DISCOVERING CLUSTERING PATTERNS FROM E-COUNTER DATA STREAMS

N.E. Ivari[a] , M. Wachowicz [b,*], P.A.H. Williams [c], T. Agnew [d]

[a] University of New Brunswick, Department of Geomatics Engineering, Fredericton, Canada - nasrin.eshraghi@unb.ca
[b] RMIT University, Geospatial Science School, Melbourne, Australia - monica.wachowicz@rmit.edu.au
[c] Flinders University, College of Medicine and Public Health, Adelaide, Australia - trish.williams@flinders.edu.au
[d] Flinders University, College of Nursing and Health Sciences, Adelaide, Australia - tamara.agnew@flinders.edu.au

Commission IV, WG IV/9

KEY WORDS: Smart Campus, Internet of Things, Stream Clustering

ABSTRACT:

Clustering data streams has gained popularity in recent years due to their potential of generating relevant information for planning building automation, evaluating energy efficiency scenarios, and simulating emergency protocols in indoor spaces. In this paper, a Data Stream Affinity Propagation (DSAP) clustering algorithm is proposed for analyzing indoor localization data generated from e-counters systems. The data streams are a sequence of potentially infinite and non-stationary data points, arriving continuously where random access to the data is not feasible and storing all the arriving data is impractical. The DSAP model is developed based on a two-phase approach (i.e., online and offline clustering phases) using the landmark time window model. It is non-parametric in the sense of not requiring any prior knowledge about the number of clusters and their respective labels. The validation and performance of the DSAP model is evaluated using real-world data streams from a behavioural experiment aimed at finding new spatio-temporal patterns in stair usage due to an educational intervention campaign.

1. INTRODUCTION

Indoor localization technologies have been explored for generating location data of people in indoor spaces, including WiFi, BLE beacons, RFID tags, visible light wave, and ultra-wideband (Namiot, 2015). In particular, infra-red, optical (e.g., RGB videos or flat images), break beam, thermal, and ultrasonic sensors have been used for counting people in indoor spaces (Mautz, 2012). One simple way of counting people is using ultra-wideband radar sensors mounted in e-counter devices that can be used for counting multiple people passing through a passage or a wide door. Two sensors equipped with antennas which have narrow beam width are used to form two invisible electronic layers in the path. These layers are used for sensing the presence of a person and recognizing the direction of the movement.

The stream data generated from e-counters can be considered as a sequence of infinitive, ordered, and fast-changing data points $d_1, d_2, ..., d_i$ that are arriving continuously (which requires an online clustering phase through the data points) where random access to the data points is not feasible and storing all the arriving data points is impractical (which requires an offline clustering phase) (Han et al., 2011). Therefore, the traditional set-up where an entire static data set is available for clustering can not be applied to data streams (Toshniwal, 2013)

Time window models including sliding, damped, landmark, and pyramidal, are also needed to develop stream clustering models (Nguyen et al., 2015). These time windows aim to handle the spatio-temporal distribution of the data streams. Selecting a time window makes it possible to analyze and store a stream within a specific time frame and discard the previous historical data (Mansalis et al., 2018).

The affinity propagation method has been chosen for analyzing indoor localization data in this research work. It is a partitioning-based message passing algorithm that treats every data point as a potential centroid, and each point is given equal importance (Delbert, 2009). Also, the AP algorithm does not require the number of clusters as an input which makes any continuously operating automated operation more robust, especially if it is implemented in a cloud environment. Overall, there are a few robust streaming algorithms based on the Affinity Propagation clustering method. The clustering process usually requires a strategy capable of continuously partitioning stream data points while taking into account restrictions of memory and time. Table 1 summarizes the main characteristics of four streaming algorithms that were proposed in previous research work.

Algorithms	Year	Time window model
StrAP	2008	Sliding (online)
IStrAP	2012	Sliding (online)
ISTRAP	2018	Sliding (online)
APDenStream	2013	Sliding (online) Pyramidal (offline)

Table 1: Overview of streaming AP clustering algorithms

This paper proposes a novel Data Stream Affinity Propagation (DSAP) model using the landmark time window for clustering data streams obtained from indoor e-counters. The DSAP model is capable of supporting the online-offline phases. In the online phase, micro-clusters are constantly computed using a landmark time window to handle the most recent data points in the stream and to continuously follow the changing data distribution. The offline phase is then performed, and the micro-clusters themselves are clustered to provide the overall clustering results. The entire online and offline phases that deliver the final clusters are performed without any user intervention.

To the best of our knowledge, no research work on indoor localization data streams using a streaming AP algorithm and the landmark time window model has been previously proposed in the literature. Out of these previously proposed algorithms, none

*Corresponding author

of them have explored the landmark time window before. Mainly because StrAP, IsrAP, ISTRAP, and APDenStream take the opportunity to use a repository to emulate the sliding time window model in the online phase.

Our proposed DSAP algorithm is actually a simplified approach in comparison to IStrAP (Zhang et al., 2008), ISTRAP (Li and Li, 2012), and APDenStream (Zhang et al., 2013). Our research premise is that indoor localization data streams have a unique time-evolving cluster structure that does not require update strategies such as outlier repository, decay function, and recurrence judgement rules to handle outliers, mainly because the constrained indoor spaces make the occurrence of outliers extremely low. To the best of our knowledge, StrAP, IsrAP, ISTRAP, and APDenStream have not yet been applied for indoor localization data streams. Unfortunately, the programming codes of these algorithms were not provided by their authors to us, hampering their evaluation in terms of finding streaming clusters from indoor localization data.

We demonstrate the potential of applying DSAP to find new insights into stair usage in indoor spaces. The experimental results validate the robustness of the DSAP model in handling dynamically evolving data streams based on intrinsic validation indices and performance evaluation metrics.

2. DATA STREAM AFFINITY PROPAGATION

The DSAP model has been developed to overcome the clustering limitations such as detecting abrupt and gradual changes as soon as they occur, and distinguishing drift from noise. It consists of three main phases described as follows:

- **Clustering Phase:** computes micro-clusters using a landmark time window model to harvest data streams. It also computes macro-clusters by re-clustering all the centroids of the micro-clusters found in each time window. The goal is to discover hidden dense clusters structures from indoor localization data streams.
- **Validation Phase**: assesses the quality of the clustering results when there is no ground truth label of data. The selected metrics are silhouette index, Caliński-Harabasz index, and Davies-Bouldin index (scikit-learn developers, 2020). The focus is to assess between-clusters dispersion and inter-cluster dispersion for all clusters.
- **Performance Evaluation Phase:** estimates the efficiency of the DSAP algorithm using the time and space complexity metrics. The aim is to find the right balance between memory consumption and the speed of execution for the DSAP algorithm.

The DSAP source code in Python programming language is freely available at https://github.com/nasrineshraghi/DSAP. The version of Python used is 3.7. Spyder 3.3.6 IDE which is part of Anaconda 2019. Main libraries applied in Python are scikit-learn, pandas, numpy, matplotlib, datetime, scipy, psutil, itertools and skmultiflow. Figure 1 summarizes the DSAP model discussed in this section. The three phases are depicted with their main steps and respective tasks.

2.1 Clustering Phase

The DSAP algorithm is based on the online and offline clustering phases. The online micro-cluster phase detects newly arriving data points and updates the historical clusters accordingly. The offline macro-cluster phase generates summary clusters from the centroids of the micro-clusters.

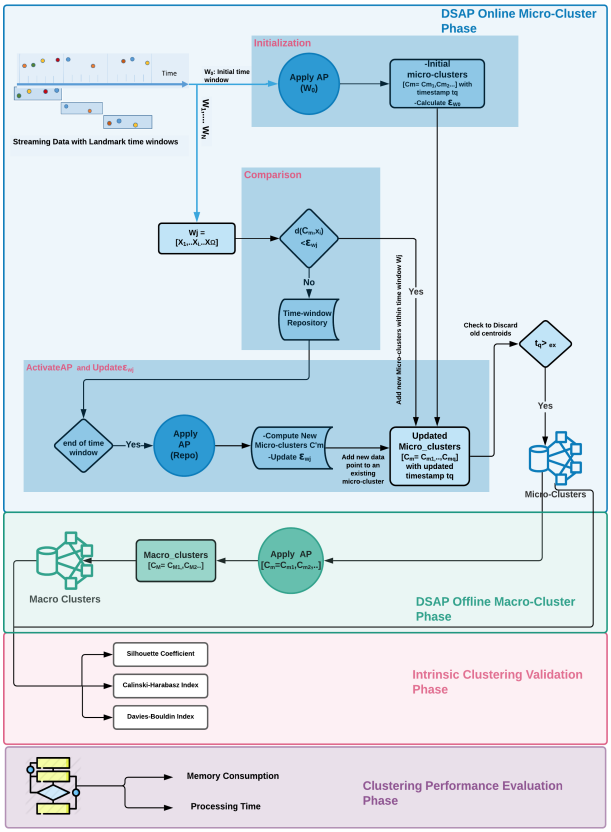

Figure 1: Flowchart of the proposed DSAP model

2.1.1 Online Micro-Clustering The online micro-cluster phase initializes by receiving data streams as a continuous sequence of data points that are harvested using the landmark time window model. Therefore, the specifications for creating a landmark time window are its start time and landmark duration. In practice, the landmark duration can be defined by using a landmark time interval (e.g. every hour) or by a landmark event (e.g. every 100 data points). Every time a new window is created, all the data points from the previous windows are discarded. All data points within the window have the same weights, which can allow us to generate hourly, daily, or monthly summary comparisons.

This online micro-cluster phase consists of three major steps: initialization, comparison, and activateAP with update ϵ_{W_j}.

Initialization Step

In the initialization step, the data stream is ingested in a windowed manner using the landmark time window model $W_0, W_1, ...W_j$ where each window W_j contains Ω data points and L is defined as the time interval of window $W_j = [x_1, x_2..., x_i, ..x_\Omega]$.

The clustering process is initiated by assigning the first sequence of data points from a data stream to the initial landmark time window W_0. The landmark time interval is a-priori determined based on user requirements, data rate, and expected latency of the expected data streams. For example, if high latency is expected in harvesting the data points, a long duration for the time intervals is advised. In contrast, in processing data from high rate streams, short time intervals will improve the clustering process.

The AP algorithm is applied to the all data points belonging to the first time window W_0 for computing the initial micro-clusters and their respective centroids $C_m = C_{m_1}, C_{m_2}, ..C_{m_q}$. The clusters

and data points produced by the DSAP algorithm are represented as sequence of tuples:

$$[s_1 = (C_{m_1}, N_1, t_1)], [s_2 = (C_{m_2}, N_2, t_2)],$$
$$..., \left[s_q = \left(C_{m_q}, N_q, t_q\right)\right]$$

where C_{m_q} is the centroid of a micro-cluster q, N_q is the number of data points assigned to the cluster q, and t_q is the last timestamp assigned to the micro-cluster q.

All data points within the initial landmark time window are considered as a potential exemplar until a robust set of centroids and their respective micro-clusters are found by computing the four AP matrices, i.e. similarity matrix, responsibility matrix, availability matrix and criterion matrix. The hyperparameters such as the preference parameter, the damping factor, the maximum number of iterations, and the convergence iteration are also set up during this phase. Their values will be constant throughout all the landmark time windows in the online micro-cluster phase. The preference value keeps as a median of the similarity matrix, and the AP algorithm automatically calculates it each time. The damping factor is in the range of 0.5 to 1, and based on data, it varies. It can be obtained by testing different values. The rest of the hyperparameters are kept as default values.

A new dynamic threshold ϵ_{W_j} is introduced in the DSAP model for incrementally computing the clusters as time passes by, and as a result, allowing the analysis to take into account the changes of recurrently occurring clusters. For the initial time window W_0, an intra-cluster distance matrix is calculated between all data points and their respective centroids in each cluster using the Euclidean distance function. The mean of all these intra-cluster distances gives the initial value of the ϵ_{W_0} threshold that will be used in the next step. The outputs of the initialization step are a set of micro-clusters centroids (C_m) and an initial ϵ_{W_0}.

Comparison Step

As new data points arrive, the DSAP model allocates them into their respective new time windows $W_j = [x_1, x_2..., x_i, ..x_\Omega]$ using the same landmark time interval L as defined in the previous step. For each new data point, a set of tasks are devised as follows:

- Compute the Euclidean distances between each new data point x_i and the current micro-cluster centroids C_{m_q} using Equation 1.

$$d\left(C_{m_q}, x\right) = \sqrt{\sum_{1}^{n}\left(x_i - C_{m_{q_i}}\right)^2} \quad (1)$$

  where C_{m_q} is a current micro-cluster centroid, x is a new data point within the current landmark time window; x_i and $C_{m_{q_i}}$ are Euclidean vectors starting from the origin; and n space.

- Compare the computed Euclidean distances against the current ϵ_{W_j}. If one of the computed distance values is less than the current ϵ_{W_j} threshold (i.e. $min(d([C_m], x_i)) < \epsilon_{W_j}$), this new data point will be straightforwardly placed within a current existing cluster. The DSAP model merges the new data point x_i with the closest existing cluster by updating the time t_q and adding to the number of data points N_q values in the cluster centroid tuple C_{m_q}. However, if all computed distance values are higher than the current ϵ_{W_j} threshold, the new points will be saved in the time-window repository temporarily until the landmark time window ends, and later be used in the next ActivateAP and Update step.

With the arrival of new data points, x_i and x_j, the intra-cluster distances T and T' are computed in a time window W_1. In this case, the T distance is less than the current ϵ_{W_1}, and the new data point is merged with the existing cluster C_{m_i}. In contrast, the computed distance T' is greater than the current the current ϵ_{W_1} threshold, and this new data point will be saved in the time-window repository until the end of the streaming for the time window W_1. These tasks are performed every time a new landmark time window is created and new data points arrive.

The outputs of the comparison step are the set of micro-clusters from the previous window, but now with new data points associated with some of the centroids, and an updated timestamp corresponding to the chosen centroids. Additionally, n number of data points whose Euclidean distance from the centroids was greater than ϵ_{W_j} are temporarily kept in the time-window repository.

ActivateAP and Update ϵ_{W_j}

All the data points that have been temporarily accumulated during the execution of the previous step are used to compute new micro-clusters C'_m using the AP algorithm once again. These new clusters will be added to the existing clusters C_m that belong to their respective landmark time windows. As a result, the updated C_m = C'_m + C_m is achieved. The new micro-clusters will have their respective timestamps associated with them.

The next task is to compute the new ϵ_{W_j} by taking into account the centroids of the new micro-clusters found within a particular landmark window. This is accomplished by computing the mean of the intra-cluster distances using the new micro-clusters. The update ϵ_{W_j} is computed by applying the mean between the new and the current ϵ_{W_j}. The updated ϵ_{W_j} is then used when the forthcoming data points from a new landmark time window arrive.

In order to avoid the number of centroids spreading beyond control, old centroids that are no longer deemed relevant are removed. To do this, an expiration time hyperparameter e_x is introduced. This value is applied to forget centroids that have not been selected in the recent windows as quantified by e_x. The e_x number can be chosen to include all the time windows from the start or just the last few windows, depending on the user requirements. All the clusters whose associated timestamps are older than e_x window lengths ($e_x \times L$) from the current time are considered obsolete and hence discarded.

2.1.2 Offline Macro-Cluster Phase The offline macro-cluster phase in DSAP starts after N time windows have passed. In this phase, all the micro-clusters C_m generated during the online micro-cluster phase are re-clustered using the AP algorithm to generate k number of macro-clusters $C_M = (C_{M_1}, C_{M_2}, ..., C_{M_k})$. This process is usually not considered time critical, and the number of macro-clusters is expected to be less than the number of micro-clusters. The hyperparameters related to the AP algorithm, such as the maximum number of iterations and the convergence iteration, are default values. The preference parameter is set to median the similarity matrix of all the micro-clusters obtained from the online phase, and the damping factor is a fixed number for this phase.

2.2 Validation Phase

Intrinsic clustering validation uses the internal information of the clustering process to evaluate the goodness of a clustering structure when the ground truth labels are unknown. The well-known metrics Silhouette Index (S), Caliński-Harabasz index (CH), and Davies-Bouldin index (DBI) were selected to validate the goodness of macro and macro clusters found by the DSAP algorithm.

2.2.1 Silhouette Index The silhouette index was introduced by (Rousseeuw, 1987) on the premise of the silhouette width of a data point to measure how similar a data point is to its own cluster compared to other clusters. The silhouette index S_i for a data point $i \in x$ in cluster $C_k \in C$ is calculated as follows.

$$a_i = \frac{1}{C_k} \sum_{j \in C_k, i \neq j} d(i,j) \tag{2}$$

$$b_i = \min_{C_m \in C, C_m \neq C_k} \frac{1}{C_m} \sum_{j \in C_m, i \neq j} d(i,j) \tag{3}$$

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} \tag{4}$$

The first parameter a is the average distance between the sample and all the others in the same class, and the second parameter b is the mean distance between the sample and all the other points in the next closest cluster. Negative S_i scores for a point i means that the data point is in the incorrect cluster. Positive scores show a robust and dense clustering. Moreover, values around zero indicate overlapping clusters. If the S_i has a higher value, it means a greater ratio between b_i as compared with a_i, causing the data point i to be more similar to the other data points in the same cluster.

2.2.2 Caliński-Harabasz Index It is also known as the variance ratio criterion, which is used to score dense and well-separated clusters. A recent comparative study of available clustering indices demonstrated this index as one of the best cluster validity indices (Arbelaitz et al., 2013).

Well defined clusters yield high values of this index. Therefore, the maximum value of the index is used to select the best partition. For n data points, k clusters where B and W are the between within cluster scatter matrices, the index is computed as (Caliński and Harabasz, 1974):

$$CH = \frac{traceB/(k-1)}{traceW/(n-k)} \tag{5}$$

where

$$B_k = \sum_{q=1}^{k} n_q (c_q - c_i)(c_q - c_i)^T \tag{6}$$

$$W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T \tag{7}$$

with C_q the set of points in cluster q, c_q the center of cluster q, c_D the center of D, and n_q the number of points in cluster q.

The CH is computationally efficient to calculate, but just like the silhouette index, the CH index is biased towards convex clusters, assigning them higher values than the density-based clusters such as those obtained through DBSCAN.

2.2.3 Davies-Bouldin Index The Davies-Bouldin Index (DBI) was introduced in 1979 by (Davies and Bouldin, 1979), and it computes the average similarity between each cluster C_i for $i = 1, 2, ..., k$ and its most similar one C_j. For this index, similarity is defined as a measure R_{ij} that optimizes s_i (i.e., the average distance between each data point in the cluster and the centroid of that cluster; and d_{ij} (i.e., the distance between cluster centroids i and j). A simple choice to construct R_{ij} is that of non-negative and symmetric as follows:

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{8}$$

Then the Davies-Bouldin index is defined as:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij} \tag{9}$$

With this index, a minimum value denotes the best partitioning of the data. The average similarity calculation is much simpler than the computations required for calculating the silhouette index. Like the previous two metrics, DBI too suffers from a preference towards convex clusters as compared to density based clusters owing to the usage of centroid distance that limits the distance metric to Euclidean space.

2.3 Performance Evaluation Phase

Data stream clustering brings about a few unique challenges as compared to traditional data clustering. The data streams are a continuous flow of data points that arrive at a rapid rate. Random access to these data points is not possible and the volatility of the data streams sometimes limits to having a single look at the data points upon their arrival. To evaluate the efficiency of the DSAP, two metrics are proposed: computational time and the memory consumed to run the DSAP algorithm. These metrics used in this phase are described in the next sections.

2.3.1 Time Complexity The time complexity of the DSAP algorithm is affected by three steps in online phase and one step in offline phase. For a data set size N, divided into time windows of size L, the complexity for the initialization step would be $O(L^2 logL) + O(L)$, for AP and threshold calculation on L points. In comparison step, Euclidean distances are calculated for the subsequent L point in the next time window leading to complexity of $O(L)$. If the time window repository has T data points, then the activateAP step will have a time complexity of $O(T^2 logT) + O(T)$ for the second instance of AP and threshold calculation. Finally C_m micro clusters are clustered by AP whose time complexity is $O(C_m^2 logC_m)$. As a result, total worst case complexity of the algorithm is $O(L^2 logL) + O(L) + N/L(O(L) + O(T^2 logT) + O(T)) + O(C_m^2 logC_m)$, which is affected by the initial window size, the number of points in the repository and the number of micro clusters. It is expected that for highly dynamic data set, the execution time would be higher since more number of micro-clusters will be generated.

2.3.2 Space Complexity Similar to the time complexity, the memory consumption quantifies the amount of memory taken by the DSAP algorithm to run as a function of the length of the input. That means how much memory, in the worst case, is needed at any point by the algorithm. The same big-O notation is used to describe the space complexity as well. This parameter represents the algorithm's scalability and performance. In simple terms, it gives the worst-case scenario of an algorithm's growth rate.

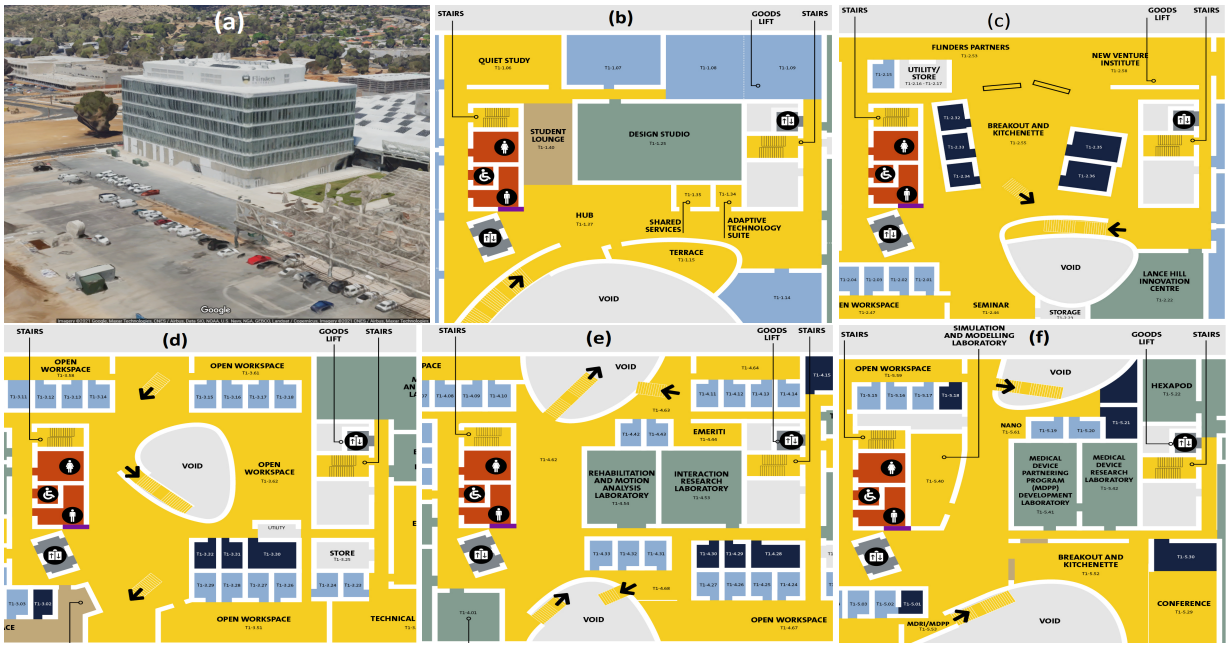

Figure 2: (a) Tonsley building view from the north-side parking lot, (b)-(f) Tonsley Building layout of levels 1-5

3. BEHAVIOURAL INTERVENTION EXPERIMENT

The experiment was performed in the Tonsley building at the campus of Flinders University, Australia in 2019. This study was conducted to observe the potential impact of motivational and educational interventions on the physical activity of people with sedentary lifestyles. Tonsley Building is a six-story building populated by students and staff with entrances facing north and south directions. The building floor layouts are illustrated in Figure 2. The north-side entrance faces a small barricaded entrance, and the main entrance is on the south side adjacent to the covered Tonsley garden. The building is fitted with lifts that reach all the levels and have stairs on the north and south sides that connect all the levels. Additionally, the building has an open space called the void and has central stairs that connect all the levels. Thus, each level, stairs are labeled as level $L_{x+1} - L_x$ North/South/Central, where $L_x = 1, 2, .5$.

E-counter sensors were deployed at all stairs. They are made by the International Road Dynamics company and consist of a pair of transmitter and receiver(transmitter: PTx20-1 and receiver: PRx20W1). The transmitter sends an infrared message to the receiver; when the infrared light is interrupted by someone passing through it, a count is registered. If no one passes, then these sensors log a count value of zero every 10 minutes. Each stair has two sensors, UP/DOWN that determine a person's direction of travel up and down the stairs. Signals from the sensors are sent to wireless hubs in real-time and then from the hubs to the server. The maximum count value for each direction is about 1,000,000.

The data were collected from February 21, 2019, until July 16, 2019, but only three months of this data set were used for the experiment from March 18 to June 23, 2019. The sensors captured movement for all 24 hours and all days continuously. The raw e-counter data streams consist of approximately 668,000 data points, out of which almost half the data points were used in our research work.

During the first month period starting on March 18, 2019, data were collected to establish a baseline for staircase usage by making measurements as unobtrusively as possible. The mid-semester break from April 15-28 was excluded from the study.

Motivational and educational interventions to increase stair usage have taken place from April 29 to May 26, 2019, including digital screens that displayed health information and live feedback throughout the building, especially near the lifts. After the intervention month, more observations have been gathered from May 27 to June 23, 2019 to study the impact of the intervention campaign on stair usage patterns.

During the experiment, hubs were temporarily switched off or sensors fell off, and some special events took place such as public holidays, term breaks, and fire drills which affected the number of people using the stairs. For example, on May 1, at 11:15 a.m, the fire alarm went off, and everyone had to evacuate the building using the stairs. Furthermore, if no one passes through the beam of sensors, the sensors indicate zero during that specific interval. A huge number of zeros were logged during the experiment especially after working hours, weekends, and holidays, and as a result, they were removed from the clustering analysis.

Finally, the sensor location and people count features were selected for computing the clusters. The hyperparameters for the AP in the DSAP algorithm were consistent and fixed for the entire experiment. The damping value was set to be equal to 0.96, maximum iteration was 100, and preference was equal to the median of the similarity matrix.

4. DISCUSSION OF THE RESULTS

The streaming cluster analysis was aimed at finding any evolutionary patterns in stair usage due to an educational intervention campaign. The clustering results were generated using three months of the experiment, which were named as before, during, and after the intervention.

4.1 Overall Stair Usage Patterns

Figure 3 top panel illustrates the daily stair usage over the three months named as before intervention (highlighted in purple), during the intervention (highlighted in blue), and after intervention (highlighted in yellow). It is quite apparent that more people preferred taking the stairs to go down rather than up. Expected low

Figure 3: Top Panel:Total number of people using the stairs: before intervention month (purple), during intervention month (blue), and after intervention month (yellow). Bottom panel: Total number of people using the stairs: before intervention month (purple), during intervention month (blue), and after intervention month (yellow)

usage of stairs is also observed on the weekends. The observations also revealed that Levels 4-3 North and 5-4 North were used the least. This could be due to fewer people using the north entrances since the north entrance leads to a low capacity parking lot.

Although the overall trend of the number of people using the stairs at each level remains consistent for the three months, one peak of stair usage occurred on Monday, April 1, before the intervention, another peak occurred on Friday May 10, during the intervention and happened again on Tuesday, June 11 after the intervention campaign has ended. After a closer inspection of the data, it was found that on May 10, a group of students went up around 12:06 pm, and presumably, the same group came down at 12:23 pm. This led to an increase in the measured count across many levels. No particular event was found to justify other peaks.

Evolutionary patterns were also observed on an hourly basis for each month of the experiment, as shown in Figure 3, bottom panel. The usage peak is around noon for most of the weekdays, most probably due to the lunch break. The people count is asymmetric on both sides of the peak, with more people using the stairs on Tuesday afternoons during all months of the experiment. Overall the month before the intervention campaign seems to have recorded the highest usage than the following months. By looking at this trend, our preliminary hypothesis is that the intervention campaign has not generated the expected impact on motivating people to take the stairs. Therefore, the clustering results play a significant role in testing this hypothesis.

4.2 Micro-cluster Evolution

The evolution of micro-clusters was interesting to be analyzed in order to detect any concept drift during the clustering process. Figure 4 shows the centroids of the micro-clusters and their respective data points that are connected using straight lines. These results illustrates a few patterns found on Wednesday, May 1. The one-hour expiration time was chosen to ensure that only the data points within the one-hour window were used for finding clusters. The micro-clusters show an increase in activity starting around 9-10 am at the lower levels of the building. In contrast, usage clusters were found at the peak around noon that were located at different levels of the building.

As the day progresses, the number of micro-clusters have gradually decreased. These generated patterns were observed throughout the experiment, demonstrating the potential of using the DSAP approach for finding and tracking meaningful micro-clusters over space and time. The strength of DSAP lies in its ability to continuously find micro-clusters by responding quickly to any changes in the stream data.

4.3 The impact of the intervention

Figure 5 illustrates the occurrence of the weekly patterns at different levels of the building, revealing that the intervention campaign had a positive impact on people taking the stairs between level 3-2 and level 4-3 south. In contrast, the campaign has also shown to have a weak impact in changing the behaviour of people when taking the stairs between level 4-3 north and level 5-4 north. Finally, the intervention campaign has not increased the usage of stairs between level 2-1. No anomalous clusters were found in these results.

4.3.1 Validation and Performance Metrics For the DSAP evaluation, the AP model was used as a baseline to compare the performance and validation metrics. We have selected AP because other stream AP clustering algorithms and their codes were not available at that time. The same weeks used for computing the macro-clusters shown in Figure 5 were used in this evaluation.

The results for each of the metrics, along with the number of clusters, processing time, and memory consumption, are summarized

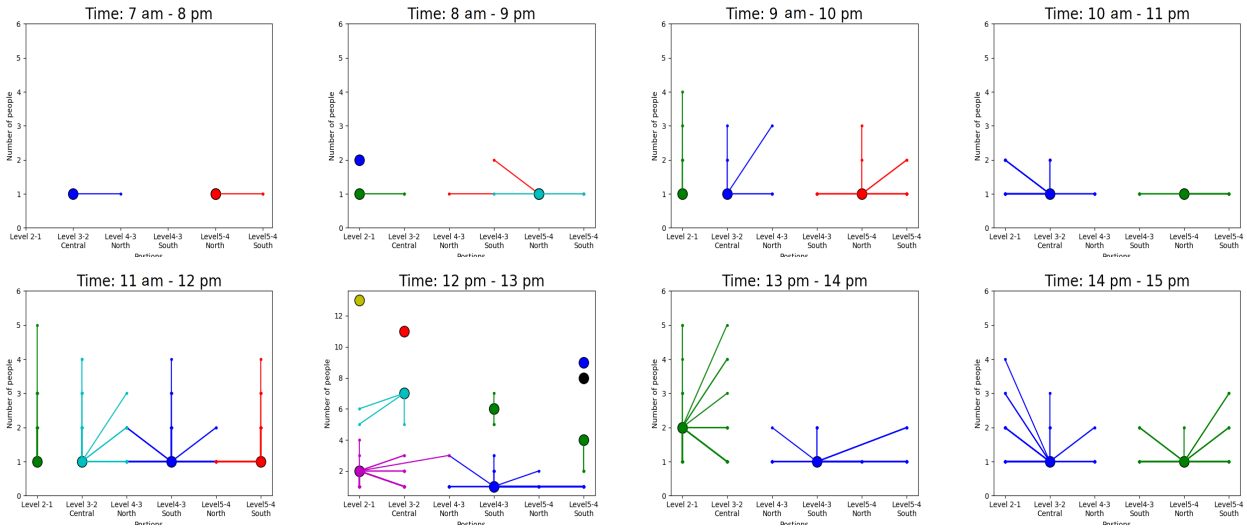

Figure 4: The evolution of micro-clusters

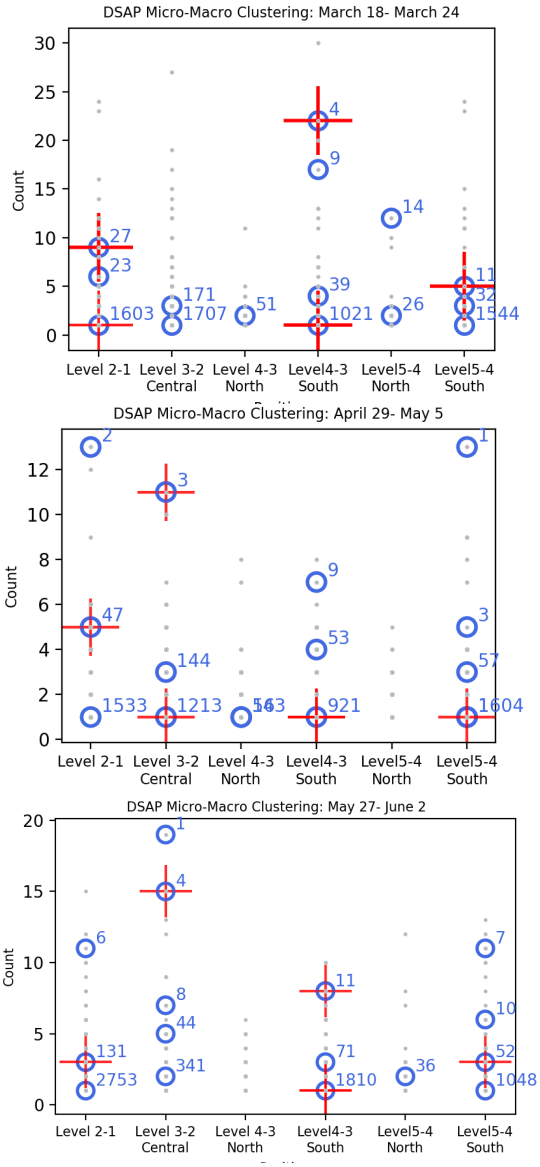

Figure 5: Clustering results for the first week before, during, and after the intervention campaign

in Table 2. A significant saving in processing time is immediately apparent between the DSAP algorithm and AP. The processing time here includes the time taken for all the phases and steps of the DSAP algorithm as it analyzes the entire data using time windows. For a real stream, the execution time of interest would be the time taken to execute one window and is expected to be much lower as in DSAP. The initial window calculation is the most time-intensive step but is performed only once and does not affect the current time window.

The accuracy of the two methods is similar, but the micro-clusters generated by DSAP show slightly improved values, as shown by the validation metric values. The number of micro-clusters generated by DSAP are similar to clusters generated by AP suggesting that the activity levels remain consistent.

The intrinsic validation metrics, Silhouette and Davies-Bouldin indices for the micro-clusters have similar values to the numbers for AP clusters. The number of micro-clusters is the same order as the macro-clusters, hence, some of the validation metrics are not accurate. It should be noted that the number of points represented by a cluster significantly affects some of these metrics, and hence internal evaluation metric numbers need to be carefully interpreted.

4.4 Conclusions and Future Research Work

In this research work, the DSAP model was developed for uncovering evolutionary patterns based on two phases: online micro and offline macro phases. The landmark time window model was proposed to ensure that only the latest data points in the stream are used in the clustering process.

The main advantage of the DSAP model relies on its simple and straightforward approach but still retaining the strengths of other streaming AP-based algorithms (i.e., StrAP, IStrAP, ISTRAP, and APdenstream) while removing some of the complexities introduced by them. The combination of a user-defined expiration time, a dynamic threshold ϵ_{W_j}, the landmark time window model, and a time-window repository have been crucial to developing a manageable, fast and accurate clustering algorithm. DSAP runs entirely autonomously without the need to specify the number of clusters once the hyperparameters are selected.

Due to stream data availability, the DSAP model was not implemented using live streaming data. Therefore, the main limitation

of the DSAP model is that it does not handle latency and bandwidth problems, which can occur very often when analyzing live streaming data. Stream data clustering of multiple variables is also expected to bring scaling issues. Highly dynamic data with large concept drift is expected to reduce the performance of the DSAP algorithm as new clusters will be formed in almost every window slowing down the entire process. These limitations are planned to be addressed in future versions of the algorithm.

Metrics	AP Algorithm	DSAP
Before Intervention Week (March 18 - March 24)		
Processing Time (s)	22	0.6
Memory Consumption (MB)	536	250
Number of Clusters	14	micro = 15 macro = 5
Silhouette Index	0.4	micro = 0.7 macro = 0.4
Caliński-Harabasz Index	2009	micro = 4872 macro = 87
Davies-Bouldin Index	0.5	micro = 0.8 macro = 0.5
Intervention Week (April 29 - May 5)		
Processing Time (s)	25	0.6
Memory Consumption (MB)	536	256
Number of Clusters	11	micro = 16 macro = 5
Silhouette Index	0.5	micro = 0.5 macro = 0.4
Caliński-Harabasz Index	2421	micro = 6292 macro = 35
Davies-Bouldin Index	0.7	micro = 1 macro = 0.2
After Intervention Week (May 27 - June 2)		
Processing Time (s)	26	0.6
Memory Consumption (MB)	541	260
Number of Clusters	12	micro = 16 macro = 5
Silhouette Index	0.4	micro = 0.6 macro = 0.4
Caliński-Harabasz Index	1278	micro = 4381 macro = 40
Davies-Bouldin Index	0.5	micro = 0.5 macro = 0.5

Table 2: Overall results from the performance evaluation and clustering validation

ACKNOWLEDGEMENTS

This research was funded by the NSERC/Cisco Industrial Research Chair, Grant IRCPJ 488403-1.

REFERENCES

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M. and Perona, I., 2013. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1), pp. 243–256.

Caliński, T. and Harabasz, J., 1974. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1), pp. 1–27.

Davies, D. L. and Bouldin, D. W., 1979. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2), pp. 224–227.

Delbert, D., 2009. *Affinity propagation: clustering data by passing messages*. Citeseer.

Han, J., Pei, J. and Kamber, M., 2011. *Data mining: concepts and techniques*. Elsevier.

Li, L. and Li, X., 2012. An improved online stream data clustering algorithm. In: *2012 Second International Conference on Business Computing and Global Informatization*, IEEE, pp. 526–529.

Mansalis, S., Ntoutsi, E., Pelekis, N. and Theodoridis, Y., 2018. An evaluation of data stream clustering algorithms. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 11(4), pp. 167–187.

Mautz, R., 2012. Indoor positioning technologies. *IEEE Internet of Things Journal*, pp. 1–127.

Namiot, D., 2015. On indoor positioning. *International Journal of Open Information Technologies*, 3(3), pp. 23–26.

Nguyen, H.-L., Woon, Y.-K. and Ng, W.-K., 2015. A survey on data stream clustering and classification. *Knowledge and information systems*, 45(3), pp. 535–569.

Rousseeuw, P. J., 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20, pp. 53–65.

scikit-learn developers, B. L., 2020. *Clustering performance evaluation*.

Toshniwal, D., 2013. Clustering techniques for streaming data-a survey. In: *2013 3rd IEEE International Advance Computing Conference (IACC)*,, IEEE, pp. 951–956.

Zhang, J.-P., Chen, F.-C., Liu, L.-X. and Li, S.-M., 2013. Online stream clustering using density and affinity propagation algorithm. In: *2013 IEEE 4th International Conference on Software Engineering and Service Science*,, IEEE, pp. 828–832.

Zhang, X., Furtlehner, C. and Sebag, M., 2008. Data streaming with affinity propagation. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*,, Springer, pp. 628–643.