Received <day> <Month>, <year>; | Revised <day> <Month>, <year>; | Accepted <day> <Month>, <year>

DOI: xxx/xxxx

ARTICLE TYPE

DYNAMITE: Dynamic Aggregation of Mutually-connected Points based Clustering Algorithm for Time Series Data

Abhimanyu Bhowmik[1] | Madhushree Sannigrahi[1] | Prithwish Guha[1] | Deepraj Chowdhury[2] | Sukhpal Singh Gill[3]

[1]Department of Artificial Intelligence, Amity University Kolkata, India
[2]Department of Electronics and Communication, IIIT Naya Raipur, India
[3]School of Electronic Engineering and Computer Science, Queen Mary University of London, State name, UK

Correspondence
Deepraj Chowdhary, Department of Electronics and Communication, IIIT Naya Raipur. Email: deepraj19101@iiitnr.edu.in

Present Address
Present address

Abstract

Time-series clustering typically entails clustering similar patterns across various time scales or comparing various point trajectories. However, this study emphasises to group data points based on their motions and forecast how the clusters will evolve across a number of immediate time frames. To achieve this, we propose a DYNamic Aggregation of Mutually-connected poInts clusTEring (DYNAMITE) based clustering algorithm for time series. DYNAMITE is based on the interaction between points in a time series and it majorly consists of three components: (1) cluster initialization; (2) calculation of mutually connected points; and (3) cluster updating.

KEYWORDS:
Evolutionary Clustering, Time Series Clustering, Unsupervised Learning

1 | INTRODUCTION

Clustering is a widely-known unsupervised machine learning algorithm. The idea behind clustering is to form several groups or clusters of unlabelled elements so that the data points of each cluster are similar to each other[1]. An ordered collection of values for a variable that were taken at predetermined intervals of time is known as a time series[2]. Time series analysis frequently involves forecasting future values in a time-series context as well as the recognition and application of patterns[3]. In this regard, clustering alludes to the grouping of historical time series data into meaningful sets of consecutive or non-consecutive points in order to detect trends or patterns. However, the study of a specific cluster, which can change over time, isn't given much attention in the literature. For grouping such changing time-dependent observations, we propose a DYNamic Aggregation of Mutually-connected poInts clusTEring (DYNAMITE) based clustering algorithm for time series. Despite the fact that 'DYNAMITE' is based on time-dependent observations, it is not necessary to use time-dependent sequences. If one is specifically interested in studying the evolution of clusters over a specified parameter, it is possible to accomplish this with our proposed method.'DYNAMITE' can be applied in instances such as clustering colonial bacteria growth, societal clustering of people over time or study galaxy/constellation formation by different stars and planets over time.

Our Contributions: The main contributions of this work are: i) Initiate an unconventional time series clustering problem,i.e., grouping dynamic observations, ii) Propose a new methodology 'DYNAMITE' based on dynamic aggregation of observations to address such problems and iii)'DYNAMITE' method can predict the clusters for immediate timeframes from the final stabilised[1] instance. Clustering over a period of time using classical methods like Gaussian mixture model (GMM), Density-based Spatial Clustering of Applications with Noise (DBSCAN)[4], and K-means, which cluster the observations in each time frame, adds computational complexity[5].'DYNAMITE' initiates with a classic DBSCAN model but minimises the time and computational complexity owing to its predictive nature across brief time instances after stabilisation of the model. Further, Section II confers the proposed framework and Section III concludes the paper and highlights future research directions.

[1]Stabilization is the condition of data points when they remain in the same position for specific variable frames. The user can adjust the value of the variable frame, which can be thought of as a hyperparameter of the algorithm (its minimum value is 2).

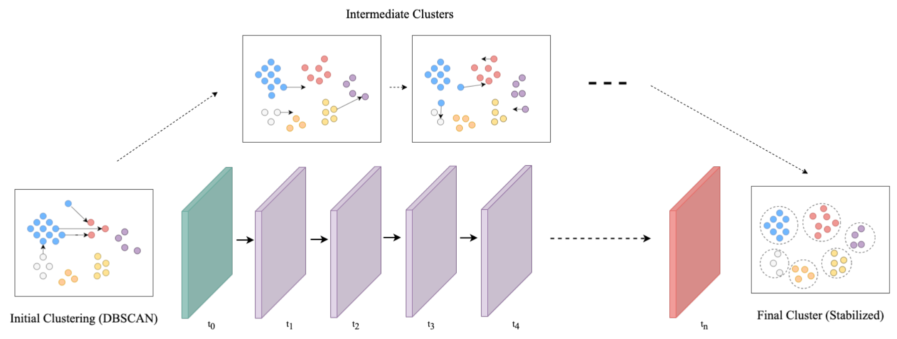

FIGURE 1 Proposed 'DYNAMITE' Model Architecture. The figure shows how the clustering evolved from the Initial Stage(t_0) to its Intermediate Form($t_1, t_2, ..., t_{n-1}$) and finally to the Stabilised Final Cluster (t_n).

2 | DYNAMITE: PROPOSED METHODOLOGY

'DYNAMITE' comprises 3 phases which are (1) initialization, (2) transition and (3) stabilisation. DBSCAN[6] is used to cluster the observations in the first phase (denoted as t_0 in Figure 1) since it can detect the number of clusters automatically. The Mutually-connected data points are determined in the second phase, and clusters are updated as $t_1 \rightarrow t_{n-1}$ (in Fig1). Stabilization is the last phase, which is denoted by t_n and occurs when the cluster stabilises and no notable changes occur. The detailed model is described below:

2.1 | Step I - Initialization

Each data point of the dataset is assigned to a specific cluster for $t = t_0$ using DBSCAN Clustering Algorithm. It divide the dataset into $k(K_1, K_2, K_3, ..., K_k)$ clusters on the basis of density. The density-based clustering algorithm[4] has the ability to cluster arbitrary shaped dataset in case of unknown data distribution. Due to its straightforward and effective features, the standard density-based clustering algorithm known as DBSCAN is frequently employed for data cluster analysis[6]. The DBSCAN algorithm divides each point into core, boundary and noise based on how densely it coexists with adjacent points. Core points are the ones that has at least minPts (threshold value) number of points (including the point itself) in its immediate vicinity with radius ϵ. If a point can be reached from a core point and there are less points in its immediate vicinity than minPts, it is a border point. Noise is a point that doesn't have any neighbours around. The core points surrounded by the boundary points, constitutes a valid cluster and the points are labelled as per their belonging to a cluster.

2.2 | Step II - Calculation of Mutually Connected Points

After the initial clustering, 'DYNAMITE' follows the following steps: (1) Define the velocity vector V_i for each of the data points, (2) Consider a communication range and (3) Calculate all the mutually connected points in accordance to the movement of point P_i in a dynamic scatter plot, at a specified time as portrayed in Figure 2 .

At a given time frame t, the position vector of any point in a *2-Dimensional plane* (P_i) is $x_{i_t}\hat{\imath} + y_{i_t}\hat{\jmath}$. To calculate the velocity vector of that point at t, the immediate prior time frame $t-1$ is taken into account , at which the position vector of the point P_i is $x_{i_{t-1}}\hat{\imath} + y_{i_{t-1}}\hat{\jmath}$. From the above equations, the velocity vector V_i of point P_i can be defined as:

$\vec{V_i} = x_{i_t}\hat{\imath}+y_{i_t}\hat{\jmath}-x_{i_{t-1}}\hat{\imath}+y_{i_{t-1}}\hat{\jmath} \Rightarrow \vec{V_i} = (x_{i_t}-x_{i_{t-1}})\hat{\imath}+(y_{i_t}-y_{i_{t-1}})\hat{\jmath} \Rightarrow \overrightarrow{V_{i_t}} = V_{x_{i_t}}\hat{\imath}+V_{y_{i_t}}\hat{\jmath}\ where, V_{x_{i_t}} = x_{i_t}-x_{i_{t-1}}, V_{y_{i_t}} = y_{i_t}-y_{i_{t-1}}$

Here, the directional vector $\overrightarrow{r_{ij_t}}$ from P_{i_t} to P_{j_t} can be represented as: $\overrightarrow{r_{ij_t}} = (P_{j_t} - P_{i_t}) = x_{j_t}\hat{\imath} + y_{j_t}\hat{\jmath} - x_{i_t}\hat{\imath} - y_{i_t}\hat{\jmath} = (x_{j_t} - x_{i_t})\hat{\imath} + (y_{j_t} - y_{i_t})\hat{\jmath} = r_{ij_x}\hat{\imath} + r_{ij_y}\hat{\jmath}$ where, P_{j_t} = Positional vector of point j at time $t(i \neq j)$. If the dot product[7] of velocity vector $\overrightarrow{V_{i_t}}$ and the directional vector $\overrightarrow{r_{ij_t}}$ is taken, we get:

$$\overrightarrow{V_{i_t}} \cdot \overrightarrow{r_{ij_t}} = V_{i_t} \cdot r_{ij_t} \cdot cos\phi_i \tag{1}$$

where ϕ_i refers to the angle inscribed between velocity vector of point P_i and the directional vector from point P_{i_t} to P_{j_t}. Equation 2 showcases the value of the angle ϕ_i. The *Communication Range* is conceptualised as an angle θ on either side of the vector V_i , as shown in Figure 3 (a). θ here acts as the boundary which facilitates the interraction between points. For every data point P_j falling in that *Communication Range*, it is checked whether point P_i falls under their communication range or not. If so, then all of these points are considered to be *Mutually Connected Data-points*.

If $\phi_i > \theta \Rightarrow$ point lies outside of the Communication Range.

Else $\phi_i \leq \theta \Rightarrow$ point lies inside the Communication Range.

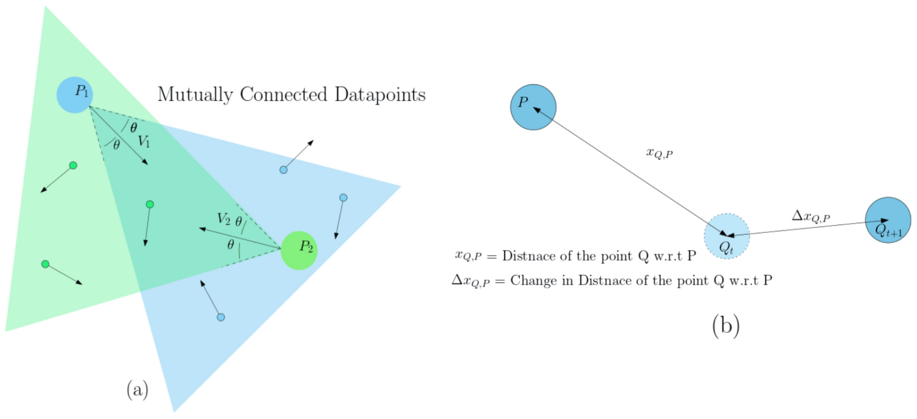
Mutually Connected Datapoints
P_1
θ
V_1
V_2
P_2
(a)
P
$x_{Q,P}$
$\Delta x_{Q,P}$
Q_t
Q_{t+1}
$x_{Q,P}$ = Distnace of the point Q w.r.t P
$\Delta x_{Q,P}$ = Change in Distnace of the point Q w.r.t P
(b)

FIGURE 2 (a) Visualisation of mutually connected data points in accordance to our proposed 'DYNAMITE' methodology. (b) Position of mutually connected data points in subsequent time frames.

Different communication ranges are obtained by adjusting the angle θ, which in turn, alters the clustering performance of the algorithm. All Mutually Connected Data-points can be achieve by checking $\phi_j \leq \theta$. By validating the condition in accordance with Equation 2 one can find all the *Mutually Connected points* for P_i at time frame t.

$$cos\phi_i = \frac{(x_{j_t} - x_{i_t})V_{x_{i_t}} + (y_{j_t} - y_{i_t})V_{y_{i_t}}}{\sqrt{V^2_{x_{i_t}} + V^2_{y_{j_t}}}\sqrt{(x_{j_t} - x_{i_t})^2 + (y_{j_t} - y_{i_t})^2}} \quad and,\ \phi_j = cos^{-1}\frac{(x_{i_t} - x_{j_t})V_{x_{i_t}} + (y_{i_t} - y_{j_t})V_{y_{i_t}}}{\sqrt{V^2_{x_{j_t}} + V^2_{y_{j_t}}}\sqrt{(x_{i_t} - x_{j_t})^2 + (y_{i_t} - y_{j_t})^2}} \tag{2}$$

For 3-Dimensional space, the velocity vector corresponding to point P_i at time t is: $\overrightarrow{V_{i_t}} = V_{x_{i_t}}\widehat{i} + V_{y_{i_t}}\widehat{j} + V_{z_{i_t}}\widehat{k}$ $where,$ $V_{x_{i_t}} = x_{i_t} - x_{i_{t-1}}$, $V_{y_{i_t}} = y_{i_t} - y_{i_{t-1}}$, $V_{z_{i_t}} = z_{i_t} - z_{i_{t-1}}$ and the directional vector $\overrightarrow{r_{ij_t}}$ from P_{i_t} to P_{j_t} can be represented as:
$\overrightarrow{r_{ij_t}} = (P_{j_t} - P_{i_t}) = (\overrightarrow{OP_j} - \overrightarrow{OP_i}) = r_{ij_x}\widehat{i} + r_{ij_y}\widehat{j} + r_{ij_z}\widehat{k}$ $where,$ $r_{ijx_t} = (x_{j_t} - x_{i_t})$, $r_{ijy_t} = (y_{j_t} - y_{i_t})$, $r_{ijz_t} = (z_{j_t} - z_{i_t})$

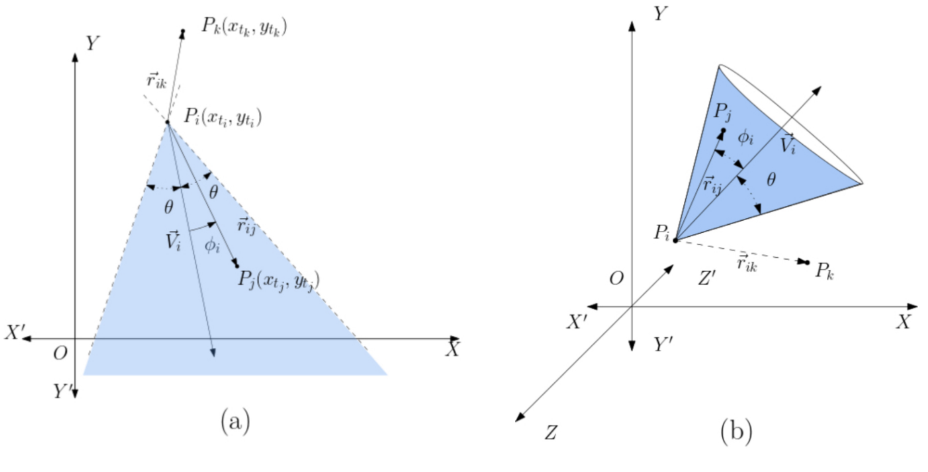

FIGURE 3 Graphical representation of Communication Range with region of interest in (a) 2-Dimension (b) 3-Dimension.

```
Algorithm 1 $[S_m, m_t] = CalMutualCP(P_i, P_j)$
    Input: $P_i, P_j(j \leftarrow (1 \rightarrow n))$    Output: Set of Mutually-connected points $m_t \leftarrow 0$ $S_m \leftarrow$ Set of Mutually-connected points
for $i \neq j$ do $\overrightarrow{V_{i_t}} \cdot \overrightarrow{r_{ij_t}} = V_{i_t} \cdot r_{ij_t} \cdot cos\phi_i$;    ▷ Equation 1
  if $\phi_i \leq \theta$ then $\overrightarrow{V_{j_t}} \cdot \overrightarrow{r_{ji_t}} = V_{j_t} \cdot r_{ji_t} \cdot cos\phi_j$;    ▷ Equation 2
    if $\phi_j \leq \theta$ then $S_m \leftarrow [S_m, P_j]$ $m_t = m_t + 1$
```

Therefore the angle ϕ_i, expressed between the directional vector $\overrightarrow{r_{ij_t}}$ from point P_{i_t} to P_{j_t} and velocity vector $\overrightarrow{V_{i_t}}$ of point P_i is shown in Equation 3. According to Figure 3 (b), a 3D cone[2] is considered around the velocity vector $\overrightarrow{V_{i_t}}$ with a semi vertical angle θ to be the *Communication Range* of P_{i_t}. To determine whether the point P_{j_t} falls inside this cone, check if $\phi_i \leq \theta$ or not. Similar to 2D plane, in 3D space, each point P_j falling inside the communication range of P_i is evaluated whether point P_i falls under their communication range or not. By calculating all such points all the *Mutually Connected Data points* for P_i at time frame t are obtained. It is achieved by checking if $\phi_j \leq \theta$ where,

$$cos\phi_i = \frac{\overrightarrow{V_{i_t}} \cdot \overrightarrow{r_{ij_t}}}{|\overrightarrow{V_{i_t}}| \cdot |\overrightarrow{r_{ij_t}}|}\ or,\ \phi_i = cos^{-1}\frac{r_{ij_{x_t}}V_{x_{i_t}} + r_{ij_{y_t}}V_{y_{i_t}} + r_{ij_{z_t}}V_{z_{i_t}}}{\sqrt{V^2_{x_{i_t}} + V^2_{y_{i_t}} + V^2_{z_{i_t}}}\sqrt{r^2_{ij_{x_t}} + r^2_{ij_{y_t}} + r^2_{ij_{z_t}}}}\ and,\ \phi_j = cos^{-1}\frac{r_{ji_{x_t}}V_{x_{j_t}} + r_{ji_{y_t}}V_{y_{j_t}} + r_{ji_{z_t}}V_{z_{j_t}}}{\sqrt{V^2_{x_{j_t}} + V^2_{y_{j_t}} + V^2_{z_{j_t}}}\sqrt{r^2_{ji_{x_t}} + r^2_{ji_{y_t}} + r^2_{ji_{z_t}}}} \tag{3}$$

Similarly for n-Dimensional space, the velocity and directional vector is represented as:

$$\overrightarrow{V_{i_t}} = \begin{bmatrix} V_{x1_{i_t}} \\ V_{x2_{i_t}} \\ \cdots \\ V_{xn_{i_t}} \end{bmatrix} and\ \overrightarrow{r_{ij_t}} = \begin{bmatrix} r_{ijx1_t} \\ r_{ijx2_t} \\ \cdots \\ r_{ijxn_t} \end{bmatrix} \quad where,\ V_{x_{n_{i_t}}} = x_{n_{i_t}} - x_{n_{i_{t-1}}}\ and\ r_{ij_{x_{n_t}}} = x_{n_{j_t}} - x_{n_{i_t}} \tag{4}$$

Therefore, the angle ϕ_i can be represented as in equation 5. Here, the points within Communication Range of point P_i are determined by checking whether $\phi_i \leq \theta$. For every such point P_j, condition $\phi_j \leq \theta$ is verified, which in turn provides all the *Mutually-Connected Data-points* in n-Dimensional space. Equation 5 demonstrates the value of ϕ_j.

$$\phi_i = cos^{-1}\frac{\sum_{n=1}^{n} V_{x_{n_{i_t}}} r_{x_{n_{ij_t}}}}{\sqrt{\sum_{n=1}^{n} V^2_{x_{n_{i_t}}}}\sqrt{\sum_{n=1}^{n} r^2_{x_{n_{ij_t}}}}}\quad and,\quad \phi_j = cos^{-1}\frac{\sum_{n=1}^{n} V_{x_{n_{j_t}}} r_{x_{n_{ji_t}}}}{\sqrt{\sum_{n=1}^{n} V^2_{x_{n_{j_t}}}}\sqrt{\sum_{n=1}^{n} r^2_{x_{n_{ji_t}}}}} \tag{5}$$

2.3 | Step III - Updation of cluster :

Let m_{t_i} represents all the mutually connected data points for each specific point P_i at time frame t, and then take into consideration a distribution based on distance, starting with the closest; where point P_i and point P_j are separated distance: $r_{ij} = \sqrt{\sum_{n=1}^{n} r^2_{x_{n_{ij_t}}}}$.

The *Distance of Socialisation* may be defined as the distance from the origin up to the median (i.e. $r_{ij} \leq M_i$ where M_i = Median of all the distances r_{ij}) of the distribution since these are the social connections that have the greatest influence at that specific point.

Weights Initialization : All points up to the Distance of Socialization have their weights initialised using any *monotonically decreasing function*,as Exponential ($\lambda_i e^{-nr_{ij}}$), Algebraic ($\lambda_i \frac{1}{1+r^n_{ij}}$), Logarithmic ($\lambda_i \ln \frac{nr_{ij}+e}{nr_{ij}+1}$), and Trigonometric ($\lambda_i \tan \frac{\pi}{n|r_{ij}|+4}$) where n= Number of dimension, r_{ij} = Distance between ith and jth observation in relation to the distance from the corresponding points. This Weight initialization is shown in Figure 4 . Since it depicts the velocity at which the data points are travelling at time instant t, λ is inversely proportional to the length of the directional vector V_i as in equation 6.

$$\lambda_i \propto \frac{1}{|V_i|} \Rightarrow \lambda_{\mathrm{i}} = \frac{\lambda_0}{|V_i|} \tag{6}$$

where, λ_0 = *Social Strength Index* is a hyperparameter provided by the user $\lambda_0 \in [0, 1]$. Social Strength Index determines whether the observations are *Sociophilic*[3] or *Sociophobic*[4] in nature. For *Sociophobic* instances, $\lambda_0 \rightarrow 0$. Contrasted with it, for *Sociophilic* instances, $\lambda_0 \rightarrow 1$.

Consider the initialised weights w_j with regards to their distances and sum up all the values for each particular cluster K_n, as $W_{P_i}(cluster = K_n) = \sum_{n=1}^{k} w_j(K_n)$. Then we determine the probability p_{P_i} of a point P_i for being in a particular cluster K_n as demonstrated below [Eq. 7]:

$$p_{P_i}(K_n) = \frac{W_{P_i}(cluster = K_n)}{\sum_{n=1}^{k} W_{P_i}(cluster = K_n)} \tag{7}$$

[2]When a 2D angle is rotated with respect to a central axis, it generates a 3D cone. Thus, a cone has been considered to determine a 3D communication range.
[3]In Sociophilic observations, the distance between observations heavily influences the clusters.
[4]Sociophobic clusters are indifferent of the distance between observations.

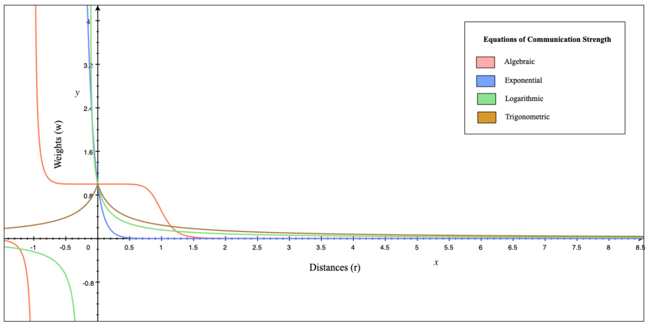

FIGURE 4 Weights initialization according to different monotonically decreasing functions.

The point P_i is assigned it to the particular cluster with maximum probability [Eq 8], and the cluster information for all data points P_i is then updated accordingly.

$$\arg\max_n (p_{P_i}(K_n)) \tag{8}$$

2.4 | Step IV - Weights Updation

Moving on to the next time frame t_{i+1} , for all the socially connected points of a point P_i , we scrutinize whether they are still *mutually connected* or not [as in Fig. 2]. Further we check if the velocity of the point in the next time instance is same or not.

If Yes: The change in distance between the data points Δx is assessed and the weights are updated as follows:

when $\Delta x > 0$:

$$w_{new} \leftarrow w_{old} - \alpha \left|\frac{\Delta x}{x}\right| \tag{9}$$

when $\Delta x < 0$:

$$w_{new} \leftarrow w_{old} + \alpha \left|\frac{\Delta x}{x}\right| \tag{10}$$

when Δ x $= 0$:

$$w_{new} \leftarrow w_{old} \tag{11}$$

where α is *interaction parameter* $\in [0, 1]$.

Else: The previous mutually connected points are discarded and weights of the new points are initialized from the initial weight distribution.

The weight-bias updation formulae used in Artificial Neural Networks (ANN)[8] heavily impact the weight update as seen in equations 9 10 11. *The interaction parameter*(α) acts similar to the learning rate in ANN methodology which can be fine-tuned by the user depending on their datasets. In Fig. 2 (b), the displacement of point Q from Q_t to Q_{t+1} has varying influence(weight) with respect to point P. This variation is introduced in the model using the interaction parameter(α).

If: Displacement of point Q has negligible influence on its weight over point P, $\alpha \approx 0$

Else if: Displacement of point Q has high influence on its weight over point P, $\alpha \approx 1$

Step - IV is repeated for the time frame t_{i+1} and subsequent others with updated weights until the dataset stabilizes, The change in number of *Transitioning Points*[5] $\leq \epsilon$, where ϵ is a predefined threshold provided by user (Stabilization of Observations).

3 | CONCLUSION AND FUTURE WORKS

In this paper, the transition of a cluster of points over time was discussed and proposed a novel method to predict how the clusters will evolve across a number of immediate time frames. Interactions between observations according to their distances is the foundation of the methodology. As a result, compared to conventional clustering algorithms like DBSCAN, K-means, and GMM, the clusters may be predicted for a brief period of time when the data points become stable, hence, reducing time and computational complexity. The model has a number of hyperparameters that can be fine tuned allowing the model to be optimised for the specific environment. 'DYNAMITE' methodology can be applied in any parametric scenario to track how the cluster evolves in relation to the parameterized variable. To sum up, we can conclude that the theoretical formulation of the aforementioned model enables the recommended technique to anticipate cluster evolution. In future work, 'DYNAMITE'

[5]The points changing its location in immediate time frames.

Algorithm 2 $K_n = ClustUpdate(S_m, w_j, K_{p_i})$

```
    Input:Set of $S_m$, $w_j$, Initial Clusters    Output: Updated cluster  $w_j \leftarrow Communication\ strength$; $n \leftarrow No.\ of\ observations$
$K_{P_i} \leftarrow Initial\ cluster\ of\ P_i$  $S_{m_i} \leftarrow S_m\ of\ P_i$;    ▷ $S_m$ from Algorithm 1
for $i \leftarrow (1 \rightarrow n)$ do
    for $j \in S_{m_i}$ do
$r_{ij} = |\overrightarrow{r_{ij}}|$; $R_{ij} \leftarrow [R_{ij}, r_{ij}]$   $M_i \leftarrow Median\ of\ R_{ij}$
        while $r_{ij} \leq M_i$ do $W_{P_i}(K) \leftarrow \sum w_j(K)$;
$K_n \leftarrow \max(W_{P_i}(K))$;    ▷ Equation 8 $K_{P_i} \leftarrow K_n$
```

Algorithm 3 $K_T = WeightUpdate(\epsilon, K_t, S_m)$

```
    Input: $\epsilon, K_t, S_m$    Output: Final Cluster  $\epsilon \leftarrow Threshold\ value$   $N_t \leftarrow No.\ of\ Transitioning\ points$   $K_t \leftarrow Cluster\ of\ pt.$
$P\ at\ t$  $n \leftarrow No.\ of\ observation$
while $N_t \leq \epsilon$ do
    for $i \leftarrow (1 \rightarrow n)$ do $S_{m_{t+1}} \leftarrow S_m\ of\ P_i\ for\ (t+1)$;    ▷ Algorithm 1
        for $j \neq i$ do
            if $j \in S_{m_{t+1}}$ and $V_{j_{t+1}} = V_{j_t}$ then $\Delta x \leftarrow Displacements\ of\ point\ P_j$  $w_j \leftarrow w_{new}$;  ▷ According to Equations 9 10 11
            else $w_j \leftarrow Communication\ strength$;  $K_{t+1} \leftarrow$ ClustUpdate($S_{m_{t+1}}, w_j, K_t$);    ▷ Algorithm 2
            if $K_t \neq K_{t+1}$ then $N_{t+1} \leftarrow N_{t+1} + 1$   $N_t \leftarrow N_{t+1}$
```

can be implemented this entire theoretical methodology as a Python module to test it on some real world datasets. Further, the following open issues could be explored[5]: (1) identifying the datasets and use-cases where the model performs well. (2) Real-time training of 'DYNAMITE' will be tested in future so that the created model should have the provisions for handling situations in which new clusters arise while the model is being run. (3) the explainability and interpretability of the 'DYNAMITE' can be experimented for better detailing of the algorithm[5].

References

1. Clément P, Bouleux G, Cheutet V. Improved Time Series Clustering Based on New Geometric Frameworks. *Pattern Recognition* 2021: 108423. doi: 10.1016/j.patcog.2021.108423

2. Che Z, Purushotham S, Cho K, Sontag D, Liu Y. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 2018; 8(1): 1–12.

3. Liu LM, Bhattacharyya S, Sclove SL, Chen R, Lattyak WJ. Data mining on time series: an illustration using fast-food restaurant franchise data. *Computational Statistics & Data Analysis* 2001; 37(4): 455–476.

4. Ester M, Kriegel HP, Sander J, Xu X, others . A density-based algorithm for discovering clusters in large spatial databases with noise.. In: . 96. ; 1996: 226–231.

5. Gill SS, al. e. AI for next generation computing: Emerging trends and future directions. *Internet of Things* 2022; 19.

6. Lafabregue B, Weber J, Gançarski P, Forestier G. End-to-end deep representation learning for time series clustering: a comparative study. *Data Mining and Knowledge Discovery* 2022; 36(1): 29–81.

7. Athreya A, Fishkind DE, Tang M, et al. Statistical inference on random dot product graphs: a survey. *The Journal of Machine Learning Research* 2017; 18(1): 8393–8484.

8. Courbariaux M, Bengio Y, David JP. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems* 2015; 28.

□