referenced time series clustering, moving clusters, trajectory clustering, and semantic based trajectory data mining. ST-events have a fixed location and only store one snapshot of variable values. The geo-referenced variable also has a fixed location but stores only the current updated values. Similarly, geo-referenced time series also have a fixed location, but it stores the entire history of variable values as time-series data for each location. Moving objects and trajectories have dynamic locations. The object changes its location over time; however, only current values are recorded. In contrast, trajectories record each object's movement as a completed time series. Similar to Madraky *et al.* [14], this research focuses on clustering geo-referenced time-series data and utilizes a nature-inspired approach.

Various clustering approaches have been proposed to enhance the traditional algorithm, such as density-based, partition-based, and hierarchical-based [15]. The density-based algorithm is widely used because it can process irregular-shaped clusters, e.g. ST-DBSCAN [16], ST-OPTICS [17], ST-Shared Nearest Neighbors (SNN) and ST-SEP-SNN [18], P-DBSCAN [19], ST-DCONTOUR [20], RT-DBSCAN [21], CorClust [22], and MDST-DBSCAN [23]. One of the commonly used spatio–temporal density-based clustering approaches is Spatio–temporal (ST)-Density-based Spatial

Clustering of Applications with Noise (DBSCAN) [16]. However, the algorithm does not address the temporal elements of data. The neighborhood parameters are defined as spatial and non-spatial objects only. Therefore, the maximum temporal distance was later introduced to improve the ST-DBSCAN algorithm.

Other techniques extended to other traditional density-based clustering algorithms. For example, Spatio–temporal Shared Nearest Neighbor (ST-SNN) and Spatio–temporal Separated Shared Nearest Neighbor (ST-SEP-SNN) extend SNN by Ertöz *et al.* [24] but introduce the polygon distance algorithm to search core polygon [18]. Their results have been compared with that of PDBSCAN [19]. 4D+SNN also added weighting for spatial and temporal attributes [25]. The ST-DCONTOUR algorithm by Zhang and Eick [20] emphasizes the batching process of SNN density-based clustering algorithms.

The ST-DBSCAN algorithm by Birant and Kut [16] is one of the standard benchmark algorithms for spatio–temporal clustering. In ST-DBSCAN, introducing a new distance limit for spatial data called *Eps 2* enhanced the DBSCAN algorithm. The sea surface temperature, sea surface height, and wave height in the Black, Marmara, Aegean, and Mediterranean seas were clustered. The result was presented as a map with labels of the cluster numbers for each group. Besides, utilizing a cluster map emphasizes the spatial aspect of the data rather than the temporal aspect. Therefore, this research aims to improve ST-DBSCAN by considering spatial and non-spatial.

The following Sections II discuss how the ST-DBSCAN algorithm is limited and how to improve it by proposing Three Neighborhood Boundary in neighborhood function and improving it again using the hierarchical ward's method. Section III discusses the dataset and parameter settings; Section IV discusses the results obtained from experiments, including comparison results with the benchmark algorithm, and the conclusion and future work.

## II. Related Work

Density-Based clustering is an unsupervised learning technique that locate distinct groups or clusters in the data. These techniques are based on the notion that a cluster in data space is a contiguous region of high point density, separated from other such clusters by contiguous regions of low point density. The fundamental density-based clustering algorithm is called Density-Based Spatial Clustering of Applications with Noise (DBSCAN). Outliers and noise-filled massive amounts of data can be used to find clusters of various sizes and shapes.

Three minor modifications to DBSCAN, known as ST-DBSCAN , are proposed by D. Birant *et al*. [16]. ST-DBSCAN deal with the detection of (i) core items, (ii) noise objects, and (iii) neighboring clusters. The ability to identify clusters on spatial-temporal data is the primary motivation behind this modification. When clusters of various densities occur, the second adjustment is required to locate noisy objects. D. Birant et al. [16] introduce the idea of density factor. Each cluster is given a density factor, which indicates how dense the cluster is. The third adjustment compares the cluster's average value with a new value that will soon be available.

In ST-DBSCAN, the DBSCAN algorithm was enhanced by introducing a new distance limit for spatial data called Eps 2. In terms of its research, sea surface temperature, sea surface height, and wave height in the Black Sea, the Marmara Sea, the Aegean Sea, and the Mediterranean Sea are clustered. The clustering result was presented on a map labeled with a cluster number for each area. Using a cluster map, the result focused more on the spatial part of data and lacked in showing the temporal part.

The problem with ST-DBSCAN is the lack of an algorithm on the temporal aspect of spatio-temporal data. This is because the temporal data were separated manually by filtering the data that occurred on a consecutive day or the same day in a different year [16]. Thus, the algorithm did not identify the cluster with a pattern that exists continuously in two different years. The cluster generated took into account the spatial and non-spatio-temporal part of the data; however, it lacks the usage of the temporal aspect of the data.

The clustering algorithm "Spatio-Temporal-Ordering Points to Identify Clustering Structure (ST-OPTICS)" was proposed by K.P. Agrawal *et al.* [17] and is a modified version of the previous density based technique "Ordering Points to Identify Clustering Structure (OPTICS)." The proposed algorithm can produce spatio-temporal clusters and address issues such as (i) handling spatio-temporal data, non-spatial values to take into account temporal dimensions, (ii) algorithm does not depend on dimensions of data, so it is ready to handle n-dimensions, and (iii) independence of ordering of observations in database can be observed by the working principle of the proposed technique like it first performs Orde, (iv) locating nested and nearby clusters, and (v) ultimately, it is also scalable.

The ST-OPTICS algorithm employs ST-DBSCAN to extract the clusters and then performs hierarchical clustering to aggregate the clusters. The result generated has better performance indices compared to ST-DBSCAN. However, comparing to a similar version of ST-DBSCAN, ST-OPTICS still performs clustering using spatial and non-spatio-temporal aspects of the data but lacks the temporal aspect of the data. Similarly, the results are also shown in the form of maps which limit the description of the temporal part of data. Therefore, there is still a need for a clustering algorithm that could cluster spatio-temporal data that will account not only the spatial and non-spatio-temporal parts of data but also the temporal part of data.

To cluster overlapping polygons that can change their positions, sizes, and shapes over time, Sujing Wang et al. [18] introduce two new spatiotemporal clustering algorithms, called Spatiotemporal Shared Nearest Neighbor clustering algorithm (ST-SNN) and Spatiotemporal Separated Shared Nearest Neighbor clustering algorithm (ST-SEP-SNN). The core polygon notion and the spatial closest neighborhood of a polygon are both redefined by Sujing Wang *et al.* [18]. Even with high-dimensional data with outliers, ST-SNN and ST-SEP-SNN may locate clusters of various sizes, shapes, and densities.

P-DBSCAN, a novel DBSCAN for analysis of locations and events utilizing a collection of geo-tagged photographs, was proposed by Slava Kisilevich *et al.* [19]. Two new ideas were introduced by P-DBSCAN: (i) the density threshold, which is based on the population of the neighborhood, and (ii) adaptive density, which is utilized to quickly converge to high-density areas.

Yongli Zhang *et al.* [20] proposed the ST-DCONTOUR algorithm, a unique serial density-contour-based spatio-temporal clustering technique that uses location streams as input and a model-based clustering approach to produce spatio-temporal clusters. In our method, the incoming data is divided into batches, and we use a serial technique to construct spatial clusters for each batch individually first. The formation of spatio-temporal clusters is then accomplished by establishing ongoing connections between spatial clusters in successive batches. Our method makes use of contouring algorithms to recognize spatial clusters as closed contours of a region whose density exceeds a predetermined threshold and contour analysis methods to recognize persistent, transient, and freshly formed spatial clusters in batches.

Yikai Gong *et al.* [21] proposed a technique called RT-DBSCAN to enable continuous cluster checkpointing-based real-time data clustering. In order to provide scalability, the platform is developed using container-based technologies and Apache Spark running on massive Cloud resources. The real-time data streams were the exclusive focus of the DBSCAN algorithm. With the use of density-based clustering, the DBSCAN algorithm handles rapidly expanding high velocity data streams.

Based on empirical geographical correlations across time, M. Hüsch *et al.* [22] proposed a method called CorClustST. CorClustST compares and interprets clustering outcomes for various scenarios, such as those involving several underlying variables or various time scales. The clustering technique is extended in a way that enables massively parallel execution and works around memory constraints.

C. Choi *et al.* [23] proposed a clustering method, called MDST-DBSCAN for large-scale, multidimensional spatiotemporal data in a reliable and efficient manner. The MDST-DBSCAN is applied to idealized patterns and a real data set, and the results from both examples demonstrate that it can identify clusters accurately within a reasonable amount of time. The MDST-DBSCAN has a limitation in that the method of defining the neighbors in the clustering process operates conservatively.

## III. Proposed Method

The spatio-temporal dataset is different from the traditional dataset in many ways. Firstly, it contains implicit spatial or temporal data that initially need to be calculated. Secondly, spatio-temporal data has granularity, and making its selection differently will impact mining results. Thirdly, spatio-temporal data has an auto-correlation.

The ST-DBSCAN algorithm was proposed by Birant and Kut [16] and has been widely utilized in various studies, achieving impressive results on spatio–temporal data. The distance between two objects *o* and *p* denoted as dist *(o, p)* is defined by measures such as the Manhattan, Euclidean and Haversine distance functions. The neighborhood of object o is defined as $\{p \in D \vee dist(o,p) \leq Eps\}$, where $p$ is another object in database $D$ and *Eps* is the predefined minimum distance between object *o* and *p*. Eq. 1 formally defines the neighborhood utilized in ST-DBSCAN as follows: *Eps 1* for spatial data and *Eps 2* for non-spatiotemporal data, whereas Eq. 2 defines the core objects. As per Eq. 2, objects that satisfy the minimum number of neighborhoods within the radius of *Eps* and are greater than or equal to the *MinPts* are defined as core objects. The ST-DBSCAN is also defined by its density-reachable, density-reachable, density-connected, density-based cluster and border objects [16].

The temporal data aspect was separated manually by filtering daily observations in different years. Thus, this method cannot identify clusters that contain patterns of two different years. The neighborhood in Eq. 1 only considers the spatial and non-spatial distances. It does not include the temporal features, as the algorithm uses two boundary neighborhood-clustering methods. Fig. 1 illustrates the limitation of ST-DBSCAN. Only the spatial dimension is considered in the clustering process; thus, only one cluster will be generated. However, the distance becomes apparent if the data is viewed from the right or topside. Hence, the blue and grey boxes will not be grouped into the same cluster.

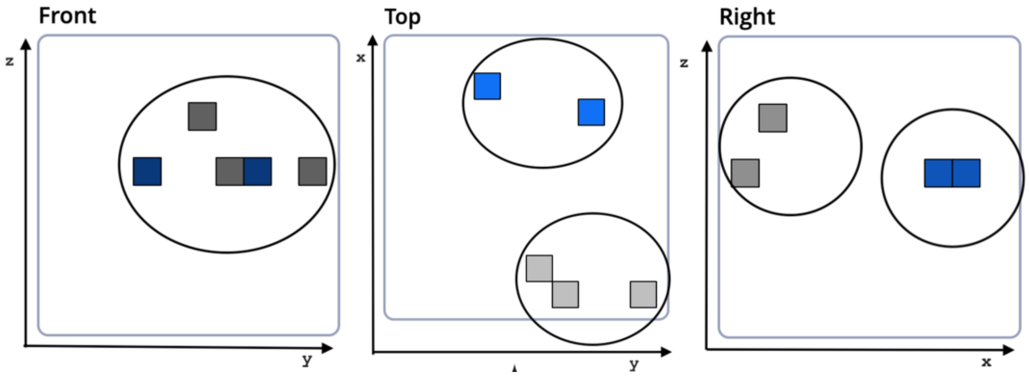

Fig. 1. Two-dimensional view of spatio–temporal data with views from the front, right and top.

The proposed Hierarchical ST-DBSCAN (ST-HDBSCAN) algorithm was developed in two parts. The first part improves the neighborhood of the ST-DBSCAN by introducing Three Neighborhood Boundary. The second part introduces the hierarchical ward's method to improve the performance of DBSCAN.

$$spatial\_dist(o,p) \leq Eps1) \wedge (var\_dist(o,p) \leq Eps2) \rightarrow Neighbor(o,p) \quad (1)$$

$$NumNeighbor(o) \geq MinPts \rightarrow CoreObject(o) \quad (2)$$

### A. *ST-DBSCAN with Three Neighborhood Boundary*

Fig. 2 illustrates the spatio–temporal data in three dimensions to incorporate the temporal aspect. The spatial dimension is represented along the z-axis and y-axis, whereas the temporal dimension is represented along the x-axis. In the three-dimensional view, it becomes clear that when the temporal aspect is taken into consideration, proper separation is achieved: the grey boxes are on the positive side of the x-axis, and the blue boxes are on the negative side of the x-axis. Therefore, ignoring the temporal aspect could lead to inconsistent clustering.

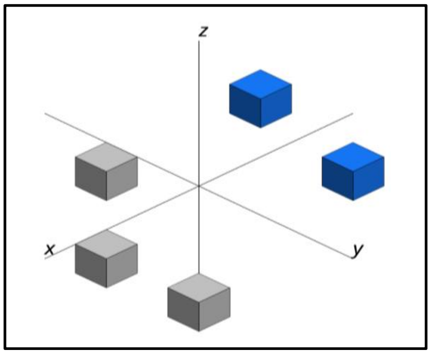

Fig. 2. A three-dimensional view of spatio–temporal data (the spatial dimension is represented along the z-axis and y-axis while the temporal dimension is represented along the x-axis).

Eq. 3 defines the spatio–temporal neighborhood utilized in the improved ST-DBSCAN algorithm. It is composed of Three Neighborhood Boundary: *Eps 1* for spatial data, *Eps 2* for non-spatio-temporal data, and *Eps 3* for temporal data.

In the improved ST-DBSCAN algorithm, temporal neighbors are defined by *Eps 3* to limit temporal distance. As a result, two objects are temporally close if the distance between them in terms of time is in the range of *Eps 3*. For example, the temporal distance could be defined over 1 year or 1 month. If values are collected daily, a temporal distance defined over 1 year will limit the data to the range of a year, that is, 182 days before and after the date for each value. Therefore, to spatially and temporally cluster the data, there is a need to develop a new algorithm that could incorporate the spatial and temporal aspects of data. Based on this information, in this research, the density-based algorithm ST-DBSCAN is enhanced by incorporating a temporal distance limit called maximum temporal distance (*Eps 3*). The maximum temporal distance is defined in Eq. 4.

The proposed approach can be illustrated as follows. Given objects O (*s1*, *s2*, *n1*, *n2*, *t1*) and P (*s3*, *s4*, *n3*, *n4*, *t2*), each comprising of five variables (latitude, longitude, wind speed, sea surface temperature, and date), Eq. 5 (haversine function) is utilized to compute the spatial distance (*spatial_dist)* between O and P and Eq. 6 (Euclidean distance) computes the non-spatiotemporal distance (*non_st_dist*). Also, Eq. 7 computes the absolute date difference between the two objects measured in days. The Euclidean distance was an unsuitable measure for the earth's curved surface due to the utilized spatial data (latitude and longitude coordinates). Therefore, the haversine distance was the preferred option for spatial distance computation.

$$(spatial_{dist}(o,p) \leq Eps1) \wedge (non_{st_{dist}}(o,p) \leq Eps2) \wedge (temp_{dist}(o,p) \leq Eps3) \rightarrow Neighbor(rs,p) \quad (3)$$

$$spatial_{dist} = 2rarcsin\left(\sqrt{sin^2\left(\frac{s_3-s_1}{2}\right) + cos(s_1)cos(s_3)sin^2\left(\frac{s_4-s_2}{2}\right)}\right) \quad (4)$$

where:

*h*: is the haversine distance

*r*: is the radius of Earth with value 6371 km

$\varphi_1$: is the latitude of point 1

$\varphi_2$: is the latitude of point 2

$\lambda_1$: is the longitude of point 1

$\lambda_2$: is the longitude of point 2

$$spatial_{dist} = 2rarcsin\left(\sqrt{sin^2\left(\frac{s_3-s_1}{2}\right) + cos(s_1)cos(s_3)sin^2\left(\frac{s_4-s_2}{2}\right)}\right) \quad (5)$$

$$non_{st_{dist}} = \sqrt{(n1-n3)^2 + (n2-n4)^2} \quad (6)$$

$$temp\_dist = |t2 - t1| \quad (7)$$

Pseudocode 1 describes the pseudocode of ST-DBSCAN with maximum temporal distance (ST-DBSCAN *Eps 3*). The algorithm starts by initializing the cluster number as described in line 13. Afterward, it searches for the neighbors of unlabelled data utilizing the retrieve-neighbor function (lines 15–16). The expectation is that if the retrieve-neighbor function meets all the conditions, the objects are considered neighbors. Conversely, if the number of neighbors does not satisfy the *MinPts*, these objects will be labeled as noise (lines 19–20). Otherwise, its neighbors are labeled as the same cluster (lines 23–24). Therefore, the algorithm checks the neighbors of each discovered neighbor (lines 26–29). Similarly, the neighbor would be labeled the same cluster (lines 30–33). These processes are repeated until all objects have been observed. Lastly, the Clustered_Data set, which contains all data labeled with a Cluster ID, is returned as the result of the algorithm (line 42).

Pseudocode 1: Pseudocode of ST-DBSCAN with Eps3 Algorithm

```
Inputs:
  D = (d1, d2, … , dn) Set of objects
  N = (d1, d2, … , dn) Set of Retrieved Neighbors
  R = (r1, r2, … , rn) Set of Retrieved Neighbors of N
  Eps1: Maximum spatial distance value
  Eps2: Maximum non-spatial distance value
  Eps3: Maximum temporal distance value
  Mint: Minimum number of points within Eps1, Eps2, and Eps3

Output
  Clustered_Data : Data that has been labeled with ClusterID

Initialize cluster label as 0

For i in range of number of data rows in D
  If label of the current data with index i set as "unmarked"
    Retrieve neighbors of data with index i based on Eps1, Eps2, and Eps3 values
    Store the neighbors in N
    If number of neighbors are less than MinPts
      Set label of data with index i as "noise"
    Else
      Create new cluster label
      For all neighbors in N
        Set cluster label for each neighbor as new cluster label
      End For
      For all neighbors in N
        Get one neighbor data and store it as current object
        Retrieve neighbors of current object based on Eps1, Eps2, and Eps3
        Store the neighbors of current object in R
        If number of neighbors in R less than or equal to MinPts
          For all neighbors in R
            If the neighbor label's is not "noise" or "unmarked"
              Set cluster label for the current object as new cluster label
            End If
          End For
        End If
      End For
    End Else
  End If
End For

Return data D with cluster label as Clustered_Data
```

Pseudocode 2 shows the pseudocode of the retrieve-neighbor function. This function returns the neighbors of an object by computing the spatial, temporal, and non-spatiotemporal distance between the object and other objects in the dataset. The algorithm starts by excluding the current object from the data (lines 12–13). It then computes the spatial distance between the two objects using the Haversine function (Eq. 5) and stores the distance in the spatial_dist variable (lines 15–16). Afterward, it computes the non-spatiotemporal distance utilizing the Euclidean distance function (Eq. 6) and stores the result in the non_st_dist variable (lines 17–18). Finally, it calculates the temporal distance by calculating the date differences between the current object and other objects in the dataset (Eq. 7) and stores it in temporal_dist (lines 19–21).

The next step determines the neighbors of each object in the dataset in turn by comparing the spatial_dist, non_st_dist and temporal_dist with the values of *Eps 1*, *Eps 2* and *Eps 3*. For each evaluation, if each distance value is less than the corresponding *Eps* value, the current object (*p*) is considered a neighbor of the examined object (*o*) and will be added to the set of the object's neighbors (Data_Neighbors) (lines 22–26). The Data_Neighbors is returned to the ST-DBSCAN *Eps 3* Algorithm (line 30).

Line Pseudocode 2: Pseudocode of the Retrieve Neighbors for ST-DBSCAN with maximum temporal distance (Eps 3)

```
Inputs:
  D     : Data of all objects
  n     : Index of current object
  Eps1: Maximum spatial distance value
  Eps2: Maximum non-spatial distance value
  Eps3: Maximum temporal distance value

Output
  Data_neighbors : Data of current object's Neighbors

For i in range of data row in D
  If i is equal to n
    Continue the program
  Else
    Calculate spatial distance of object index i with object index n using Harversine function
    Store the spatial distance as spatial_dist
    Calculate non-spatio-temporal distance of object i with object index n using Euclidean function
    Store the non-spatio-temporal distance as non_st_dist
    Calculate temporal distance of the object index i with object index n by
    subtracting time of object index i with object index n
    Store the temporal distance as temporal_dist
    If spatial_dist is less than Eps1 and
      non_st_dist is less than Eps2 and
      temporal_dist less than Eps3
        Set object index i as neighbor of object index n
        Add object index i to Data_neighbors
    End If
  End If
End For
Return Data_neighbors
```

### *B. An Improved ST-DBSCAN with Hierarchical Ward's Method*

ST-HDBSCAN improves the ST-DBSCAN algorithm by utilizing the hierarchical ward's method. Pseudo code 3 shows the ST-HDBSCAN algorithm, which consists of the clustering and hierarchical phases. Lines 15–41 outline the clustering procedure with *Eps 3*. After clustering, the hierarchical ward's method is employed to aggregate the results (line 50). It combines close and similar clusters into new larger ones based on the temporal distance value (computed by fast dynamic time warping) (lines 43–48). For the hierarchical phase, the Fast Dynamic Time Warping (FastDTW) algorithm warps the timeline of data points into compressed values (line 47) to simplify hierarchical cluster grouping. After computing the similarity pair of each cluster, the distance is condensed to minimize duplication (line 49). The minimum distance is selected as the cut point of the hierarchy to obtain the proper clusters (line 51).

Line Pseudocode 3: Pseudocode of the ST-HDBSCAN Algorithm

```
 1.  Inputs:
 2.    D = (d1, d2, … , dn) Set of objects
 3.    N = (d1, d2, … , dn) Set of Retrieved Neighbors
 4.    R = (r1, r2, … , rn) Set of Retrieved Neighbors of N
 5.    Eps1: Maximum spatial distance value
 6.    Eps2: Maximum non-spatial distance value
 7.    Eps3: Maximum temporal distance value
 8.    Mint: Minimum number of points within Eps1, Eps2, and Eps3
 9.
10.  Output
11.    Clustered_Data : Data that has been labeled with ClusterID
12.
13.  Initialise cluster label as 0
14.
15.  For i in range of number of data rows in D
16.    If label of the current data with index i set as "unmarked"
17.      Retrieve neighors of data with index i based on Eps1, Eps2, and Eps3 values
18.      Store the neighbors in N
19.      If number of neighbors are less than MinPts
20.        Set label of data with index i as "noise"
21.      Else
```

```
      Create new cluster label
      For all neighbors in N
        Set cluster label for each neighbor as new cluster label
      End For
      For all neighbors in N
        Get one neighbor data and store it as current object
        Retrieve neighbor of current object based on Eps, Eps2, and Eps3
        Store the neighbor of current object in R
        If number neighbor in R less than or equal to MinPts
          For all neighbor in R
            If the neighbor label’s is not “noise” or “unmarked”
              Set cluster label for the current object as new cluster label
            End if
          End for
        End if
      End for
    End Else
  End If
End For
Return data D with cluster label as Clustered_Data

For i in range of number of cluster in Clustered_Data
  For j in range of number of cluster in Clustered_Data
    Get non spatial data with ClusterID i
    Get non Spatial data with ClusterID j
    Calculate FastDTW time series distance between cluster1 and cluster2
    Store the calculated value in variable ‘dist’
Convert pair distance ‘dist’ into condensed distance
Generate hierarchy with ward method
Get new cluster label from hierarchy with cutting level of threshold value
For i in range of Clustered_Data
  Replace cluster label with the new cluster label from hierarchy
Return Cluster_Data with new label
```

## IV. Experiment Results and Discussion

### A. Equations

The El-Niño dataset from the UCI Repository was utilized for this research. This dataset was provided to study the El-Niño/Southern Oscillation cycle phenomena, which is well-known in climatology as the cause of the climate anomalies worldwide [26]. The data was collected from 1980 to 1998 and comprised different observations for each buoy. In total, there are 75 buoys and 178,080 observations in the dataset. The dataset was pre-processed using several techniques such as data cleaning, data transformation, and data reduction. The data cleaning technique employed the K-Nearest Neighbor Algorithm as the imputation technique. A new variable was added for data transformation, and Min-Max normalization was implemented. Since the wind speed value is vital in determining El-Niño cycles, a new variable, ‘Wind Speed,’ is also estimated using the Zonal and Meridional wind values. On the other hand, air temperature and humidity were removed from the dataset because air temperature values are similar to sea surface temperature. It does not contribute much to determine El-Niño because 36% were missing the humidity values.

### B. Parameter Setting

Table I shows the parameter settings used in the two experiments. The settings are as follows. Experiment 1 improves the ST-DBSCAN algorithm with the maximum temporal distance, whereas Experiment 2 improves it with the hierarchical ward’s method.

TABLE I. Parameters Settings of Experiment 1 and 2

| Parameters | Experiment 1 | | Experiment 2 | |
|---|---|---|---|---|
| | *ST-DBSCAN* | *ST-DBSCAN Eps 3* | *ST-HDBSCAN 0.3* | *ST-HDBSCAN 0.1* |
| Eps 1 (spatial) | 5000 | 5000 | 5000 | 5000 |
| Eps 2 (non-spatio–temporal) | 0.3 | 0.3 | 0.3 | 0.3 |
| Minimum points | Log(178080) = 12.0899 | Log(178080) = 12.0899 | Log(178080) = 12.0899 | Log(178080) = 12.0899 |
| Eps 3 (temporal) | N/A | 182 | 182 | 182 |
| Threshold value | - | - | 0.3 | 0.1 |

### C. ST-DBSCAN with Three Neighbourhood Boundary Result

Table II and Table III show the clustering results for ST-DBSCAN compared with ST-DBSCAN Eps 3. The results indicate that the ST-DBSCAN algorithm does not consider the dataset's temporal aspect. Obs 87567–174633 was recognized as one cluster: Cluster 6. Its similarity term was based on the spatial distance (latitude, longitude) and non-spatio–temporal distance (wind speed and sea surface temperature) only. Similarly, in Table III, Obs 87799–151340 was grouped into Cluster 4 only. On the other hand, ST-DBSCAN *Eps 3* grouped Obs 87567–174633 into two clusters; 233 (Obs 174633–177614) and 166 (Obs 87230–87567), as shown in Table II. In Table III, Obs 87799–151340 was grouped into four clusters, which are 217, 220, 228, and 166. The results show that ST-DBSCAN *Eps 3* clusters the data according to the temporal distance. It separated observation 177614, collected on November 5, 1998, from observation 87230, collected on April 5, 1994, which have about a four-year difference (Table II).

TABLE II. SAMPLE DATA 1 CLUSTER RESULT FOR ST-DBSCAN COMPARED WITH OTHER ALGORITHMS

| Obs | Date | Lati-tude | Longi-tude | Wind Speed | Sea Surface Temp | Buoy | ST-DBSCAN Eps 3 | STHDB SCAN (0.1) | STHDB SCAN (0.3) | ST-DBSCAN |
|---|---|---|---|---|---|---|---|---|---|---|
| 174633 | 21/05/1998 | 1 | −95.02 | 0.8591 | 0.4721 | 28 | 233 | 9 | 2 | 6 |
| 174692 | 22/05/1998 | 1 | −139.96 | 0.7599 | 0.3475 | 20 | 233 | 9 | 2 | 6 |
| 174750 | 23/05/1998 | 1 | −109.96 | 0.8296 | 0.4640 | 38 | 233 | 9 | 2 | 6 |
| 174930 | 26/05/1998 | 1 | −139.88 | 0.8052 | 0.4815 | 41 | 233 | 9 | 2 | 6 |
| 174932 | 26/05/1998 | 1 | −94.95 | 0.8922 | 0.1858 | 48 | 233 | 9 | 2 | 6 |
| 175050 | 28/05/1998 | −1 | −95.07 | 0.7038 | 0.3980 | 57 | 233 | 9 | 2 | 6 |
| 175161 | 30/05/1998 | 1 | −109.96 | 0.7656 | 0.4383 | 38 | 233 | 9 | 2 | 6 |
| 175273 | 01/06/1998 | 1 | −125.05 | 0.8303 | 0.5245 | 18 | 233 | 9 | 2 | 6 |
| 175584 | 06/06/1998 | −1 | −110 | 0.6650 | 0.3932 | 49 | 233 | 9 | 2 | 6 |
| 175736 | 15/06/1998 | 1 | −95.04 | 0.7965 | 0.5333 | 28 | 233 | 9 | 2 | 6 |
| 175890 | 20/06/1998 | 1 | −94.95 | 0.8520 | 0.5123 | 67 | 233 | 9 | 2 | 6 |
| 177012 | 01/10/1998 | −1 | −95.1 | 0.7074 | 0.5374 | 74 | 233 | 9 | 2 | 6 |
| 177614 | 05/11/1998 | −1 | −95.1 | 0.7664 | 0.6282 | 74 | 233 | 9 | 2 | 6 |
| 87230 | 29/04/1994 | 1 | −179.9 | 0.7779 | 0.5883 | 47 | 166 | 17 | 3 | 6 |
| 87232 | 29/04/1994 | 1 | −124.98 | 0.8210 | 0.3705 | 39 | 166 | 17 | 3 | 6 |
| 87234 | 29/04/1994 | −1 | −154.99 | 0.8699 | 0.4464 | 52 | 166 | 17 | 3 | 6 |
| 87288 | 30/04/1994 | 1 | −110.1 | 0.6636 | 0.3485 | 17 | 166 | 17 | 3 | 6 |
| 87459 | 03/05/1994 | 1 | −125.02 | 0.6858 | 0.3758 | 18 | 166 | 17 | 3 | 6 |
| 87460 | 03/05/1994 | 1 | −179.85 | 0.7786 | 0.3967 | 27 | 166 | 17 | 3 | 6 |
| 87461 | 03/05/1994 | 1 | −179.89 | 0.7994 | 0.3079 | 47 | 166 | 17 | 3 | 6 |
| 87462 | 03/05/1994 | 0.04 | −124.35 | 0.6190 | 0.2969 | 2 | 166 | 17 | 3 | 6 |
| 87513 | 04/05/1994 | 0.04 | −139.99 | 0.7297 | 0.4579 | 3 | 166 | 17 | 3 | 6 |
| 87567 | 05/05/1994 | 1 | −140.26 | 0.7311 | 0.5450 | 0 | 166 | 17 | 3 | 6 |

TABLE III. SAMPLE DATA 2 CLUSTER RESULT FOR ST-DBSCAN COMPARED WITH OTHER ALGORITHMS

| Obs | Date | Lati-tude | Longi-tude | Wind Speed | Sea Surface Temp | Buoy | ST-DBSCAN Eps 3 | STHDB SCAN (0.1) | STHDBS CAN (0.3) | ST-DBSCAN |
|---|---|---|---|---|---|---|---|---|---|---|
| 151340 | 01/04/1997 | 1 | −125.01 | 0.6679 | 0.3474 | 61 | 217 | 15 | 2 | 4 |
| 153131 | 01/05/1997 | 1 | 156.01 | 0.9001 | 0.3355 | 23 | 220 | 11 | 2 | 4 |
| 153133 | 01/05/1997 | −1 | 164.86 | 0.8605 | 0.1847 | 71 | 220 | 11 | 2 | 4 |
| 153134 | 01/05/1997 | −1 | 164.33 | 0.8807 | 0.3707 | 34 | 220 | 11 | 2 | 4 |
| 154959 | 01/06/1997 | 1 | 147.05 | 0.8648 | 0.2051 | 42 | 220 | 11 | 2 | 4 |
| 156725 | 01/07/1997 | 1 | 147.05 | 0.8562 | 0.2225 | 42 | 220 | 11 | 2 | 4 |
| 158575 | 01/08/1997 | −1 | 156.01 | 0.8706 | 0.3821 | 53 | 220 | 11 | 2 | 4 |
| 160492 | 01/09/1997 | 1 | 164.98 | 0.8850 | 0.1768 | 25 | 220 | 11 | 2 | 4 |
| 162354 | 01/10/1997 | 1 | −179.86 | 0.8713 | 0.2448 | 47 | 228 | 8 | 2 | 4 |
| 164284 | 01/11/1997 | −1 | −170.03 | 0.7692 | 0.3121 | 35 | 228 | 8 | 2 | 4 |
| 87679 | 07/05/1994 | −0.5 | 166.92 | 0.8534 | 0.1437 | 16 | 166 | 17 | 3 | 4 |
| 87681 | 07/05/1994 | 1 | −154.93 | 0.8296 | 0.4567 | 43 | 166 | 17 | 3 | 4 |
| 87682 | 07/05/1994 | −1 | −170.03 | 0.8454 | 0.3735 | 35 | 166 | 17 | 3 | 4 |
| 87742 | 08/05/1994 | 1 | −125.07 | 0.6147 | 0.5030 | 18 | 166 | 17 | 3 | 4 |
| 87743 | 08/05/1994 | −0.03 | −170.02 | 0.8541 | 0.3364 | 12 | 166 | 17 | 3 | 4 |
| 87744 | 08/05/1994 | 1 | −170.04 | 0.8864 | 0.3627 | 26 | 166 | 17 | 3 | 4 |
| 87745 | 08/05/1994 | 1 | −154.92 | 0.8224 | 0.3427 | 43 | 166 | 17 | 3 | 4 |
| 87746 | 08/05/1994 | 0.04 | −139.97 | 0.6276 | 0.1990 | 3 | 166 | 17 | 3 | 4 |
| 87799 | 09/05/1994 | 1 | −179.89 | 0.8929 | 0.1243 | 47 | 166 | 17 | 3 | 4 |

A heatmap can be utilized to visualize the performance of the algorithms. Fig. 3 shows the heatmap of the 14 clusters by ST-DBSCAN. Each color represents a different cluster, and each line represents the timeline of a buoy. Each buoy changes its cluster membership accordingly in its timeline, represented by the changing color in the buoy timeline. The heatmap shows that the clusters generated by ST-DBSCAN do not separate the data based on time. The small clusters indicate this problem with various colors spread across the timeline. This problem shows that clusters contain data with similar values but could occur at any time.

Fig. 4 shows the heatmap of the ST-DBSCAN *Eps 3* algorithm. The Fig. 4 clearly shows the separation of data based on time, although 240 clusters were generated in the process. Though the cluster areas are enormous, the members in each cluster are small. However, the clusters are well-ordered based on time. However, since the number of clusters is enormous, it is difficult to interpret the data pattern, and each cluster has a tiny number of members. Therefore, to reduce the number of clusters and increase the understanding of the patterns generated, an improvement is conducted by aggregating the similar clusters based on the temporal similarity of the clusters. This technique was previously employed in ST-OPTICS, where the algorithm combined the density-based and hierarchical-based methods to cluster spatio–temporal data [17].

## D. ST-HDBSCAN Result

The ST-HDBSCAN algorithm was utilized to minimize the number of clusters generated by ST-DBSCAN with *Eps 3*. The clusters generated from the previous experiment were aggregated according to their similarity to form a hierarchy of clusters. FastDTW was employed as the cluster aggregation function. If the FastDTW value is small, the cluster is considered similar and is grouped into a larger cluster.

The cut point of the dendrogram decides the number of clusters generated by ST-HDBSCAN. The higher the cut point affect to the fewer clusters generated. The number of vertical lines at the different cut-point levels indicates the number of clusters. Fig. 5 shows the hierarchical clustering dendrogram for the ST-HDBSCAN algorithm using the hierarchy threshold (0.1). The cut point of the dendrogram is calculated by multiplying the threshold with the maximum hierarchy distance or maximum FastDTW distance of all clusters. For the data utilized in our experiment, the maximum distance was 249.95.

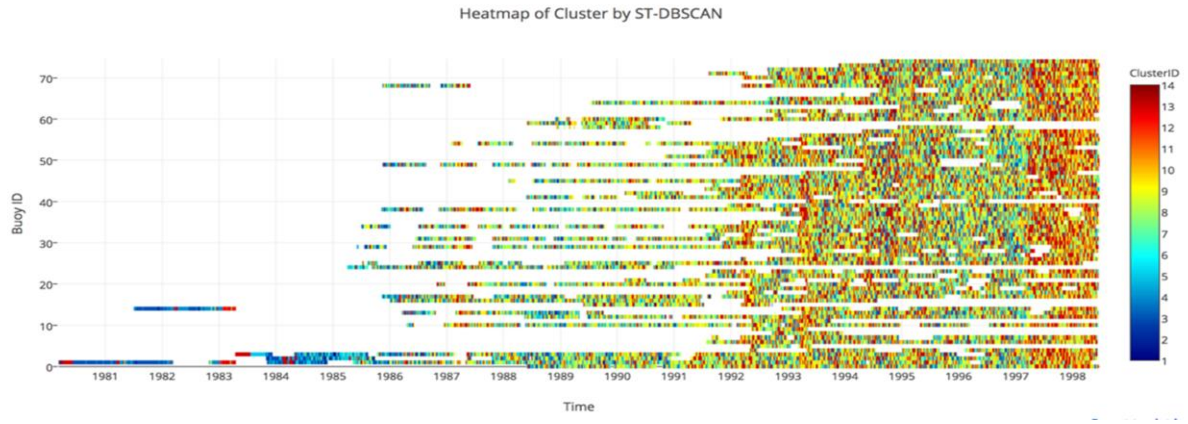

Fig. 3. The cluster heatmap of ST-DBSCAN.

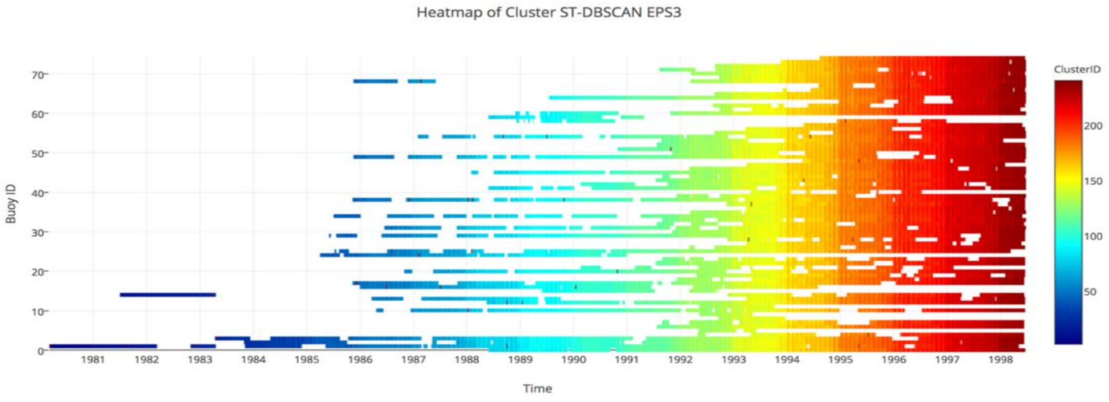

Fig. 4. The cluster heatmap of ST-DBSCAN with Eps 3.

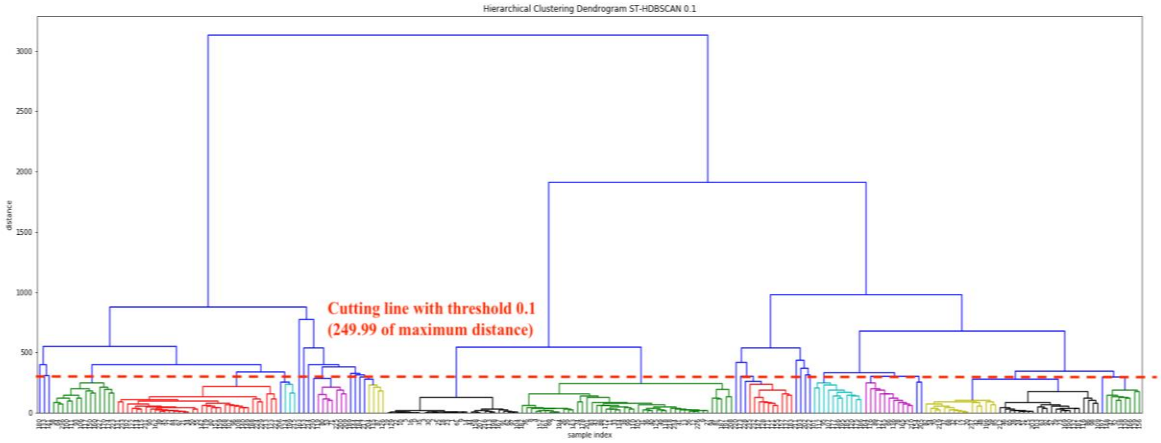

Fig. 5. Hierarchical Clustering Dendrogram ST-HDBSCAN 0.1.

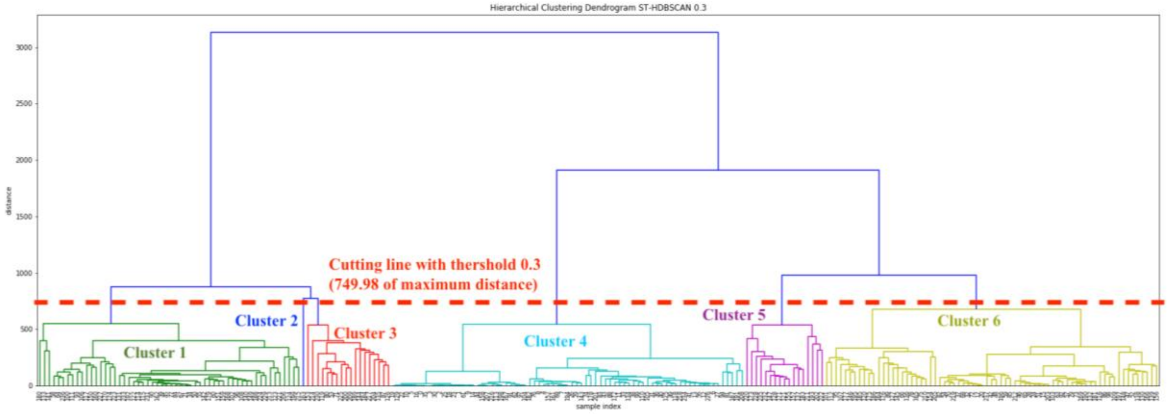

Fig. 6. Hierarchical Clustering Dendrogram ST-HDBSCAN 0.3.

On the other hand, Fig. 6 shows the cut point hierarchy for the threshold (0.3). It achieved a maximum distance of 749.98 and generated six clusters.

It is evident from Fig. 5 and 6 that the cluster colors are similar because the groups are the same; only the level of detail is different. Also, the result of the ST-HDSBCAN algorithm with the threshold (0.1) is more detailed. It is observed when the smaller cluster size and more significant cluster number in Fig. 5 compared with Fig. 6. However, the result of ST-HDBSCAN with a threshold (0.3) has interesting cluster patterns.

Furthermore, Fig. 7 shows the heatmap of ST-HDBSCAN with a threshold (0.3). It indicates how each cluster is distinguished. This pattern appears at a particular timeline, identified by ‘ClusterID 3’ in cyan color. This distinctive cluster occurred between 1994 and 1995. According to the Earth System Research laboratory, one of the National Oceanic and Atmospheric Administration research centers, the El-Niño that occurred within the above specified years is considered one of the top 24 strongest between 1895 and 2015. It is also regarded as one of the new El-Niño events that warmed the central–equatorial area of the Pacific Ocean rather than the eastern part of the Pacific Ocean.

It is evident from the heatmaps of ST-HDBSCAN with a threshold (0.3) that the clusters are spread across the timeline, except for Cluster 3. Cluster 1 (dark blue) mostly appears between 1990 to 1995 and starts to disappear after a few years. On the other hand, cluster 2 (light blue) appears close to Cluster 1 and spreads through the entire timeline. Cluster 4 (yellow color) appears mostly early in the timeline but starts to disappear between 1990 to 1995. It is replaced by Cluster 6 (brown color), although Cluster 4 reappears in the following years. Cluster 5 only appeared in 1990, and its effect on the environment was significant in later years. Cluster 6 appears in the early years of the timeline but mostly has a strong presence between 1990 to 1995. It gradually diminishes, and it completely vanishes in early 1997.

Fig. 8 shows the heatmap of the ST-HDBSCAN of threshold (0.1). The heatmap indicates a similar pattern to that of Fig. 7. Furthermore, the clusters are not as clear compared with the ST-HDBSCAN threshold (0.3). A cluster 1 to 6 mostly appears from 1990 to 1995, whereas Cluster 7 to 9 appears after. Clusters 1 to 6 shows a resurgence in 1998. This pattern is similar to Cluster 1 and 2 in ST-HDBSCAN of threshold (0.3).

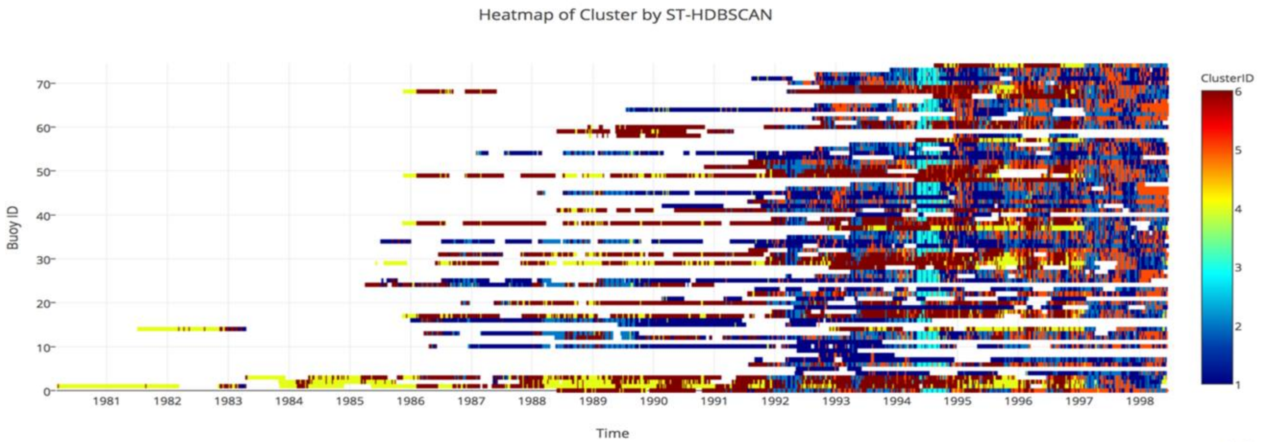

Fig. 7. Seven cluster 3 heatmaps of Cluster ST-HDBSCAN with Threshold (0.3).

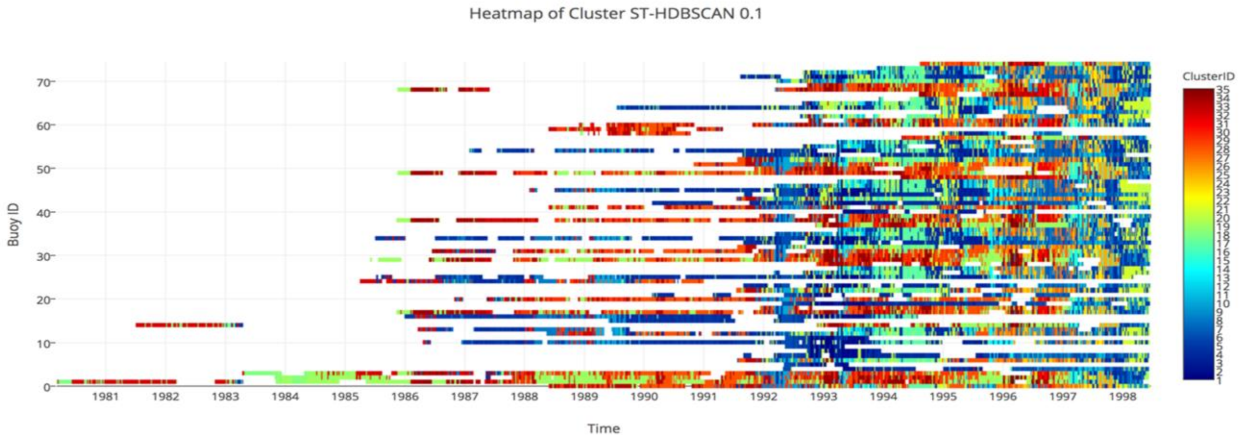

Fig. 8. Cluster heatmap of ST-HDBSCAN with Threshold (0.1).

Similar to Clusters 4 and 6 in ST-HDBSCAN of threshold (0.3), Clusters 16 to 19 start appearing early (1981) and are dominant from 1994 to 1995. Likewise, Clusters 28 to 30 appear early (1988), but they appear primarily between 1990 and 1994. They are replaced by Clusters 24 to 26 but reappear in early 1997 and are replaced again by Clusters 24 to 26. However, other clusters generally spread across the timeline and are not noticeable in the heatmap.

### *E. Performance Indices of the Proposed Method*

Table IV shows six performance indices for ST-HDBSCAN, ST-DBSCAN, and ST-DBSCAN with Eps 3. The high and low values differ according to the type of performance indices method, as indicated in column Best If Value. For example, in the performance index Ball-Hall, the higher value indicates the best performance, whereas Det Ratio is in direct contrast to it. The result specifies that ST-HDBSCAN with a threshold (0.3) has the best performance indices values for all performance indices except for KsqDetW. Table V specifies the comparison of the performance indices of ST-DBSCAN with ST-HDBSCAN 0.3. From Table V, it is evident that ST-HDBSCAN with 0.3 improves the performance indices of ST-DBSCAN by more than 40%. Similarly, the GDI51 and Trace W performance indices of ST-HDBSCAN 0.3 improve ST-DBSCAN by more than 100%.

The negative percentage of Det Ratio and Log Det Ratio were converted to positive values to calculate the average percentage improvement. This conversion is because a negative percentage indicates that the index is getting lower. Note that lower indices imply better performance indices. The average percentage improvement in the Dat Ratio index by ST-HDBSCAN 0.3 is 75%. Thus, adding the maximum temporal distance (*Eps 3*) and hierarchical aggregation using the ward's method improves all performance indices. However, this is not the case for the KsqDetW index.

The KsqDetW index value is lower because the number of clusters generated by ST-HDBSCAN (6 clusters) is smaller than that by ST-DBSCAN (14 clusters). Given that the function to evaluate the KsqDetW is highly dependent on the number of clusters, a smaller number of clusters results in lower KsqDetW values. Eq. 8 evaluates the value of KsqDetW, where K is the number of clusters and WG is the sum of the within-group scatter matrix.

$$KsqDetW = K^2 det(WG) \tag{8}$$

TABLE IV. PERFORMANCE INDICES COMPARISON OF ST-DBSCAN, ST-DBSCAN WITH EPS 3, ST-HDBSCAN 0.1 AND ST-HDBSCAN 0.3

| Indices | Best If Value | ST-DBSCAN | ST-DBSCAN Eps 3 | ST-HDBSCAN 0.1 | ST-HDBSCAN 0.3 |
|---|---|---|---|---|---|
| Ball-Hall | high | 0.01615323 | 0.0157 | 0.02011439 | **0.0230969** |
| Det Ratio | low | 14.4842 | 6.41 | 4.69 | **3.55** |
| GDI51 | high | 0.0331 | 0.0315 | 0.0779143 | **0.0944** |
| KsqDetW | high | 1.69E+08 | **1.11E+11** | 3.27E+09 | 1.27E+08 |
| Log Det Ratio | low | 476018.2 | 330972 | 275336.7 | **225691** |
| Trace W | high | 1904.39 | 2951 | 3386.383 | **3912** |

TABLE V. PERFORMANCE INDICES COMPARISON OF ST-DBSCAN AND ST-HDBSCAN 0.3

| Indices | Best If Value | ST-DBSCAN | ST-HDBSCAN 0.3 | Difference | Percentage |
|---|---|---|---|---|---|
| Ball-Hall | high | 0.01615323 | **0.0230969** | 0.00694367 | 43% |
| GDI51 | high | 0.0331 | **0.0944** | 0.0613 | 185% |
| KsqDetW | high | 1.69E+08 | **1.27E+08** | -42000000 | 25% |
| Trace W | high | 1904.39 | **3912** | 2007.61 | 105% |
| Det Ratio | low | 14.4842 | **3.55** | -10.9342 | 75% |
| Log Det Ratio | low | 476018.2 | **225691** | -250327.2 | 53% |

## V. CONCLUSION AND FUTURE WORK

The proposed ST-HDBSCAN effectively incorporates the spatial and non-temporal data aspects in clustering. Likewise, it utilizes the temporal aspect of data in the clustering process, thus significantly improving the clustering performance indices. The algorithm introduces an additional temporal distance boundary (*Eps 3*), known as the Three Neighborhood boundary. ST-HDBSCAN further improves the clustering performance by employing the hierarchical ward's method and fast dynamic time warping. We made comparisons with the ST-DBSCAN algorithm utilizing heatmaps and some performance indices. ST-HDBSCAN outperformed by 185% and indicated interesting patterns with spatio–temporal data. Several directions in future work are possible such as adapting the algorithm to process streaming data, indexing spatial and temporal aspects of data, and implementing parallel computing to improve the efficiency of the ST-HDBSCAN algorithm. Also, implementing the experiments for other spatio–temporal datasets in other fields such as criminology, medical and social media will also be beneficial to discovering hidden knowledge in the data.

## ACKNOWLEDGMENT

This work was supported by the Universiti Kebangsaan Malaysia under Grant FRGS/1/2019/ICT02/UKM/02/7.

## REFERENCES

[1] U. Khalil, O. Ahmed Malik, D. T. Ching Lai, and O. Sok King, “Cluster Analysis for Identifying Obesity Subrouops in Health and Nutritional Status Survey Data,” Asia-Pacific J. Inf. Technol. Multimed., vol. 10, no. 02, pp. 146–169, Dec. 2021, doi: https://doi.org/10.17576/apjitm-2021-1002-11.

[2] G. Atluri, A. Karpatne, and V. Kumar, “Spatio-Temporal Data Mining,” ACM Comput. Surv., vol. 51, no. 4, pp. 1–41, Jul. 2019, doi: https://doi.org/10.1145/3161602.

[3] A. Madraky, Z. A. Othman, and A. R. Hamdan, “Hair-oriented data model for spatio-temporal data representation,” Expert Syst. Appl., vol. 59, pp. 119–144, Oct. 2016, doi: https://doi.org/10.1016/j.eswa. 2016.04.028.

[4] V. Bogorny and S. Shekhar, “Spatial and Spatio-temporal Data Mining,” in 2010 IEEE International Conference on Data Mining, Dec. 2010, pp. 1217–1217. doi: https://doi.org/10.1109/ICDM.2010.166.

[5] J. D. Mazimpaka and S. Timpf, “Trajectory data mining: A review of methods and applications,” J. Spat. Inf. Sci., no. 13, Dec. 2016, doi: https://doi.org/10.5311/JOSIS.2016.13.263.

[6] G. Atluri, A. Karpatne, and V. Kumar, “Spatio-Temporal Data Mining: A Survey of Problems and Methods,” ACM Comput. Surv., vol. 51, no. 4, pp. 1–41, Jul. 2019, doi: https://doi.org/10.1145/3161602.

[7] H. Mahidin et al., “Spatio-temporal of surface ozone (O3) variations at urban and suburban sites in Sarawak region of Malaysia,” in IOP Conference Series: Earth and Environmental Science, Oct. 2021, vol. 880, no. 1, p. 012004. doi: https://doi.org/10.1088/1755-1315/880/1/012004.

[8] M. Tonini, M. G. Pereira, J. Parente, and C. Vega Orozco, “Evolution of forest fires in Portugal: from spatio-temporal point events to smoothed density maps,” Nat. Hazards, vol. 85, no. 3, pp. 1489–1510, Feb. 2017, doi: https://doi.org/10.1007/s11069-016-2637-x.

[9] N. Ahmed, M. A.-A. Hoque, B. Pradhan, and A. Arabameri, “Spatio-Temporal Assessment of Groundwater Potential Zone in the Drought-Prone Area of Bangladesh Using GIS-Based Bivariate Models,” Nat. Resour. Res., vol. 30, no. 5, pp. 3315–3337, Oct. 2021, doi: https://doi.org/10.1007/s11053-021-09870-0.

[10] U. M. Butt et al., “Spatio-Temporal Crime Predictions by Leveraging Artificial Intelligence for Citizens Security in Smart Cities,” IEEE Access, vol. 9, pp. 47516–47529, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3068306.

[11] B. Priambodo, A. Ahmad, and R. A. Kadir, “Spatio-temporal K-NN prediction of traffic state based on statistical features in neighbouring roads,” J. Intell. Fuzzy Syst., vol. 40, no. 5, pp. 9059–9072, Apr. 2021, doi: https://doi.org/10.3233/JIFS-201493.

[12] S. Kisilevich, F. Mansmann, M. Nanni, and S. Rinzivillo, “Spatio-temporal clustering,” in Data Mining and Knowledge Discovery Handbook, Boston, MA: Springer US, 2009, pp. 855–874. doi: https://doi.org/10.1007/978-0-387-09823-4_44.

[13] M. Y. Ansari, A. Ahmad, S. S. Khan, G. Bhushan, and Mainuddin, “Spatiotemporal clustering: a review,” Artif. Intell. Rev., vol. 53, no. 4, pp. 2381–2423, Apr. 2020, doi: https://doi.org/10.1007/s10462-019-09736-1.

[14] A. Madraky, Z. A. Othman, and A. R. Hamdan, “Analytic Methods for Spatio-Temporal Data in a Nature-Inspired Data Mode,” Int. Rev. Comput. Softw., vol. 9, no. 3, pp. 547–556, 2014.

[15] Z. A. Othman, A. A. Bakar, A. M. Adabashi, and Z. Muda, "A Similarity Normal Clustering Labelling Algorithm for Clustering Network Intrusion Detection," J. Appl. Sci., vol. 14, no. 10, pp. 969–980, 2014, doi: https://doi.org/10.3923/jas.2014.969.980.

[16] D. Birant and A. Kut, "ST-DBSCAN: An algorithm for clustering spatial–temporal data," Data Knowl. Eng., vol. 60, no. 1, pp. 208–221, Jan. 2007, doi: https://doi.org/10.1016/j.datak.2006.01.013.

[17] K. P. Agrawal, S. Garg, S. Sharma, and P. Patel, "Development and validation of OPTICS based spatio-temporal clustering technique," Inf. Sci. (Ny)., vol. 369, pp. 388–401, Nov. 2016, doi: https://doi.org/10.1016/j.ins.2016.06.048.

[18] S. Wang, T. Cai, and C. F. Eick, "New Spatiotemporal Clustering Algorithms and their Applications to Ozone Pollution," in 2013 IEEE 13th International Conference on Data Mining Workshops, Dec. 2013, pp. 1061–1068. doi: https://doi.org/10.1109/ICDMW.2013.14.

[19] S. Kisilevich, F. Mansmann, and D. Keim, "P-DBSCAN: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos," in Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research & Application - COM.Geo '10, 2010, p. 1. doi: https://doi.org/ 10.1145/1823854.1823897.

[20] Y. Zhang and C. F. Eick, "ST-DCONTOUR: a serial, density-contour based spatio-temporal clustering approach to cluster location streams," in Proceedings of the 7th ACM SIGSPATIAL International Workshop on GeoStreaming, Oct. 2016, pp. 1–4. doi: https://doi.org/10.1145/ 3003421.3003429.

[21] Y. Gong, R. O. Sinnott, and P. Rimba, "RT-DBSCAN: Real-Time Parallel Clustering of Spatio-Temporal Data Using Spark-Streaming," in Computational Science -- ICCS 2018, Cham: Springer International Publishing, 2018, pp. 524–539. doi: https://doi.org/10.1007/978-3-319-93698-7_40.

[22] M. Hüsch, B. U. Schyska, and L. von Bremen, "CorClustST—Correlation-based clustering of big spatio-temporal datasets," Futur. Gener. Comput. Syst., vol. 110, pp. 610–619, Sep. 2020, doi: https://doi.org/10.1016/j.future.2018.04.002.

[23] C. Choi and S.-Y. Hong, "MDST-DBSCAN: A Density-Based Clustering Method for Multidimensional Spatiotemporal Data," ISPRS Int. J. Geo-Information, vol. 10, no. 6, p. 391, Jun. 2021, doi: https://doi.org/10.3390/ijgi10060391.

[24] L. Ertöz, M. Steinbach, and V. Kumar, "Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data," in Proceedings of the 2003 SIAM International Conference on Data Mining, May 2003, pp. 47–58. doi: https://doi.org/10.1137/1.9781611972733.5.

[25] R. Oliveira, M. Y. Santos, and J. M. Pires, "4D+SNN: A Spatio-Temporal Density-Based Clustering Approach with 4D Similarity," in 2013 IEEE 13th International Conference on Data Mining Workshops, Dec. 2013, pp. 1045–1052. doi: https://doi.org/10.1109/ICDMW .2013.119.

[26] Earth System Research Laboratory, "Top 24 Strongest El Nino and La Nina Event Years by Season," 2015. https://www.esrl.noaa.gov/psd/ enso/climaterisks/years/top24enso.html (accessed Apr. 07, 2022).