TEMPORAL DECISION MAKING USING UNSUPERVISED LEARNING

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

OMAR IBRAHIM

Dr. James Keller, Dissertation Supervisor

May 2021

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

TEMPORAL DECISION MAKING USING UNSUPERVISED LEARNING

presented by Omar Ibrahim,

a candidate for the degree of Doctor of Philosophy,

and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. James Keller

Dr. Marjorie Skubic

Dr. Derek Anderson

Dr. Mihail Popescu

To my grandparents, and my uncles in loving memory.

ACKNOWLEDGEMENTS

I would like to thank my mentor Dr. James Keller who has given much time, effort and knowledge to aid in the completion of this dissertation. I still remember the moment when I asked him to be my advisor and in a completely new area for me, he was encouraging and supporting from that moment. Thank you for your patience and friendship. He continues to be a source of inspiration and I could not have asked for a better adviser.

My sincere thanks to Dr. Mihail Popescu for providing me the opportunity to work on his grants, which along with providing me funding, helped me gain important work experience. You have been of immense help. I am grateful to Dr. Marjorie Skubic for her guidance and the opportunity to work with a great interdisciplinary group at the Center for Eldercare and Rehabilitation Technology. I am also thankful to Dr. Derek Anderson for being part of My Ph.D committee. Special thanks to Dr. James Bezdek for the collaboration and the support. You really inspired me and learned from him what dedication means. I would like to thank all my friends who made my time at Mizzou delightful. I will miss our weekly meetings with a group that made me feel home even when home was far away.

I cannot begin to express my gratitude to my family for all of the love, support, encouragement and prayers they have sent my way along this journey. To my parents, thank you for being my motivation and encouragement. Your unconditional love and support has meant the world to me, I hope that I have made you proud. Special and profound thanks to my brother and sisters who offered invaluable support and humor over the years. I am so thankful for your love and support over the years. To my children, Awab, Jumanah, and Raihanah, you are my inspiration to achieve greatness and the fuel of my life. Without you, I would not be where I am today. I love you to the moon and back. And to my lovely wife, thank you for being there for me always when I needed you. Your support, motivation, and encouragement is what has gotten me through when I wanted to give up. Without you, this was not possible. My uncles, Aunts, and Relatives your encouragement and prayers is what kept me working hard to make you proud. Words can't express how grateful and thankful I am to all my family. Without you guys, I most certainly would not be where I am today. Finally, I am grateful to HCED for providing me the financial support during my graduate studies. Thank you for this opportunity.

Table of Contents

Table of Figures

ABSTRACT

With the explosion of ubiquitous continuous sensing, on-line streaming clustering continues to attract attention. The requirements are that the streaming clustering algorithm recognize and adapt clusters as the data evolves, that anomalies are detected, and that new clusters are automatically formed as incoming data dictate. In this dissertation, we develop a streaming clustering algorithm, MU Streaming Clustering (MUSC), that is based on coupling a Gaussian mixture model (GMM) with possibilistic clustering to build an adaptive system for analyzing streaming multi-dimensional activity feature vectors. For this reason, the possibilistic C-Means (PCM) and Automatic Merging Possibilistic Clustering Method (AMPCM) are combined together to cluster the initial data points, detect anomalies and initialize the GMM. MUSC achieves our goals when tested on synthetic and real-life datasets. We also compare MUSC's performance with Sequential k-means (sk-means), Basic Sequential Clustering Algorithm (BSAS), and Modified BSAS (MBSAS) where MUSC shows superiority in the performance and accuracy.

The performance of a streaming clustering algorithm needs to be monitored over time to understand the behavior of the streaming data in terms of new emerging clusters and number of outlier data points. Incremental internal Validity Indices (iCVIs) are used to monitor the performance of an on-line clustering algorithm. We study the internal incremental Davies-Bouldin (DB), Xie-Beni (XB), and Dunn internal cluster validity indices in the context of streaming data analysis. We extend the original incremental DB (iDB) to a more general version parameterized by the exponent of membership weights. Then we illustrate how the iDB can be used to analyze and understand the performance of MUSC algorithm. We give examples that illustrate the appearance of a new cluster, the effect of different cluster sizes, handling of outlier data samples, and the effect of the input order on the resultant cluster history.   In addition, we investigate the internal incremental Davies-Bouldin (iDB) cluster validity index in the context of big streaming data analysis. We analyze the effect of large numbers of samples on the values of the iCVI (iDB).

We also develop online versions of two modified generalized Dunn's indices that can be used for dynamic evaluation of evolving (cluster) structure in streaming data. We argue that this method is a good way to monitor the ongoing performance of online clustering algorithms and we illustrate several types of inferences that can be drawn from such indices. We compare the two new indices to the incremental Xie-Beni and Davies-Bouldin indices, which to our knowledge offer the only comparable approach, with numerical examples on a variety of synthetic and real data sets.

We also study the performance of MUSC and iCVIs with big streaming data applications. We show the advantage of iCVIs in monitoring large streaming datasets and in providing useful information about the data stream in terms of emergence of a new structure, amount of outlier data, size of the clusters, and order of data samples in each cluster. We also propose a way to project streaming data into a lower space for cases where the distance measure does not perform as expected in the high dimensional space.

Another example of streaming is the data acivity data coming from TigerPlace and other elderly residents' apartments in and around Columbia. MO. TigerPlace is an eldercare facility that promotes aging-in-place in Columbia Missouri. Eldercare monitoring using non-wearable sensors is a candidate solution for improving care and reducing costs. Abnormal sensor patterns produced by certain resident behaviors could be linked to early signs of illness. We propose an unsupervised method for detecting abnormal behavior patterns based on a new context preserving representation of daily activities. A preliminary analysis of the method was conducted on data collected in TigerPlace. Sensor firings of each day are converted into sequences of daily activities. Then, building a histogram from the daily sequences of a resident, we generate a single data vector representing that day. Using the proposed method, a day with hundreds of sequences is converted into a single data point representing that day and preserving the context of the daily routine at the same time. We obtained an average Area Under the Curve (AUC) of 0.9 in detecting days where elder adults need to be assessed. Our approach outperforms other approaches on the same datset. Using the context preserving representation, we develoed a multi-dimensional alert system to improve the existing single-dimensional alert system in TigerPlace. Also, this represenation is used to develop a framework that utilizes sensor sequence similarity and medical concepts extracted from the EHR to automatically inform the nursing staff when health problems are detected. Our context preserving representation of daily activities is used to measure the similarity between the sensor sequences of different days. The medical concepts are extracted from the nursing notes using MetamapLite, an NLP tool included in the Unified Medical Language System (UMLS). The proposed idea is validated on two pilot datasets from twelve Tiger Place residents, with a total of 5810 sensor days out of which 1966 had nursing notes.

Chapter 1: Introduction

1.1 Problem Statement

This work encompasses several years of research into algorithms for temporal data stream analysis to produce decisions as data points evolve. It addresses the problem of streaming clustering of a temporal data stream where the data arrives one sample at a time. Streaming clustering algorithms need to be incremental, process each new arriving data points only once and discard it, adapt as data stream evolves, flag outliers, and most importantly, spawn new emerging structures. To make a decision from a streaming clustering algorithm's output, the performance of the algorithm needs to be monitored in real time. The monitoring process should provide information about number and size of clusters in the data stream, amount of outlier data samples, streaming order of incoming data points, and emerging clusters in the data stream. Incremental Cluster Validity Indices (iCVIs) is one possible approach to monitor streaming clustering algorithms. Traditional cluster validity indices (CVIs) cannot be applied directly to work with streaming clustering algorithm because they require partitions produced from static datasets. Therefore, CVIs must be extended to work online and incrementally as data samples evolve.

Synthetic and real-life streaming datasets are used in this dissertation. The real-life datasets are: Weather dataset [1], heron Iceland dataset [2], and eldercare datasets from elderly residents' apartments such as TigerPlace, an eldercare facility that promotes aging-in-place, [3]. For the first two datasets, the data stream are numeric features which are used to test the streaming clustering and the iCVIs. TigerPlace data is a combination of motion sensors and bed sensor data. The existing alert system in TigerPlace is one dimensional, and hence, treats the data from each sensor independently. Our proposal is that by combining more than one alert parameter, we can find more of the potential alerts we are missing; and by using streaming clustering, iCVIs, and other adaptive models, we can capture more predictive alerts and more customized alerts that can help us detect early signs of illness (health decline). The idea is to develop a new representation on the sensor data to measure the similarity among different days and find days when the resident needs to be assessed. This data captures the resident activity, but it also captures the movement of care givers, family visits, and hardware issues. Therefore, these issues should be taken into account when developing the features representation. Streaming clustering and other adaptive models can be applied after coming up with a suitable distance measure.

1.2 Motivation

In recent years, new ways of continuously collecting data have been introduced, in large part due to the advances in the field of hardware technology. Similarly, developments in information technology have allowed large flows of data across IP networks. Every day use of credit cards, phones or browsing the web lead to a large data volume that may be impossible to store on a disk. These kinds of online data are referred to as data streams or streaming data. Moreover, even if the data can be stored locally, the size of the incoming samples may be large which makes it impractical to process an individual record more than once. Therefore, traditional data mining algorithms such as classification, clustering, and frequent pattern mining become more challenging to apply in these situations. It can be even more difficult from an algorithmic and computational point of view in cases where the data patterns may evolve continuously [4]. Hence, it is crucial in the design of the mining algorithms to take into account changes in underlying structure of the data stream.

For streaming and temporal datasets, decisions need to be made in an online fashion as the data stream evolves with time.

Clustering is a data mining technique that is commonly used to look for structure in an unlabeled dataset. Traditional clustering algorithms require access to the whole dataset to find prototypes. These algorithms normally iterate multiple times over the whole dataset to converge on stable cluster representation. Such approaches are infeasible for applications where size of a dataset is large either in size or dimension, or when it arrives in a streaming fashion.

Streaming clustering is an approach that tackles the issue of large and streaming datasets. Traditional clustering algorithms must be modified, or new algorithms designed for use in streaming data applications. Online clustering algorithms can be subdivided into two groups. The first group is general online clustering algorithms which can be applied to any sequential dataset. Examples of algorithms in this group are sequential k-means (sk-means) [5] and sequential agglomerative clustering [6]. Batch clustering is another example in this category which uses a window of the data stream and clusters data samples in that window using traditional clustering algorithm. Clustering results of adjacent windows are combined to get the final clustering results [7, 8, 9]. These algorithms require the number of clusters to be specified a priori and do not assume any ordering (temporal behavior) in the data stream. The second group of online clustering algorithms assume some natural ordering in the data and rely on the assumption that close observations in time are closely related in the feature space, such as the case with time series. This assumption is used to dynamically create clusters in evolving data streams. Online clustering algorithms from the second category have two main parts: (1) a change detection mechanism that helps the algorithm identify new emerging structures in the data, (2) an adaptive model for the clusters. The online clustering algorithm in this dissertation falls under the umbrella of the second category of online clustering algorithms.

The behavior of streaming algorithms must be monitored as data samples evolve to track outliers and to create new structures. Even for low dimensional streaming datasets, the monitoring procedure is not feasible if data points arrive over a long period, or at a rapid rate. Therefore, coming up with techniques to monitor streaming clustering algorithms is a crucial part of streaming data analysis. In [10], the idea of incremental Cluster Validity Indices (iCVIs) was introduced using incremental cluster validity analysis. Davies-Bouldin (DB) [11] and Xie-Beni (XB) [12] indices were extended to work incrementally as data samples evolve. However, the interpretation of iCVIs results were not investigated in a way that could provide information about the streaming clustering algorithm. In other words, a basic question is whether the iCVIs values asses the clustering result quality or do they reflect the behavior of the algorithm. Furthermore, DB and XB indices were the only ones extended to work incrementally, whereas there are other indices that can be extended and studied, such as Dunn's indices, the Partition Coefficient, and the Exponential Separation (PCAES) index.

The second part of this research is to use the MUSC algorithm, iCVIs, and other adaptive models on activity data from TigerPlace to predict early signs of illness. Early detection of health changes is critical to promote health while controlling healthcare costs. Identifying and assessing problems early provides a window of opportunity for intervention to solve the problems before they become serious. To make it possible for elders to live independently at home and yet get help from health care providers when small changes in health conditions take place, smart home technologies are developed to enhance safety and monitor health conditions via noninvasive sensors and other devices. TigerPlace is an example of smart home technologies in Columbia, Missouri. This environment is designed to help residents avoid expensive and debilitating hospitalizations and, for most residents, to avoid relocation to a nursing home. An integrated monitoring system has been developed to capture data about a resident using non-wearable, environmentally-mounted sensors. This system is now being tested with TigerPlace residents using very simple algorithms to generate alerts on health changes related to a single alert parameter and to capture the clinical relevance of the alerts through feedback from the clinical staff. Although evidence shows that early illness alerts do improve health outcomes [13], half of the alerts generated are false alarms due to the current one-dimensional strategy. Abnormal sensor patterns produced by certain resident behaviors are linked to early signs of illness. Each resident included in the study (around 100 as of January 2019) has a data logger in his/her apartment that collects data from a wireless sensor network. Each sensor network consists of several types of sensors mounted throughout the resident's apartment, including motion and bed sensors. The health data for each resident is stored in a home-grown nursing EHR (see more details at http://eldertech.missouri.edu/papers).

Our goal is that by combining more than one alert parameter, we can find more of the potential problems that we are missing; and by using more sophisticated temporal analysis method, we can capture more predictive alerts and more customized alerts that can help us detect more meaningful health changes before they become big problems. After looking at years of embedded sensor data, it is observed that feature vectors generated from normal days tend to form clusters in Euclidean feature space while abnormal days appear as outliers or clusters in different locations (may be of smaller size) (**Figure *1-1***) [14]. Yet to determine the real health outliers can be complicated by noise in the sensor data caused by sensor failure, visitor activity or extended absence. Streaming clustering is a candidate for early illness recognition in elderly because of the continuous changing in their life patterns due to the fact normal and abnormal behaviors could form different clusters.

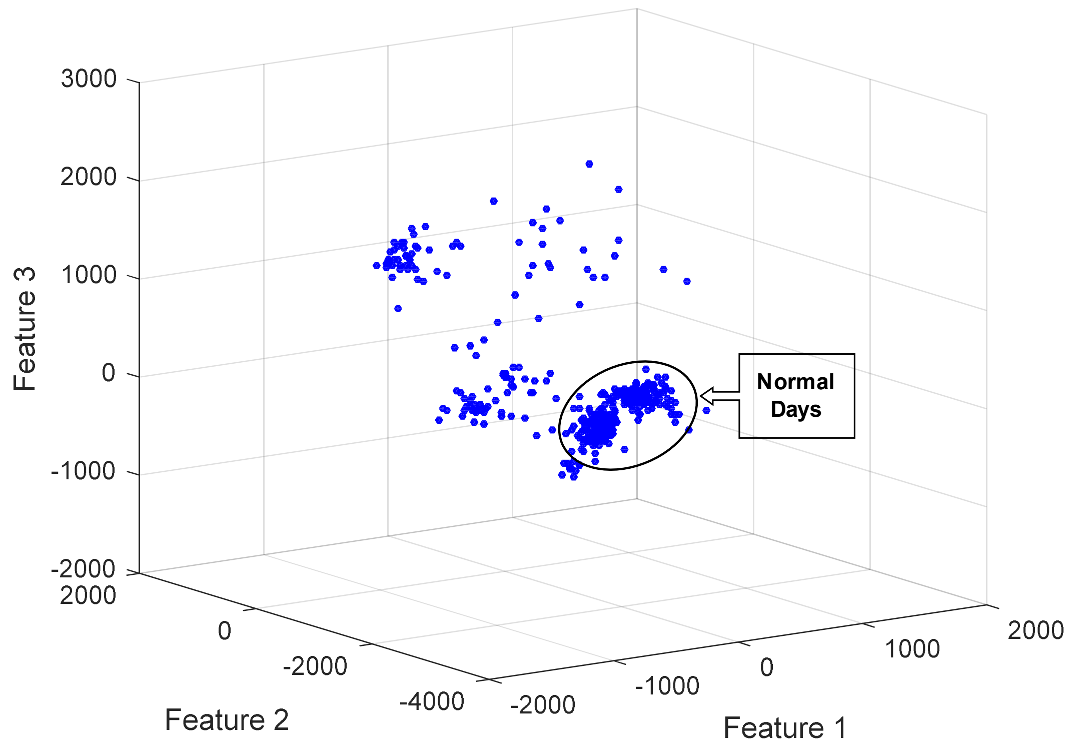


Figure 1-1 Year and a half worth of data for a resident in TigerPlace

1.3 Contributions

This dissertation is comprised of nine research papers. Each of the papers comprises a single chapter; starting with Chapter 2 and finishing with Chapter 10. These papers address the issues previously discussed, and develop and evaluate our streaming clustering algorithm, incremental cluster validity indices, and a novel representation of sensor data from TigerPlace activity dataset.

The first paper (chapter two) [15], “Robust On-Line Streaming Clustering,” focuses on the streaming clustering algorithm, MU Streaming Clustering (MUSC) algorithm. It is an extension of our paper in [16] and it addresses the issue of clustering streaming data and building an adaptive model. MUSC uses Gaussian mixture model (GMM), coupled with possibilistic clustering to build an adaptive system for analyzing streaming multi-dimensional activity feature vector with the goal of identifying signs of early diseases. The system is based on temporal analysis, including outlier detection, customization and adaption to new changes, together with the creation of new components for GMM in the case of emerging new normal patterns. Possibilistic C-Means (PCM) [17] and Automatic Merging Possibilistic Clustering Method (AMPCM) [18] are combined together to cluster the initial data points, detect anomalies and initialize the GMM. Incoming vectors that match an existing mixture component are used to update that component’s parameters. Data points that do not fit in any of the existing clusters are flagged as anomalies. The points in the anomaly history may or may not indicate the emergence of a new cluster. We check the anomaly list in two different ways. First, we compute the Mahalanobis distance between the outliers and cluster centers. Points are assigned to their closest Gaussians if they are within a pre-specified threshold (the cluster has “grown” to encompass what was initially an anomaly). Second, we look for single or multiple emerging structures by clustering the outliers. It worth mentioning that our approach is incremental, wherein new data are used to update the clustering parameters and then removed from memory. We keep cluster representatives such as means, covariance matrices, cluster cardinalities, as well as the outlier data points. We show the superiority of MUSC over traditional streaming clustering algorithms on synthetic and real data sets.

To draw decisions from a streaming clustering output, we need to develop a real-time monitoring of the streaming clustering algorithm performance. Incremental Cluster Validity Indices (iCVIs) are used to assess the clustering results incrementally as data samples evolve. Daves-Buldin, Xie-Beni, and Dunn indices are among of these that were selected for investigation in this proposal. Incremental Validity Indices (iCVIs) employ the same formula of the traditional CVIs except all the calculations have to be computed incrementally and online as streaming data samples arrive over time.

The second paper (chapter three) [19], “Analysis of Streaming Clustering Using an Incremental Validity Index,” studies the internal incremental Davies-Bouldin (iDB) cluster validity index in the context of streaming data analysis. We extend the original index [8] to a more general version parameterized by the exponent of membership weights. Then we illustrate how the iDB can be used to analyze and understand the performance of the MUSC algorithm. We give examples that illustrate the appearance of a new cluster, the effect of different cluster sizes, handling of outlier data samples, and the effect of the input order on the resultant cluster history.

Similar to the traditional CVIs, we need to develop several iCVIs and compare their performance, robustness, and sensitivity in monitoring a streaming clustering algorithm output and their ability to provide useful information about the streaming data. Therefore, the third paper (chapter four) [20], “Evaluating Evolving Structure in Streaming Data with Modified Dunn's Indices,” extends Dunn’s index [21] to be applicable in streaming clustering. Dunn's original internal cluster validity index was designed to assess partition quality and to identify a "best" crisp c-partition of n objects built from static data sets. This index is quite sensitive to inliers and outliers in the input data, so a subsequent study developed a family of 17 generalized Dunn's indices that extended and improved the original measure in various ways [22]. We present online versions of two modified generalized Dunn's indices for use in dynamic evaluation of evolving (cluster) structure in streaming data. We argue that this method is a good way to monitor the ongoing performance of online clustering algorithms, and we illustrate several types of inferences that can be drawn from such indices. We compare the two new indices to the incremental Xie-Beni and Davies-Boudin indices, which to our knowledge offer the only comparable approaches, with numerical examples on a variety of synthetic and real data sets.

The fourth paper (chapter five) [23], "Analysis of incremental cluster validity for big data applications," employs our streaming clustering algorithm and iCVIs in the context of big streaming data analysis. We investigate the effect of large number of samples and high dimensions on the values of the iCVI (iDB). Finally, we propose a way to project streaming data into a lower space for cases where the distance measure does not perform as expected in the high dimensional space.

The fifth paper (chapter six) [24], "a new incremental cluster validity index for streaming clustering analysis," presents an incremental version of the Partition Coefficient and Exponential Separation (PCAES) cluster validity index in the context of streaming data analysis. Incremental PCAES (iPCAES) can be used to monitor evolving structures in streaming data. We investigate the use of the proposed index to understand and analyze the performance of the MU Streaming Clustering (MUSC) algorithm. We compare the performance of iPCAES index with the incremental Davies-Boudin index (iDB) because iDB was found to be the most stable among other incremental indices that offer comparable approaches.

The above approaches are tested with synthetic datasets and real-life datasets. Another real-life streaming dataset comes from activity monitoring in elderly residents' facilities such as TigerPlace in Columbia, Missouri. TigerPlace offers various kinds of services as needed, promoting independence and helping residents remain healthier and active longer by providing ongoing assessments for early illness recognition and health promotion activities. MUSC is used with a dataset from TigerPlace, part of the experiment in chapter 5 [23]. We use multiple (30) features such as time in bed, time in each room, number of bathroom visits, etc. Due to having different types of features (ordinal, numerical, etc.) in this high dimensional space (called the upspace), Euclidean distance measure did not perform as expected, resulting in clustering output that did not match the health record of the selected resident. Therefore, Random Projection [25] is used to map the data from 30-dimensional space to 3-diminsional space. Since the data arrives in a stream (one data point per day), we use a small window (100 data points) to initialize the algorithm and find the projection matrix (R) to map the data. We generate multiple projection matrices and select the one that best preserves the pairwise distance. The best projection matrix (R) is then used to project new arriving data points before passing through MUSC to either be assigned to one of the existing clusters or the anomaly list, or used to create a new cluster.

To avoid losing some information about the resident's behavior by projecting the data to a lower space, our goal is to develop a feature representation that can capture the resident's daily activity because activity recognition is one possible approach to detect early signs of illness. The idea is to find the activities that an elderly would perform in normal days and when he or she deviates from these activities, an alert is generated. The sixth paper (chapter seven) [26], "Unsupervised Analysis of Activity Patterns in Eldercare Monitoring," investigates a new way to convert sensor data of residents in TigerPlace to a set of sub activities. For each day, the sensors firings are converted into sequences of discrete symbols where each symbol represents one sensor type. The idea is to consider the activity (behavior) of a resident represented by a sequence of sensor firings as it would be represented by his/her genome. We split the daily sensor sequence in subsequences using a separation threshold of 30 seconds, which provides enough granularity to capture daily activities. The resultant subsequences are of different lengths. Since it is unknown to us how many activities an elderly resident performs, an unsupervised approach is more suitable to our problem. Clustering is employed to find the number of activities for a specific resident using the sensor data. The distance measure between the subsequences is the key factor for finding and comparing daily activities. The Smith-Waterman algorithm [27] is used initially since we have symbolic sequences that mimic bioinformatics sequences. Since the sequences vary vastly in their length, standard hierarchical clustering does not accommodate sequences of the same activities into the same cluster. Therefore, the symbolic sequences are normalized to have the same length where each feature is one of the sensors reading (pulse, restlessness, bedroom motion etc.). We use a bag-of-words approach to map each sequence into an M-Dimensional (M is number of sensors) Euclidean space representing the percentage of each symbol in a sequence. After converting the symbolic sequences to numeric sequences, Euclidean distance is used, and any clustering algorithm can be applied on the data. To ground truth our approach, we employed a normal/abnormal labeling of each day based on clinically-validated health alerts from our EHR. Days on which a health alert was generated by a fall or other health event were labeled “abnormal”, while days on which no alert was generated were labeled “normal”.

The days preceding and following a health alert were excluded from analysis. To explore behavior patterns captured by our sensors, we clustered both the normal and the abnormal datasets using hierarchical clustering, and used the Calinski-Harabasz index to find the most probable number of clusters.

From the approach in chapter 7, we end up with several hundreds of sequences per day for each resident. To measure the similarity/dissimilarity between different days, we need to combine the sequences of a day into a single feature vector that preserves the context of daily activities. The seventh paper (chapter eight) [28], "Context Preserving Representation of Daily Activities in Elder Care," proposes an unsupervised method for detecting abnormal behavior patterns based on a new context preserving representation of daily activities. The daily numeric sequences are converted into single data point by concatenating a five-bin histogram on each feature together. Using the proposed method, a day with hundreds of sequences is converted into a single data point representing that day and preserving the context of the daily routine at the same time.

The existing early illness alert system in TigerPlace is univariate, meaning it treats each sensor data stream as an independent variable. The alert system triggers an alert if there is an increase or decrease in the sensor data during a day as compared to the average and the standard deviation from W (a given window size) previous days. However, if there is an increase in multiple sensor values for a day but it is not significant to trigger an alert for any of the sensors, some health changes could potentially go undetected. It was demonstrated through a survey from our clinical team that some health issues (such as UTI and dementia) in older adults are captured by a combination of sensors. Therefore, we need an alert system that takes into account multiple sensor data changes at the same time to detect early signs of illness. In chapter 9 (the eighth paper) [29], "an unsupervised framework for detecting early signs of illness in eldercare," we use the representation developed in chapter 8 and we extend the idea in chapter 7 to introduce a novel multi-dimensional alert system to detect (abnormal) days with early signs of illness based on daily activities.

As we mentioned, abnormal sensor patterns produced by certain resident behaviors can be linked to early signs of illness. The ninth paper, (chapter 10) [30], "An Automatic Framework for Semantic Annotation of Eldercare Sensor Data," introduces a framework for detecting health patterns based on non-wearable sensor sequence similarity and natural language processing (NLP). The proposed framework utilizes sensor sequence similarity and medical concepts extracted from the EHR to automatically inform the nursing staff when health problems are detected. The context preserving representation of daily activities is used to measure the similarity between the sensor sequences of different days. The medical concepts are extracted from the nursing notes using MetamapLite, an NLP tool included in the Unified Medical Language System (UMLS) (http://metamap.nlm.nih.gov/). The proposed idea is validated on two datasets from twelve Tiger Place residents, with a total of 5810 sensor days, out of which 1966 had nursing notes.

1.4 List of Publications

1- **O. A. Ibrahim**, J. Shao, J. M. Keller, M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Vancouver, Canada, 2016, pp. 186-193.

2- **O. A. Ibrahim**, Y. Du, J. Keller, “Extended robust on-line streaming clustering (EROLSC),” Proceedings, 17th International Conference on *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Cádiz, Spain, June, 2018, pp. 467-478.

3- **O. A. Ibrahim**, J. M. Keller, and J. C. Bezdek, “Analysis of Streaming Clustering Using an Incremental Validity Index,” in *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, Rio de Janeiro, Brazil, 2018, pp. 212-219.

4- **O. A. Ibrahim**, J. M. Keller, J. Bezdek “Evaluating Evolving Structure in Streaming Data with Modified Dunn's Indices,” submitted to *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2018.

5- **O. A. Ibrahim**, Y. Wang, J. M. Keller, “Analysis of Incremental Cluster Validity for Big Data Applications,” in press, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*.

6- **O. A. Ibrahim**, M. Popescu, and J. M. Keller, “Unsupervised Analysis of Activity Patterns in Eldercare Monitoring,” in *American Medical Informatics Association* Annual Symposium, Washington DC, November 4-8, 2017.

7- **O. A. Ibrahim**, M. Popescu, J. Keller “Context Preserving Representation of Daily Activities in Elder Care,” *Proceedings, IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Kansas City, USA, 2017, pp. 547-55.

8- **O. A. Ibrahim**, M. Popescu, and J. M. Keller, "An Automatic Framework for Semantic Annotation of Sensor Data," under preparation (to be submitted to *IEEE Transactions on Biomedical Engineering)*.

9- **O. A. Ibrahim**, James M. Keller, and M. Popescu. "A New Incremental Cluster Validity Index for Streaming Clustering Analysis." In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (pp. 1-8). 2019.

10- **O. A. Ibrahim**, James M. Keller, and M. Popescu. "An Unsupervised Framework for Detecting Early Signs of Illness in Eldercare." In IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 1043-1050, 2019.

1.5 References

[1] SensorScope. http://lcav.epfl.ch/page-86035-en.html, 2007.

[2] Bezdek, J. C., Rajasegarar, S., Moshtaghi, M., Leckie, C., Palaniswami, M., and Havens, T. C (2011). Anomaly detection in Environmental Monitoring Networks, Computational Intelligence Magazine. 6(2), 52-58.

[3] MJ Rantz, et al., "TigerPlace: A new future for older adults," Journal of Nursing Care Quality, vol 20, no. 1, 2005, pp. 1-4.

[4] Aggarwal, Charu C., ed. Data streams: models and algorithms. Vol. 31. Springer Science & Business Media, 2007.

[5] MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1, no. 14, pp. 281-297. 1967.

[6] P. Angelov and X. Zhou, "Evolving fuzzy-rule-based classifiers from data streams," IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pp. 1462–1475, 2008.

[7] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "On Clustering Massive Data Streams: A Summarization Paradigm," Springer US, vol. 31, pp 9–38, 2007.

[8] N. Ailon, R. Jaiswal, and C. Monteleoni, "Streaming k-means approximation," In Neural Information Processing Systems, vol. 22, pp. 10–18, 2009.

[9] M. Salehi, C. Leckie, M. Moshtaghi, and T. Vaithianathan, "A relevance weighted ensemble model for anomaly detection in switching data streams," In Proceedings of PAKDD, pp. 461–473, 2014.

[10] M. Moshtaghi, J. Bezdek, S. Erfani, C. Leckie, and J. Bailey, "Online Cluster Validity Indices for Streaming Data," arXiv preprint arXiv:1801.02937, 2018.

[11] D. L. Davies and D. W. Bouldin, "A cluster separation measure", IEEE Trans. Patt. Anal. and Mach. Intell., 1(4), 224-227, 1979.

[12] X. L. Xie and G. Beni, “A validity measure for fuzzy clustering,”IEEE Trans. Patt. Anal. and Mach. Intell., 13(8), pp. 841–847, 1991.

[13] Skubic M, Rantz M, Miller S, Guevara RD, Koopman R, Alexander G, Phillips L,” Non-Wearable In-Home Sensing for Early Detection of Health Changes,” Quality of Life Technology for the Disabled and Elderly, R Schultz, ed. Boca Raton, FL: CRC Press, pp 227-244, 2013.

[14] Wang S., “Change Detection for Eldercare Using Passive Sensing,” PhD Dissertation, Electrical & Computer Engineering, University of Missouri, Columbia, MO, Dec, 2011.

[15] O. A. Ibrahim, Y. Du, and J. M. Keller, “Extended robust on-line streaming clustering (EROLSC),” In IPMU, 2018.

[16] O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.

[17] Krishnapuram R., Keller 1.M., "A possibilistic approach to clustering," IEEE Trans. Fuzzy Syst., vol. I, no. 2, pp. 98-1 10, May 1993.

[18] Miin-Shen Yang; Chien-Yo Lai, "A Robust Automatic Merging Possibilistic Clustering Method," Fuzzy Systems, IEEE Transactions on , vol.l9, no.l, pp.26,41 , Feb. 201 1.

[19] O. A. Ibrahim, J. M. Keller, and J. C. Bezdek, “Analysis of Streaming Clustering Using an Incremental Validity Index”, In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1-8, 2018.

[20] O. A. Ibrahim, J. M. Keller, J. C. Bezdek “Evaluating Evolving Structure in Streaming Data with Modified Dunn's Indices,” submitted to IEEE Transaction on Evolving Topics in Computational Intelligence, 2018.

[21] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters", J. Cybernetics, 3(3), 1973, pp. 32-57.

[22] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," IEEE Trans. SMC, 28(3), pp. 301-315, 1998.

[23] O. A. Ibrahim, Y. Wang, J. M. Keller, “Analysis of Incremental Cluster Validity for Big Data Applications,” accepted at the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2018.

[24] O. A. Ibrahim, James M. Keller, and M. Popescu. "A New Incremental Cluster Validity Index for Streaming Clustering Analysis." In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (pp. 1-8). 2019.

[25] W. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," Contemporary Mathematics, vol. 26, 1984, pp. 189-206.

[26] O. A. Ibrahim, M. Popescu, J. C. Keller, " Unsupervised Analysis of Activity Patterns in Eldercare Monitoring," In American Medical Informatics Association (AMIA) Annual Symposium Proceedings, 2017.

[27] Smith TF, “Waterman MS. Identification of common molecular subsequences,” Journal of Molecular Biology. 1981;147:195–197.

[28] O. Ibrahim, J. Keller, J. and M. Popescu, “Context preserving representation of daily activities in elder care,” In Bioinformatics and Biomedicine (BIBM), 2017 IEEE International Conference on (pp. 547-551). IEEE.

[29] O. A. Ibrahim, James M. Keller, and M. Popescu. “An Unsupervised Framework for Detecting Early Signs of Illness in Eldercare.” In IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 1043-1050, 2019.

[30] O. A. Ibrahim, M. Popescu, and J. M. Keller, “An Automatic Framework for Semantic Annotation of Sensor Data,” to be submitted to IEEE transaction in Biomedical Engineering.

Chapter 2: Robust On-Line Streaming Clustering

Omar A Ibrahim[1], Yizhuo Du[1] and James Keller [1]

[1] Electrical Engineering and Computer Science Dept

University of Missouri, Columbia MO 65211, USA

<oai9bc,ydypb>@mail.missouri.edu

kellerj@missouri.edu

Abstract. With the explosion of ubiquitous continuous sensing, on-line streaming clustering continues to attract attention. The requirements are that the streaming clustering algorithm recognize and adapt clusters as the data evolves, that anomalies are detected, and that new clusters are automatically formed as incoming data dictate. In this paper, we extend an earlier approach, called Extended Robust On-Line Streaming Clustering (EROLSC), which utilizes both the Possibilistic C-Means and Gaussian Mixture Decomposition to perform this task. We show the superiority of EROLSC over traditional streaming clustering algorithms on synthetic and real data sets.

Keywords: Streaming Clustering, Outlier Detection, Change Detection

2.1 Introduction

Monitoring systems, the internet of things (IoT), and mining content from social media are new emerging applications that rely on processing large amounts of streaming data. For any data analytics technique to be applied on these applications, it has to be unsupervised, online and temporal, i.e., adaptive over time. Clustering is a data mining technique that searches for specific structures on streaming data and detects abnormal patterns in the data [1]. To find clusters in a dataset, clustering algorithms usually require multiple runs over the data, which necessitates all of the dataset to be available before running the algorithm. In a streaming data problem, there is a desire to learn structure as the data arrives instead of waiting for processing by standard techniques. Therefore, developing efficient streaming (online) clustering algorithms is not an easy task.

Online clustering algorithms can be divided into two groups [2]. The first group is general clustering algorithms which do not assume any ordering on the data stream and work on any streaming data, such as sequential k-means (sk-means) [3]. Algorithms of this group requires the number of clusters to be known in advance. On the other hand, the second group relies on the assumption that close observations in time are highly related. Consequently, online clustering algorithms from this group assume natural ordering on the data stream as in time series. These algorithms have a change detection technique to detect new emerging clusters in the streaming information. This paper is an extension of our previous work [4]. In that paper, we only tested the algorithm with synthetic datasets that mimic the behavior of elder adults. Here, we extend the original algorithm by first investigating the ability of our algorithm to detect multiple emerging structures at the same time, whereas in [4], we looked for one new cluster only. Also, we use only online incremental update for all the parameters on the algorithm. Consequently, new data points are used to update the parameters and then removed from memory, which makes our algorithm applicable for big data applications. This new algorithm is called Extended Robust On-Line Streaming Clustering (EROLSC). Finally, we test EROLSC with synthetic and real-life datasets, and compare the clustering results with sk-means, Basic Sequential clustering algorithm (BSAS) [5], and Modified Basic Sequential clustering algorithm (MBSAS) [6]. The next section describes background information and related work. In section 2.2, we present the datasets used on the evaluation process. In Section 2.3, we describe our online clustering algorithm and the other algorithms used in this paper. Section 2.4 shows numerical evaluation of our method and comparisons to previous approaches. A summary and conclusions are given in Section 2.5.

2.2 Background

There are two main categories to cluster streaming data [7]. The first category uses a window (S) of the data stream and clusters data points in S using batch-clustering techniques. Clustering results of adjacent windows are combined to get the final clustering results [8, 9, 10]. Computational complexity is one drawback of algorithms in this category due to the multiple passes over the data in each window. The second category uses incremental learning techniques and are known as online clustering or streaming clustering algorithms [11, 12]. After initialization, data points are processed one at a time, which makes them good candidates for big data. Online clustering algorithms that assume natural ordering of the data streams have two main parts. The first part is a change detection mechanism that helps the algorithm in identifying new structures in the data. The second part is an adaptive model for the clusters [2].

Change detection is a major part of the online clustering algorithm because it detects new emerging structures and finds outliers on the data streams. In [13], the change detection is based on the violation of the exchangeability condition using a randomized power Martingal that makes it unsuitable for time series.

The detection mechanism used in this paper is based on possibilistic c-means clustering and cluster dispersion. To the best of our knowledge, all existing online clustering algorithm look for one new structure at a time. However, EROLSC can detect multiple emerging structures at the same time because outliers are clustered with more than one cluster and only dense regions in the outlier set are flagged as new clusters.

2.3 Incremental Clustering Algorithms (Sequential Algorithms)

Sequential clustering algorithms form one approach to produce a single clustering by iterating through subsets of the data once or a few times. In our evaluation, we examine sk-means, BSAS, and MBSAS algorithms to compare with our algorithm. In the next section, we provide a brief overview of these incremental clustering algorithms.

2.3.1 Extended Robust On-Line Streaming Clustering Algorithm

In [4], we proposed a streaming clustering algorithm based on Gaussian Mixture Models (GMM) combined with possibilistic fuzzy clustering. The idea is to combine the Possibilistic C-Means (PCM) [14] and the Automatic Merging Possibilistic Clustering Method (AMPCM) [15] to initialize the cluster structure in a window S, initializing the GMM as can be seen in the initialization of figure 2-5. PCM is used to detect anomalies in that first window, S. When a new data point x_{n+1} arrives at time n+1, its Mahalanobis distance is computed to all Gaussians as in equation 1. If the minimum distance falls within pre-specified threshold, x_{n+1} is incorporated into the winning Gaussian. The mean and covariance of the winning Gaussian are incrementally updated using equations 2 and 3. After updating the Gaussian parameters, x_{n+1} is removed from the records.

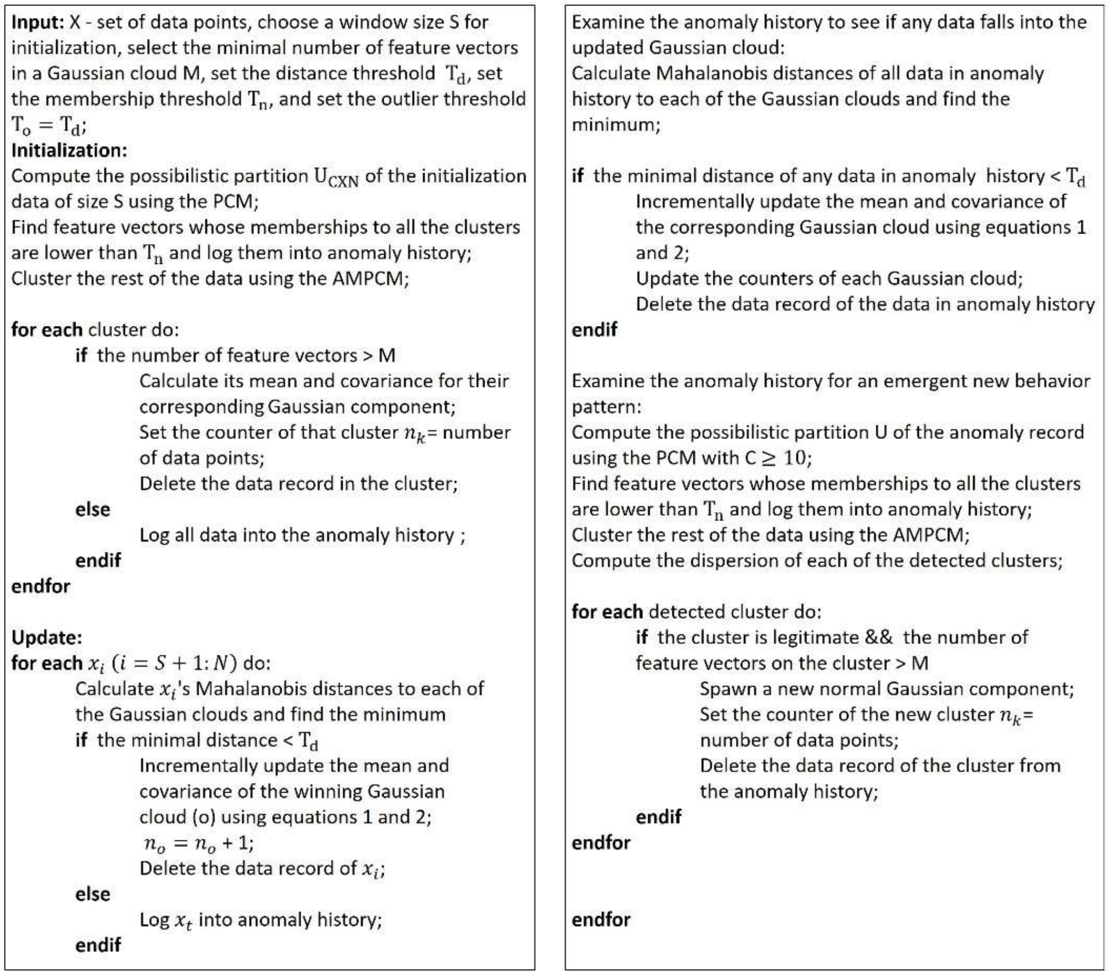
$$\mathrm{d} = \sqrt{(\mathrm{x} - \mu)^{\mathrm{T}}\Sigma^{-1}(\mathrm{x} - \mu)} \tag{1}$$

$$\mu_{new} = \mu_{\mathrm{old}} + \frac{x_{n+1} - \mu_{\mathrm{old}}}{|\mu_{new}|} \tag{2}$$

$$\Sigma_{new} = \frac{(|\mu_{new}|-1) * \Sigma_{\mathrm{old}} + (x_{n+1} - \mu_{\mathrm{old}})^{\mathrm{T}}(x_{n+1} - \mu_{\mathrm{new}})}{|\mu_{new}|} \tag{3}$$

where $|\mu_{new}|$ is the cardinality of the winning cluster, μ and Σ are mean and covariance of the cluster.

On the other hand, the new input vector is flagged as an outlier and saved on the anomaly list if it does not meet the threshold. The points in the anomaly history may or may not indicate the emergence of a new cluster. We check the anomaly list in two different ways. First, we compute the Mahalanobis distance between the outliers and cluster centers. Points are assigned to their closest Gaussians if they are within a pre-specified threshold as can be noticed on **Figure** ***2-5*** (the cluster has "grown" to encompass what was initially an anomaly). Second, we look for single or multiple emerging structures by clustering the outliers as shown in the pseudo code in **Figure** ***2-1***. See [4] for more detailed description of the basic algorithm, along with details on initialization and new cluster formation. One significant feature of EROLSC is that the PCM can be used with C = 1, or with a larger value of C causing co-incident clusters if there are fewer actual groups. It worth mentioning that our approach is incremental, wherein new data are used to update the clustering parameters and then removed from memory. We keep cluster representatives such as means, covariance matrices, cluster cardinalities, as well as the outlier data points.



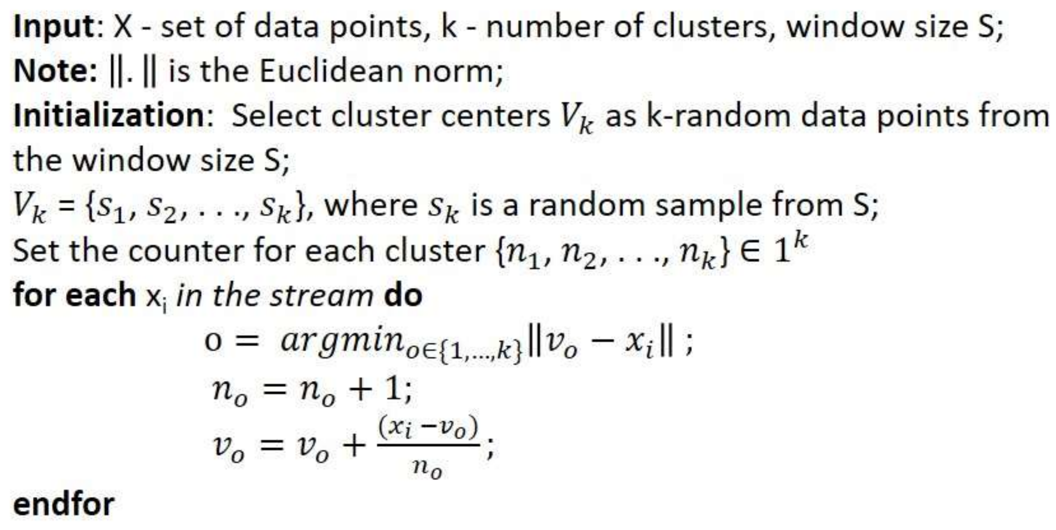
```
Input: X - set of data points, choose a window size S for
initialization, select the minimal number of feature vectors
in a Gaussian cloud M, set the distance threshold $T_d$, set
the membership threshold $T_n$, and set the outlier threshold
$T_o = T_d$;
Initialization:
Compute the possibilistic partition $U_{CXN}$ of the initialization
data of size S using the PCM;
Find feature vectors whose memberships to all the clusters
are lower than $T_n$ and log them into anomaly history;
Cluster the rest of the data using the AMPCM;

for each cluster do:
    if the number of feature vectors > M
        Calculate its mean and covariance for their
        corresponding Gaussian component;
        Set the counter of that cluster $n_k$= number
        of data points;
        Delete the data record in the cluster;
    else
        Log all data into the anomaly history ;
    endif
endfor

Update:
for each $x_i$ $(i = S + 1: N)$ do:
    Calculate $x_i$'s Mahalanobis distances to each of
    the Gaussian clouds and find the minimum
    if the minimal distance < $T_d$
        Incrementally update the mean and
        covariance of the winning Gaussian
        cloud (o) using equations 1 and 2;
        $n_o = n_o + 1$;
        Delete the data record of $x_i$;
    else
        Log $x_t$ into anomaly history;
    endif

Examine the anomaly history to see if any data falls into the
updated Gaussian cloud:
Calculate Mahalanobis distances of all data in anomaly
history to each of the Gaussian clouds and find the
minimum;

if the minimal distance of any data in anomaly history < $T_d$
    Incrementally update the mean and covariance of
    the corresponding Gaussian cloud using equations 1
    and 2;
    Update the counters of each Gaussian cloud;
    Delete the data record of the data in anomaly history
endif

Examine the anomaly history for an emergent new behavior
pattern:
Compute the possibilistic partition U of the anomaly record
using the PCM with $C \geq 10$;
Find feature vectors whose memberships to all the clusters
are lower than $T_n$ and log them into anomaly history;
Cluster the rest of the data using the AMPCM;
Compute the dispersion of each of the detected clusters;

for each detected cluster do:
    if the cluster is legitimate && the number of
    feature vectors on the cluster > M
        Spawn a new normal Gaussian component;
        Set the counter of the new cluster $n_k$=
        number of data points;
        Delete the data record of the cluster from
        the anomaly history;
    endif
endfor


endfor
```

Figure 2-1. Pseudo code of EROLSC

2.3.2 Sequential k-means (sk-means)

Sk-means is a well-known clustering algorithm, which was introduced by Macqueen [3] and the pseudo code can be seen in **Figure 2-2**. Different forms of sk-means have been introduced in the literature [9]. One drawback of sk-means is that number of clusters, k, needs to be stated in advance in Macqueen's algorithm. It can be initialized in different ways: selecting the first k data points, randomly selecting k data points from a window of size S (S<N), or randomly selecting k data points from the whole dataset (size N). We use the second way to initialize the prototypes because we use a window size S to initialize

EROLSC, which leads to a fair comparison. After that, those k data points represent the cluster centers, $V_k = \{v_1, v_2, \ldots, v_k\}$, each with cluster cardinality of 1. When a new data point arrives at time n+1, its distance is computed to all k prototypes, and it is assigned to the closest cluster center. Then the parameters of winning cluster are updated as shown in the pseudo code in **Figure *2-2***.



```
Input: X - set of data points, k - number of clusters, window size S;
Note: ||. || is the Euclidean norm;
Initialization: Select cluster centers $V_k$ as k-random data points from the window size S;
$V_k = \{s_1, s_2, \ldots, s_k\}$, where $s_k$ is a random sample from S;
Set the counter for each cluster $\{n_1, n_2, \ldots, n_k\} \in 1^k$
for each $x_i$ in the stream do
        $o = argmin_{o \in \{1,\ldots,k\}} \|v_o - x_i\|$ ;
        $n_o = n_o + 1$;
        $v_o = v_o + \frac{(x_i - v_o)}{n_o}$;
endfor
```

Figure 2-2. Pseudo code of sequential k-means algorithm ([20])

2.3.3 Basic Sequential Algorithm (BSAS)

BSAS is a basic clustering technique where data points are presented to the algorithm only once and number of clusters is not known a priori [5]. Clustering results rely on the dissimilarity measure d (x, V), dissimilarity threshold Θ, and the number of maximum clusters allowed, q. The first data point is used to initialize the first cluster and it represents its cluster center v_1. When a new data point (X_{n+1}) comes in at time n+1, the algorithm computes the distance between the new data point and existing clusters prototypes. If the distance to the closest cluster is within the Θ and maximum number of clusters (q) is not met, the new data point is assigned to the closest cluster. Then, the parameters of winning cluster are updated as can be seen in the pseudo code, shown in **Figure *2-3*** (a). Otherwise, the new data point spawns a new cluster.

2.3.4 Modified Basic Sequential Algorithm (MBSAS)

MBSAS is a modified version of BSAS where it runs through the data samples twice [6]. It overcomes the drawback of BSAS where a sample is assigned to a cluster before all the clusters have been created. In the first phase, the clusters prototypes are determined by assigning only one data point to each cluster. The second phase of the algorithm assigns the remaining data samples to the nearest cluster center. The pseudo code of this algorithm is shown in **Figure *2-3*** (b).

```
Input: X - set of data points, select Θ and q
thresholds;
Initialization: Select the first data point as
the first cluster center:
$v_1$ = {$x_1$};
Set number of clusters m = 1;
Set the counter for each cluster $n_1$ = 1;

for each $x_i$ in the stream from 2 to N do
        $o = argmin_{o \in \{1,\ldots,m\}} \|v_o - x_i\|$ ;
        if d ($x_i$, $v_o$ ) > Θ and m < q
                Create a new cluster
                m = m + 1;
                $v_m$ = {$x_i$};
                $n_m$ = 1;
        else

                Update   cluster o
                parameters
                $n_o$ = $n_o$ + 1;
                $v_o$ = $v_o$ + ($x_i$ − $v_o$) / $n_o$;

        endif

endfor
```

```
Input: X - set of data points, select Θ and q
thresholds;
Initialization: Select the first data point as
the first cluster center:
$v_1$ = {$x_1$};
Set number of clusters m = 1;
Set the counter for each cluster $n_1$ = 1;
for each $x_i$ in the stream from 2 to N do
        $o = argmin_{o \in \{1,\ldots,m\}} \|v_o - x_i\|$ ;
        if d ($x_i$, $v_o$ ) > Θ and m < q
                Create a new cluster
                m = m + 1;
                $v_m$ = {$x_i$};
                $n_m$ = 1;
        endif
endfor

for each $x_i$ in the stream from 1 to N do
        if $x_i$ has not assigned to a cluster
        $O = argmin_{o \in \{1,\ldots,m\}} \|v_o - x_i\|$ ;
        Update the cluster o parameters
        $n_o$ = $n_o$ + 1;
        $v_o$ = $v_o$ + ($x_i$ − $v_o$) / $n_o$;
        endif
endfor
```

(a) (b)

Figure 2-3. Pseudo codes of: a) BSAS algorithm ([5]), b) MBSAS algorithm ([6])

2.4 Datasets

Synthetic and real-life datasets are used in this paper, all of which are presented in a streaming on-line fashion. The first synthetic dataset, S1, has 490 instances in 2-dimensional space, generated from five Gaussian distributions as in **Figure** *2-4* (a). S1 has five clusters (two with 100 samples each and 3 with 80 instances) and random noise (50 samples). The first real-life dataset used in our evaluation is the LG dataset, which is a collection of weather station nodes in the Le Genepi (LG) region in Switzerland [16]. We use two weeks of data at node 18 starting from October 10, 2007. Average surface temperature (T) and humidity (H) readings over 10-minute intervals were used to create a two-dimensional feature vector $\{x_i = \{(T_i, H_i)\}$. The scatter plot of the data can be seen in **Figure** *2-4* (b). By looking at the scatter plot, it does not provide clear visual evidence about number of clusters in the LG data. Therefore, the imagery information from the site is used to show that there is a snowy day during the two-week period. A windy and cold day precedes the snow. For that reason, we consider the LG data to have three different events: sunny days before and after the snow, cold front moving in, and the snowy day as in the "ground truth" seen in **Figure** *2-4* (c). We use k = 3 as the number of clusters for sk-means whereas EROLSC finds the expected number of clusters. BSAS and MBSAS on the other hand rely on the distance threshold to find clusters. We try multiple distance threshold values and select the best results (number of clusters) for comparative purposes.

The second real-life dataset is the wine dataset from UCI [17]. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. In the Wine data, the number of instances are 59, 71 and 48 in classes 1, 2 and 3, respectively. The third real-life dataset is the Iris dataset [17]. It has three classes with 50 instances each in 4-dimensional space. **Table *2-1*** shows a summary.

Table 2-1. Summary characteristics of the datasets used in the evaluation.

Dataset	# instances	# dimensions	# clusters	Noise	Labeling
S1	490	2	5	50	Exact
Weather	1817	2	3	unknown	Our Estimate
Wine	178	13	3	none	Exact
IRIS	150	4	3	None	Exact

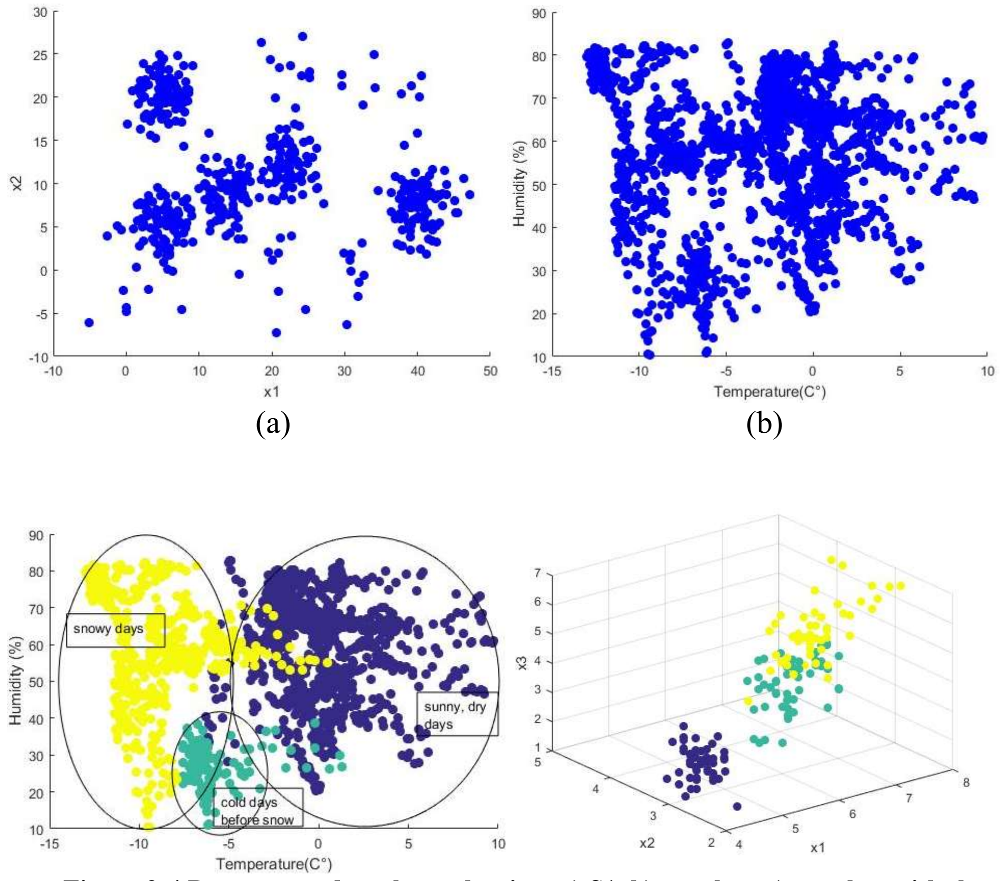


Figure 2-4 Datasets used on the evaluation: a) S1, b) weather, c) weather with the expected number of clusters, d) Iris data

2.4.1 Evaluation

During the evaluation process, we compare EROLSC with sk-means, BSAS, and MBSAS. The setting of each algorithm will be described first. Then, synthetic and real-life datasets are used to discuss the performance of the algorithms. Accuracy is used to compare the performance if class labels are available. Visual inspection in terms of finding the desired number of clusters and/or detecting outliers is used when there are no labels.

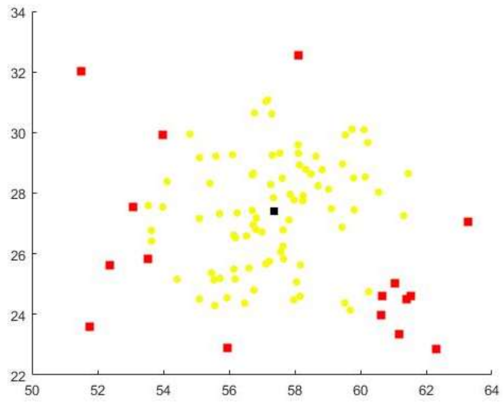
2.4.2 Parameter Settings

Sk-means is fed with number of clusters (K) expected in the dataset because it needs a priori knowledge of the number of clusters. We randomly select K data samples as the initial clusters centers from a window size S. BSAS and MBSAS are fed with the maximum number of desired clusters (K) and distance threshold, which we get by 3experimenting and selecting the best results. For EROLSC, we use a window of size S to initialize the algorithm and we rely on the recommended table in [4] to select the distance threshold. Similar window size is used for sk-means.

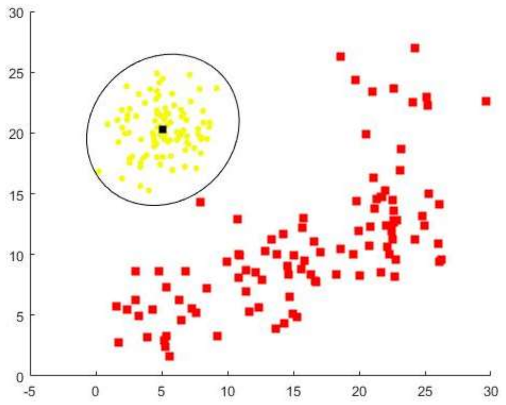
2.4.3 Clustering Results

To demonstrate the ability of EROLSC in detecting multiple clusters at the same time, we use S1 dataset. In S1, data points from cluster 1 come first then data points from cluster 2, 3, and 4 arrive randomly. This enables us to evaluate the robustness of the algorithm to detect multiple structures at the same time. **Figure *2-5*** (a) shows the initialization of the algorithm where PCM is used to detect outliers (red points). When the outliers become dense enough as in **Figure *2-5*** (b), the algorithm in [4] detects only one cluster because it looks for one new emerging structure as in **Figure *2-5*** (c). Clusters 2, 3, and 4 are merged into one cluster which could limit the ability of using the algorithm in real applications where multiple structures could emerge at the same time. EROLSC on the other hand detects the three structures as can be seen in **Figure *2-5*** (d) because it takes into account that multiple clusters could happen a similar time.

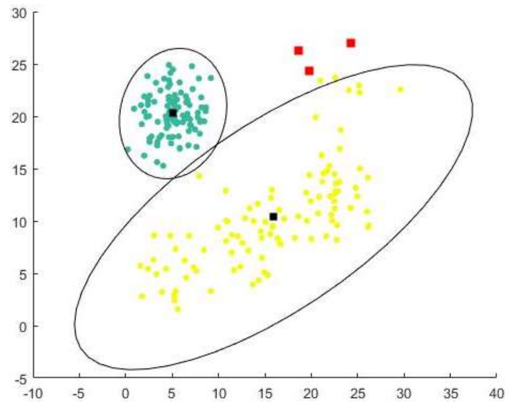
The clustering results on the first dataset S1 are shown in **Figure *2-6***. This dataset has five different clusters where it starts: with one cluster, three clusters form at the same time, the fifth cluster forms after that and outlier's data points arrive in between. Result of our algorithm is shown in **Figure *2-6*** (a) where it detects all the clusters and finds 45 out of 50 outliers. Sk-means result can be seen in **Figure 2-6** (b) where 3 centers get trapped in the first cluster due to the initialization. BSAS and MBSAS have better clustering results compared to sk-means as can be noticed in **Figure *2-6*** (c) and **Figure *2-6*** (d). MBSAS detected all the 5 clusters correctly. However, it fails to detect outliers because each sample must be assigned to the nearest prototype.



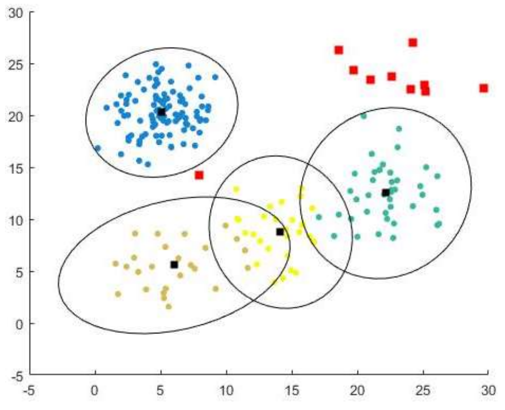
(a)



(b)

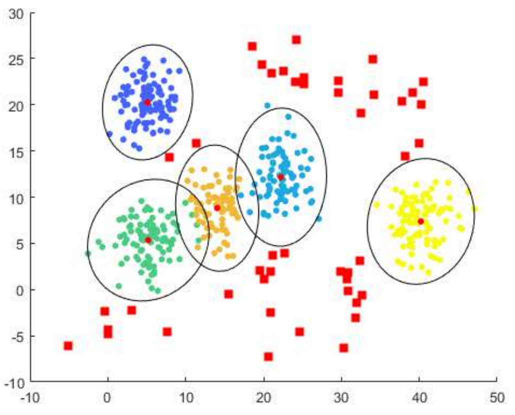


(c)

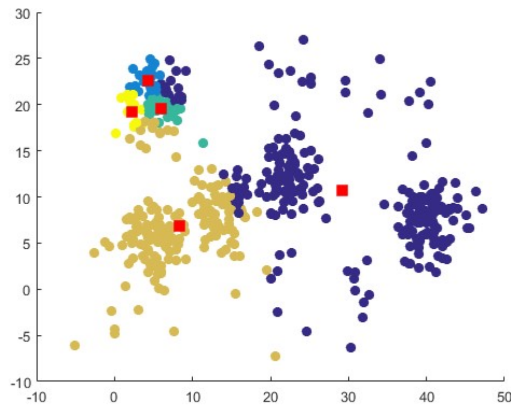


(d)

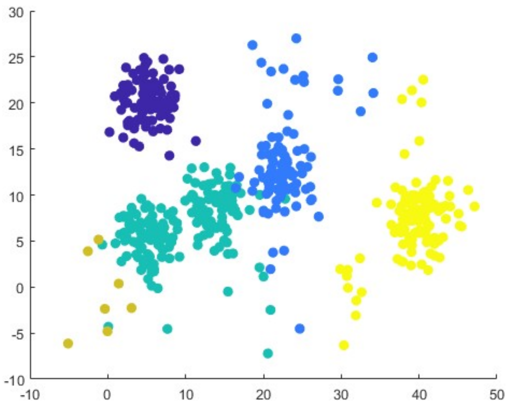
Figure 2-5 Effect of detecting multiple clusters: a) EROLSC initialization, b)Outliers become dense enough to detect new cluster or clusters, c) Using our algorithm in [4] to look for new structures in the outliers, d) Using EROLSC to detect multiple structures in the outliers.

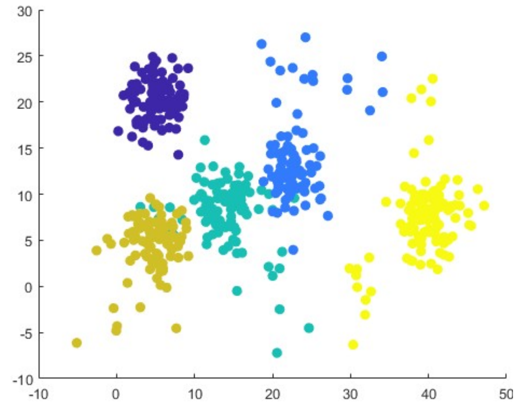


(a)



(b)

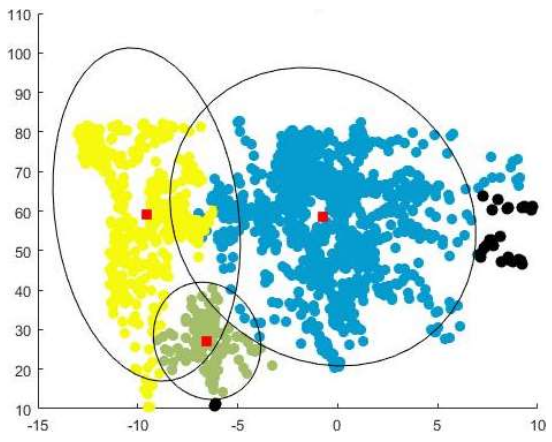


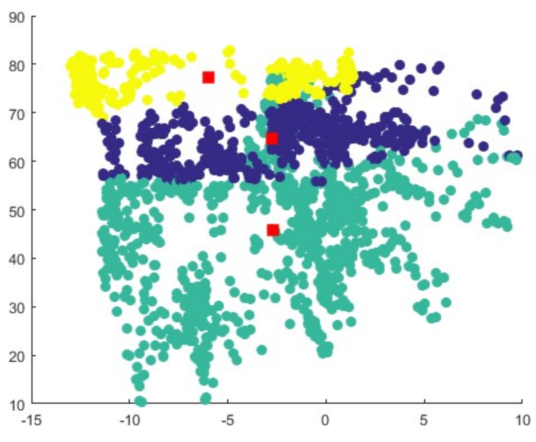


(c) (d)

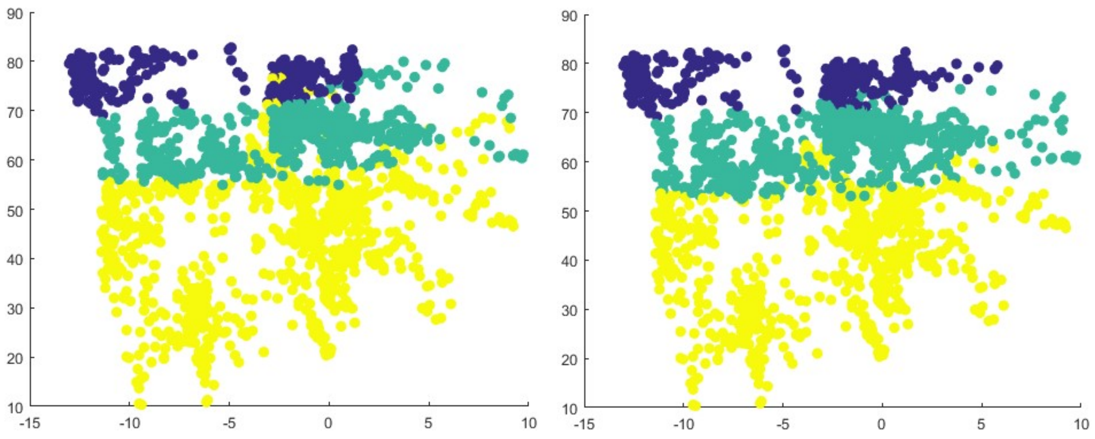
Figure 2-6. Final hardened clustering results on S1 dataset for: a) EROLSC, b) sk-means algorithm, c) BSAS algorithm, d) MBSAS algorithm.

The weather dataset is more complicated and less obvious to cluster. EROLSC finds three clusters and matches the ground truth based on the explanation earlier. **Figure *2-7*** (a) shows the cluster structures of our algorithm, which follows the data evolution as time progresses. We notice that EROLSC flagged some of the sunny cluster data points as outliers due to the way the data is presented to the algorithm. The algorithm initialized itself at the "sunny days" cluster and new samples dragged the cluster center to the left. After that, these data points arrived and are flagged as outliers because they are far from the cluster center. Sk-means detects three clusters as well, but it tries to achieve good separation between clusters as can be seen in **Figure *2-7*** (b). The results of BSAS and MBSAS are not much different from sk-means, where they also look for separated clusters as in **Figure *2-7*** (c) and (d). We conclude that only EROLSC finds the three expected clusters (sunny, windy and cold before snow, and snowy days).





(a) (b)



(c) (d)

Figure 2-7. Clustering results on weather dataset for: a) EROLSC, b) sk-means algorithm, c) BSAS algorithm, d) MBSAS algorithm.

Both the wine and Iris datasets are labeled, so accuracy can be used to evaluate the performance. EROLSC achieves the highest performance on the Iris dataset with 94% accuracy as in **Table *2-2***. The miss-assigned samples (around 9) are those which belong to one class, but they are closer to another, as depicted in **Figure *2-4*** (d). BSAS and MBSAS achieved better accuracy compared to sk-means due to the initialization, we think. Similarly, EROLSC has the highest accuracy in the Wine dataset where it finds the exact number of classes and misclassifies a few data points. This is normal for a clustering algorithm because our main goal is to detect the desired number of structures in the data. Sk-means, BSAS, and MBSAS do poorly on the wine dataset as expected, based on their performance on other datasets.

Table 2-2 Accuracy of the final clusters to match classification labels.

Dataset	**EROLSC**	**Sk-means**	**BSAS**	**MBSAS**
Iris	94%	49%	67%	67%
Wine	92%	51%	33%	33%

2.5 Conclusion

Streaming clustering, directed by change detection, can identify structures in online data whereas traditional batch approaches fail. General online clustering algorithms such as sk-means, BSAS and MBSAS suffer from the same drawback of batch clustering due to their lack of having an effective change detection mechanism. Our approach in this paper, the Extended Robust On-Line Streaming Clustering Algorithm has a change detection mechanism by flagging data points that are far from the existing state or states, and monitors them over time. If the outliers become dense enough, they are added to the current states as new clusters. Otherwise, they are treated as anomalies unless they fall into one of the data structures over time. Having this property of dealing with outliers enables EROLSC to identify anomalies during the process of clustering. For instance, the S1 dataset has multiple outliers, which was only detected by our algorithm. EROLSC detects the expected structures in the real-life datasets. In addition, it outperforms sk-means, BSAS and MBSAS in all datasets tested. More research needs to be done to automatically determine correct parameter settings and to adapt EROLSC to Big Data applications.

2.6 References

1. J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. ao Gama, “Data stream clustering: A survey,” in ACM Computing Surveys, vol. 46, no. 1, pp.13–31, July 2013.
2. M. Moshtaghi, C. Leckie, and J. C. Bezdek, "Online Clustering of Multivariate Time-series," In Proceedings, 2016 SIAM International Conference on Data Mining, pp. 360-368, 2016.
3. J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1, no. 14, pp. 281-297. 1967.
4. O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.
5. S. Theodoridis and K. Koutroumbas, “Basic Sequential Algorithmic Scheme (BSAS),” Academic Press, London England, 1999.
6. S. Theodoridis and K. Koutroumbas, “Pattern Recognition,” Academic Press, London England, 2006.
7. S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O’Callaghan, “Clustering data streams: Theory and Practice,” IEEE Transaction on Knowledge and Data Engineering, vol. 15, no. 3, pp. 515–528, 2003.
8. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, “On Clustering Massive Data Streams: A Summarization Paradigm,” Springer US, vol. 31, pp 9–38, 2007.
9. N. Ailon, R. Jaiswal, and C. Monteleoni, “Streaming k-means approximation,” In Neural Information Processing Systems, vol. 22, pp. 10–18, 2009.

10. M. Salehi, C. Leckie, M. Moshtaghi, and T. Vaithianathan, “A relevance weighted ensemble model for anomaly detection in switching data streams,” In Proceedings of PAKDD, pp. 461–473, 2014.

11. M. Ackerman and S. Dasgupta, “Incremental clustering: The case for extra clusters,” In Advances in Neural Information Processing Systems, pp. 307–315, 2014.

12. P. Angelov and X. Zhou, “Evolving fuzzy-rule-based classifiers from data streams,” IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pp. 1462–1475, 2008.

13. S.-S. Ho, “A martingale framework for concept change detection in time-varying data streams,” In Proceedings of the Int. Conf. on Machine Learning, pages 321–327, 2005.

14. Krishnapuram R., Keller 1.M., "A possibilistic approach to clustering," IEEE Trans. Fuzzy Syst., vol. I, no. 2, pp. 98-1 10, May 1993.

15. Miin-Shen Yang; Chien-Yo Lai, "A Robust Automatic Merging Possibilistic Clustering Method," Fuzzy Systems, IEEE Transactions on , vol.l9, no.l, pp.26,41 , Feb. 201 1.

16. SensorScope. http://lcav.epfl.ch/page-86035-en.html, 2007.

17. Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Chapter 3: Analysis of Streaming Clustering Using an Incremental Validity Index

Omar A. Ibrahim

Dept of Electrical Engineering and Computer Science

Columbia, Mo, USA

oai9bc@mail.missouri.edu

James M. Keller

Dept of Electrical Engineering and Computer Science

Columbia, Mo, USA

kellerj@missouri.edu

James C. Bezdek

Dept of Electrical Engineering and Computer Science

Columbia, Mo, USA

jcbezdek@gmail.com

Abstract — In this paper, we study the internal incremental Davies-Bouldin (iiDB) cluster validity index in the context of streaming data analysis. We extend the original index to a more general version parameterized by the exponent of membership weights. Then we illustrate how the iiDB can be used to analyze and understand the performance of the Extended Robust Online Streaming Clustering (EROLSC) algorithm. We give examples that illustrate the appearance of a new cluster, the effect

of different cluster sizes, handling of outlier data samples, and the effect of the input order on the resultant cluster history.

Keywords—Online clustering, incremental internal cluster validity, change detection, incremental Davies-Bouldin index

3.1 Introduction

New emerging applications such as monitoring systems, the internet of things (IoT), and mining content from social media rely on processing large amounts of streaming data. For any data analytics method to be suitable for these applications, it has to be online, temporal and unsupervised. Clustering is an unsupervised method that looks for structure in the data and attempts to detect abnormal patterns [1]. Clustering algorithms usually involve multiple passes over the data to find cohesive groups in the data, which requires the existence of all data points before running the algorithm. Developing efficient online clustering algorithms is not an easy task due to the desire of learning the structure as data vectors arrive, instead of processing by traditional techniques.

Online clustering algorithms provide a way to analyze and extract knowledge from streaming data, which makes them desirable in applications such as environmental sensing and packet analysis [1, 2]. These applications involve massive data streams with high velocity, which makes processing a data point more than once infeasible. Sequential clustering is one category of online clustering where data points arrive one at a time, and the clustering parameters (such as cluster centers and covariance matrices) are updated with each sample [3, 4, 5].

Once the data are clustered and the partitions found, important questions arise: how good are the partitions? Are there better results? Therefore, an important aspect of clustering is cluster validity, which determines the quality of the clustering results. *Cluster validity indices* (CVIs) are algorithms and computational models that search for the best clustering results for a dataset. Usually, CVIs are either max-optimal or min-optimal, which means the best partitions are those with smallest (largest) CVI value. In the literature, CVIs have been applied to batch clustering algorithms as well as to clustering results generated by different parameter settings of a particular clustering algorithm.

In online clustering, each data point is processed only once and then removed, so that it is not available for later analysis. Therefore, it is more important to monitor the performance of the clustering algorithm at any point in time, and the response of the algorithm to emerging structure in the data. A wide range of cluster validity methods are available for analyzing clusters found by processing static data [7]. By contrast, only two cluster validity indices have been extended to online clustering [6]. Compactness and separation of the clusters are used by most of the batch indices to assess clustering results [7, 6]. Separation is usually measured using the distance between cluster centers, while compactness is usually some measure of the density of the data points in each cluster. Therefore, in an online scheme, an incremental approach for updating the compactness is required because each data point is processed only once.

In this paper, we use one of the *incremental internal cluster validity indices* (iiCVIs) that were developed in [6], and investigate the relationship between the iiCVI output and the partitions produced by our online clustering algorithm. We perform this analysis using the *Extended Robust On-Line Streaming Clustering* (EROLSC) [8]. The derivations of incremental compactness given in [6] fix the membership exponent (m) that weights $\|x_k - v_i\|^2$ to m=2. We show here that any value of m>1 can be used to good effect. We will investigate the effective range of m with a numerical example. This paper has the following contributions: (1) we generalize the membership exponent value in the derivation of incremental compactness; (2) we show the relationship between the iiCVI and EROLSC clusters in the streaming data. Specifically, we show how to interpret the iiCVI results in terms of: when a new cluster is formed (sudden drop in the iiCVI value), the effect of different cluster sizes on the results, and the streaming process of data samples in each cluster; (3) we show the effect of different parameter settings for (m) on the iiCVI results (how to understand the knee points in the graph of the iiCVI). This is important for understanding large volume datasets in high dimensional spaces.

In section 3.2, a summarization of related work is given. Section 3.3 presents the EROLSC online clustering algorithm used in this paper. In section 3.4, background information and the definition of the *Davies-Bouldin* (DB) [9] index is provided. The problem statement is introduced in section 3.5. Section 3.6 shows the datasets used in the evaluation. In Section 3.7, initialization of the parameters is discussed. Experimental results and discussion are shown in section 3.8. Finally, section 3.9 contains our conclusions.

3.2 Background

In this section, we briefly describe related work in online clustering algorithms and cluster validation. It is important to define an online clustering algorithm as our goal is to validate the results of these algorithms using incremental cluster validity indices.

Clustering in data streams can be divided into two main strategies [1]. The first category buffers a window of streamed inputs and clusters the data samples in the buffer by applying traditional clustering algorithms such as the c-means algorithms. Then, clustering results of adjacent windows are merged to obtain a final clustering of the whole data stream [10, 11, 12]. Due to multiple runs over the dataset in each window, computational complexity is one drawback of algorithms in this category. The second group uses incremental learning methods to find clusters in the growing data stream and are known as online or incremental clustering algorithms [8], [13], [14], [15], [16], [17]. Algorithms in the second category reduce the time complexity because after initializing the algorithm, data points are processed one at a time and then discarded, which makes them good candidates for big data streams.

Online clustering algorithms can be subdivided into two groups. General clustering algorithms can be applied to any sequence of data. Examples of algorithms in this group are sequential k-means (sk-means) and sequential agglomerative clustering [18]. These algorithms require the number of clusters to be specified a priori and do not assume any ordering in the data stream. The second group of online clustering algorithms rely on the assumption of some natural ordering in the data such as time-series, and depend on the assumption that close observations in time will be closely related. This assumption is used to dynamically create clusters in evolving data streams. Online clustering algorithms that assume a natural ordering of the data streams have two main parts: (1) a change detection mechanism that helps the algorithm identify new emerging structures in the data, (2) an adaptive model for the clusters [13]. The cluster validation method in this paper applies to the second category of online clustering algorithms.

Cluster validity is a way to evaluate clustering results. For example, after getting a c-partition from a clustering algorithm, is there a better partition that we didn't find? Cluster validity indices (CVIs) can be divided into two main categories. The first category is internal indices, which use only the unlabeled data and information from the algorithm output. The second group is external CVIs, which require external information about the structure in the data. External CVIs are used to compare ground truth labels of the data to the partitions obtained by a clustering algorithm. Therefore, external CVIs can be used to correlate internal and external evaluations of labeled data, and from this comparison, a “good” internal CVI can be chosen [7].

Comparison of internal CVIs (iCVIs) are well discussed in [7, 19]. Internal CVIs provide a non-parametric technique to assess the clustering result while most of the goodness-of-fit measures presented in [20], use parametric methods. In other words, the majority of iCVI models attempt to capture cohesion and separation of the clusters whereas goodness of fit indices typically evaluate the fit of a model to the data that generates it. Internal CVIs are grouped into two categories based on the way that they measure cohesion and separation. The first group determines the quality of the clustering using only the partitions generated by the algorithm. Indices in this category are often simply measures of fuzziness, such as the partition coefficient and partition entropy [21]. More generally, most iCVIs (such as the DB index) use the unlabeled data, the partition, and any auxiliary parameters produced by the clustering such as cluster centers to assess the quality of each partition.

In [6], two incremental internal CVIs, iiCVIs, are developed by deriving an incremental formula for the cohesion term of two well-known iCVIs viz. the *Xie-Beni* (XB, [22]) and Davies-Bouldin (DB, [9]). In their derivations, the exponent of the memberships in all equations is set to m=2, as shown in equation (6). A careful look at the derivations in [6], however, reveals that the term $(u_{ik})^2$ can be replaced by $(u_{ik})^m$ for any m>1. This adds a lot of flexibility to the iiCVIs presented in [6]. Here we investigate the relationship between different values of m and the performance of the online DB index. We will investigate the relationship between the DB iiCVI and the *Extended Robust On-Line Streaming Clustering* (EROLSC) algorithm. More specifically, what can we learn about data streams in term of cluster sizes, how the input data are streamed, and the effect of the mechanism used by the EROLSC algorithm to create a new cluster.

3.3 Extended Robust On-Line Streaming Clustering Algorithm

This section briefly describes the EROLSC online clustering algorithm developed in [23] and extended in [8]. EROLSC is based on combining *Gaussian Mixture Models* (GMM) with possibilistic fuzzy clustering. GMMs are initialized by combining the *Possibilistic C-Means* (PCM) [24] and the *Automatic Merging Possibilistic Clustering Method* (AMPCM) [25] in a window S of the streaming data. PCM is mainly used to detect outliers in the initialization window, S, and for creating a new cluster because of its tendency to look for dense regions in the dataset by generating coincidence clusters. We set the number of clusters that PCM looks for to a large number (for example if 100 data samples are in S, we look for 10 clusters). After removing data points with low typicalities to all existing prototypes, AMPCM is applied to the noise-free data in S and the final clustering result is used to initialize a GMM. When a new data point x_{n+1} arrives at time n+1, its Mahalanobis distance to the means in the GMM is computed as in (1). If the minimum distance falls within pre-specified threshold (T), x_{n+1} is incorporated into the winning Gaussian cluster. The mean and covariance of the winning Gaussian are incrementally updated using (2) and (3). After updating the Gaussian parameters, x_{n+1} is discarded.

$$d = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \qquad (1)$$

$$\mu_{new} = \mu_{old} + \frac{x_{n+1} - \mu_{old}}{|\mu_{new}|} \tag{2}$$

$$\Sigma_{new} = \frac{(|\mu_{new}| - 1) * \Sigma_{old} + (x_{n+1} - \mu_{old})^T (x_{n+1} - \mu_{old})}{|\mu_{new}|} \tag{3}$$

where $|\mu_{new}|$ is the new cardinality of the winning cluster, μ and Σ are mean and covariance matrix of the cluster.

If the minimum distance exceeds the threshold, the new data point x_{n+1} is flagged as an outlier and saved in the anomaly list. The points in the anomaly history may or may not indicate the emergence of a new cluster. We track changes in the anomaly list in two different ways. First, we check if any point in the list could fit in one of the existing clusters by computing the Mahalanobis distance between the outliers and updated cluster centers. Points are assigned to their closest Gaussians if they are within a pre-specified threshold (T) (the cluster has "grown" to incorporate what was an outlier before). Second, we cluster the outliers following the same approach to initialize the GMM at the beginning, where we look for multiple emerging structures. See [23] for more detailed description of the basic algorithm, along with details on initialization, selecting the threshold values, and new cluster formation. The historical footprint of the evolving structure is represented by the means, covariance matrices, and cluster cardinalities, as well as outlier data points.

3.4 Davies-Bouldin Index

In the case of streaming data, there is no ground truth, so external CVIs are not applicable. Therefore, internal CVIs are the only choice to evaluate the evolving performance of online clustering algorithms. A well-known index is used in the evaluation, which is suitable for both hard and soft partitions. The index is a relative of the Davies-Bouldin (DB) index [9] introduced by Araki et al. [26] as in (4) below.

$$DB(U,V;X) = \frac{1}{k}\sum_{i=1}^{k} \max_{j,j\neq i} \frac{L_i + L_j}{\left\|v_i - v_j\right\|^2} \tag{4}$$

$$L_i = \frac{\sum_{j=1}^{n} u_{i,j}^2 \left\|x_j - v_i\right\|^2}{\sum_{j=1}^{n} u_{i,j}^2};1 \le i \le k \tag{5}$$

where k is number of clusters, v_i is the cluster center of cluster i, and $u_{i,j}$ is the membership of data sample j to cluster i.

The DB index measures the overall average similarity between each cluster and its most similar neighbor, so smaller values indicate better partitions. A common factor in many validity indices is the way that the within-cluster dispersion is computed. Moshtaghi et al. [6] defined the fuzzy within cluster dispersion of cluster i at step n as

$$C_{i,n} = \sum_{j=1}^{n} u_{i,j}^2 \left\|x_j - v_{i,n}\right\|^2 \tag{6}$$

Equation (6) depends directly on the input data. In [6], an incremental calculation of the within-cluster dispersion was developed for the DB index (iiDB). **Figure 3-1** shows the procedure for updating the parameters used to compute the iiDB index when data point x_{n+1} arrives [6]. Let DB(n) represent the value of the Davies-Bouldin index after the n-th data sample. We want to compute incrementally updated values DB(n+1) when data sample x_{n+1} arrives. The index DB(n + 1) at time step n+1 can be computed as

$$DB(n+1) = \frac{1}{k}\sum_{i=1}^{k}\max_{j,j\neq i}\frac{L_{i,n+1}+L_{j,n+1}}{\left\|v_{i,n+1}-v_{j,n+1}\right\|^2} \tag{7}$$

where

$$L_{i,n+1} = \frac{C_{i,n+1}}{M_{i,n+1}} \tag{8}$$

$$M_{i,n+1} = M_{i,n} + u_{i,n}^2 \tag{9}$$

In ERLOSC [8], when a new data point arrives, its distance to all (K) prototypes is computed using (1), and its membership in the i-th cluster is computed using (10).

$$u_{i,n+1} = \frac{1}{\sum_{l=1}^{k}\left(\frac{d(x_{n+1},c_i)}{d(x_{n+1},c_l)}\right)^{\frac{2}{m-1}}};1\le i\le k;m>1\ . \tag{10}$$

```
Data: $\boldsymbol{v}_{i,n}, \boldsymbol{v}_{i,n+1}, u_{i,n+1}, \boldsymbol{x}_{n+1}$
Input  : $G_{i,n}, M_{i,n}, C_{i,n}$
Output: $G_{i,n+1}, M_{i,n+1}, C_{i,n+1}$
/* note $i = 1, \ldots, k$
foreach $i \in \{1, \ldots, k\}$ do
    $Q_{i,n+1} = [\boldsymbol{v}_{i,n} - \boldsymbol{v}_{i,n+1}]^T G_{i,n}$;
    $B_{i,n+1} = \|\boldsymbol{v}_{i,n} - \boldsymbol{v}_{i,n+1}\|^2$;
    $A_{i,n+1} = (u_{i,n+1})^2 \|\boldsymbol{x}_{n+1} - \boldsymbol{v}_{i,n+1}\|^2$;
    $C_{i,n+1} = C_{i,n} + A_{i,n+1} + M_{i,n}B_{i,n+1} + 2Q_{i,n+1}$;
    $G_{i,n+1} = G_{i,n} + M_{i,n}(\boldsymbol{v}_{i,n} - \boldsymbol{v}_{i,n+1}) +$
                $(u_{i,n+1})^2 (\boldsymbol{x}_{n+1} - \boldsymbol{v}_{i,n+1})$;
    $M_{i,n+1} = M_{i,n} + u_{i,n+1}^2$;
end
```

Figure 3-1 Pseudo code for updating the compactness parameters [6].

3.5 Problem Statement

In [6], an incremental DB (iiDB) index was proposed and the procedure to compute its parameters is shown in **Figure 3-1**. The exponent of the membership $u_{i,n+1}$ was set to 2 as shown in **Figure 3-1** and in equations (6), (11), (12) and (13). It is easy to check that the exponent can actually take any value m≥1 without violating the constraints of the incremental derivation One question addressed here is what can be learned about the behavior of EROLSC by varying the membership exponent?

$$A_{i,n+1} = (u_{i,n+1})^2 \left\| x_{n+1} - v_{i,n+1} \right\|^2 \tag{11}$$

$$G_{i,n+1} = G_{i,n} + M_{i,n}(v_{i,n} - v_{i,n+1}) + (u_{i,n+1})^2 (x_{n+1} - v_{i,n+1}) \tag{12}$$

$$M_{i,n+1} = M_{i,n} + (u_{i,n+1})^2 \tag{13}$$

where $u_{i,n+1}$ is the membership of a new arriving data point (x_{n+1}) to cluster i, which is computed using (10), $v_{i,n}$ is the cluster center of cluster i after n inputs and $v_{i,n+1}$ is the new cluster center of cluster i after data point x_{n+1} arrives, which is updated using (2) with v instead of μ.

The relationship between the iiDB and dataset variation is not discussed in [6]. More specifically, we address the questions. After spawning a new cluster (seen by an abrupt drop in the iiDB value), what is the relationship between the cluster size and the time it takes before it reaches the next knee point in the iiDB index graph. The knee point is the point in the iiDB graph where there is a sudden increase in the slope, as in **Figure 3-4** (b). Furthermore, the relationship between the iiDB and the procedure followed by the online clustering algorithm to deal with new data samples in term of forming a new prototype can reveal interesting properties of the clustering algorithm.

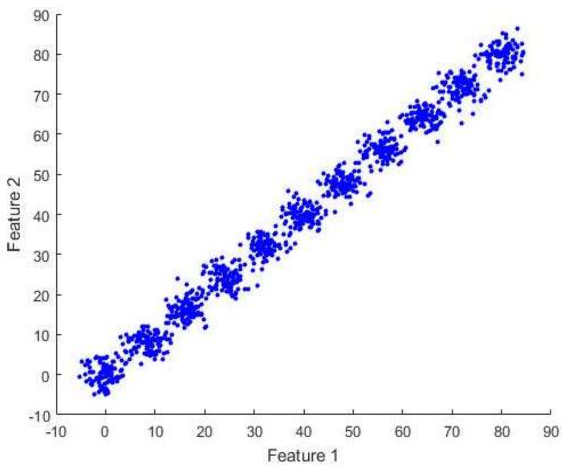
3.6 Datasets

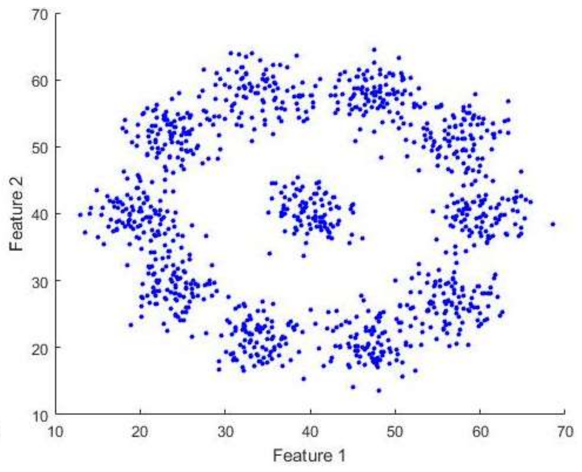
In this section, we describe the datasets used in this paper. The first synthetic dataset is S1, which has 1100 instances in 2- dimensional space generated from 11 Gaussian distributions distributed along a line as in **Figure 3-2** (a). S1 has 11 clusters (100 samples each) as can be seen in Table 3-1. The dataset is ordered such that data vectors from cluster 1 arrive before data points from cluster 2 and so on.

The second synthetic dataset is S2, which has 1100 data points with 11 clusters. Each cluster has 100 samples and the clusters are arranged in a circular shape with one cluster in the middle, as seen in **Figure 3-2** (b). The data samples are ordered so that data points from cluster 1 arrive first. After that, 10 data points from the center cluster arrive, followed by vectors from cluster 2. A similar pattern is repeated for the remaining clusters. Data sets S1 and S2 are similar to two data sets used by the authors of [6].

Table 3-1. Summary characteristics of the datasets used in the evaluation.

Dataset	**# instances**	**# dimensions**	**# clusters**	**Labeling**
S1	1100	2	11	Exact
S2	1100	2	11	Exact





(a) (b)

Figure 3-2 Synthetic datasets: a) S1 dataset, b) S2 dataset.

3.7 Initialization

DB index computation requires the existence of at least two clusters. Therefore, in the ERLOSC cluster algorithm, computation of the iiDB begins after two clusters exist. After using data points in window size S to initialize the algorithm, the parameters in **Figure 3-1** are computed for each cluster after each data points arrives. However, iiDB is calculated once the second cluster is formed (after, say n data points have been processed). Each time a cluster is formed, its parameters are initialized with $C_{i,1} = 0$, $M_{i,1} = w$ (number of elements in cluster i) and $G_{i,1} = \vec{0}$ (zero vector in $\Re^p$). This initialization is repeated each time a new cluster is created. After initialization, the clustering algorithms and iiDB process the data one sample at a time.

3.8 Experimental results

3.8.1 Effect of Membership Exponent on iiDB

The first experiment studies the effect of the membership exponent (m) on the DB index. Different values of m ∈ {1, 1.01, 1.1, 2, 2.5, 3} are used. Hardened clustering results of ERLOSC are shown in **Figure 3-3** where we can see the algorithm detected all the clusters on the two datasets. **Figure 3-4** (a) and **Figure 3-5** (a) graph the iiDB values for different values of (m) on both datasets.

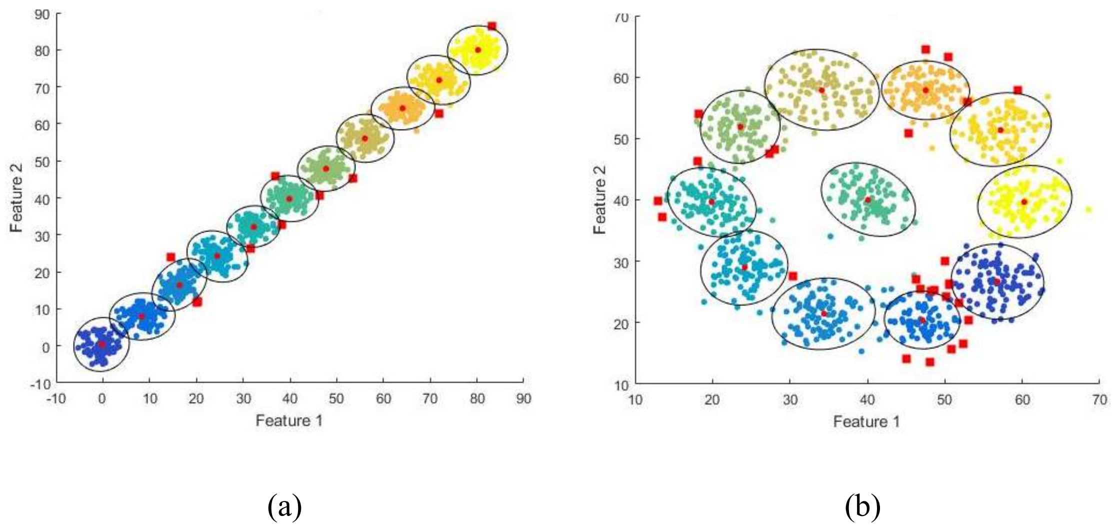
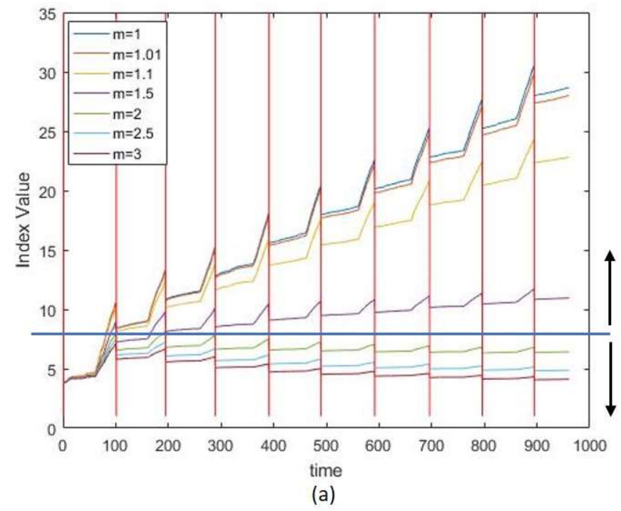


Figure 3-3 Clustering results of a) S1 dataset, b) S2 dataset where the red dots are the cluster centers and the red squares are outliers flagged by the algorithm.

As the exponent (m) of the membership increases, the overall (trend) value of the DB index decreases (this index is min-optimal). As shown in **Figure 3-4** (a), selecting an exponent value greater than or equal to 2 is sufficient to meet the min-optimal DB requirement for these two data sets. Whether this is always true is an open question.

The ERLOSC algorithm performs very well at detecting those times when new distributions are created in both datasets. Sudden changes in in iiDB index indicate changes in the cohesion and separation of the clusters being produced by the clustering algorithm. More generally, a sudden change in an online validity index usually can be associated with the appearance of a new cluster in the streaming data. The red vertical lines in **Figure 3-4** (a) and **Figure 3-5** (a) indicate the times where the algorithm detects a new emerging cluster in the S1 and S2 datasets. The graphs in **Figure 3-4** (a) and **Figure 3-5** (a) have initial vertical lines when two clusters have formed. The first vertical line to the right of the origin corresponds to the third cluster detected. We notice that there is a sudden drop in the value of the index after each new cluster appears in the data stream when the algorithm learns the prototype that represents the new emerging cluster. Then, new data are grouped in the latest cluster structure so the index value is almost constant until data points from the next cluster start accumulating. EROLSC flags data points from a new structure as outliers until they are dense enough to be designated as a new cluster.

This increases the index value starting from the knee point as shown in **Figure 3-4** (b). **Figure 3-5** shows the DB index for dataset S2. In this dataset, there is a cluster in the center, which is indexed so that 10 data points are added to it before each of the surrounding clusters appears in the stream. EROLSC detects the central cluster right after the sixth cluster, which can be seen in **Figure 3-5** (b) where there is a close line pair corresponding to clusters 5 and 6. All the clusters have almost similar knee point locations and similar distances from the knee point to the next solid line (the start of a new cluster) except the cluster at the center, as shown in **Figure 3-5** (b). This is because of the way the data stream is fed to the algorithm. The surrounding clusters are presented sequentially, whereas data points in the center cluster are fed 10 data samples at a time after each cluster. This can be useful to understand the data stream, especially in real applications such as activity recognition.



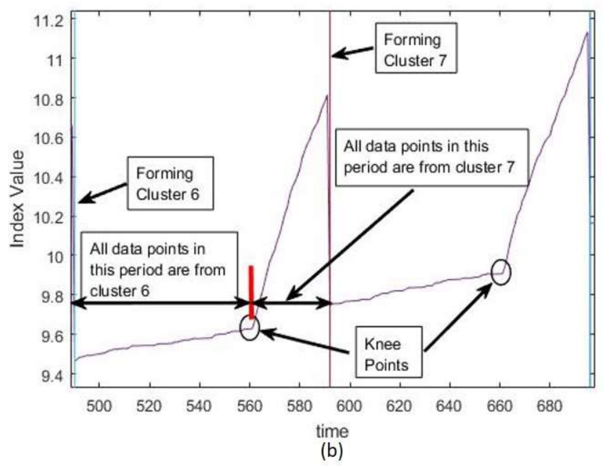


Figure 3-4 a) Effect of membership exponent on DB index on S1 dataset and knee location where the arrow pointing up is the increasing trend as m→1 and the arrow pointing down is the decreasing trend for m ≥ 2, b) Exploded view of (a) from time 480 to time 700.

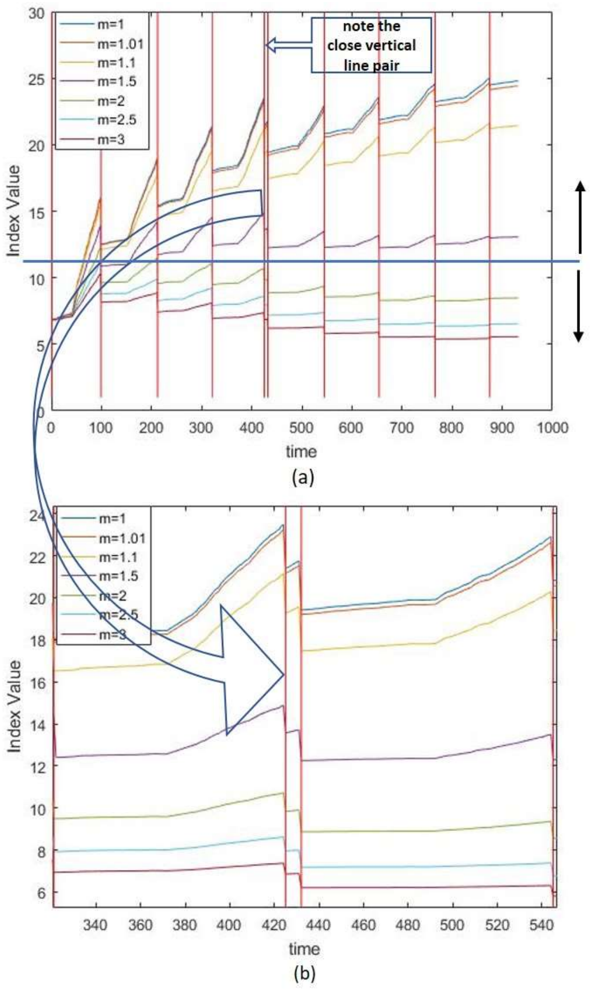
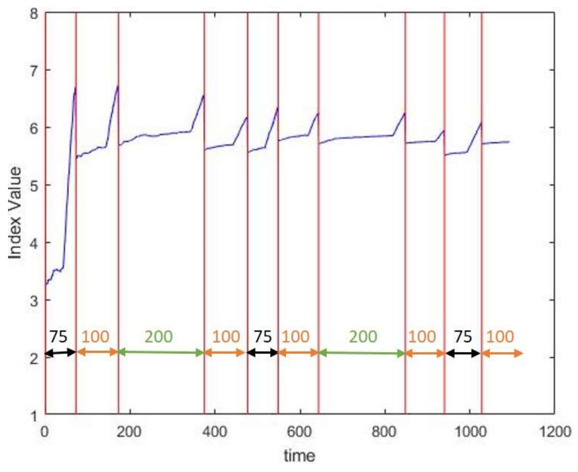


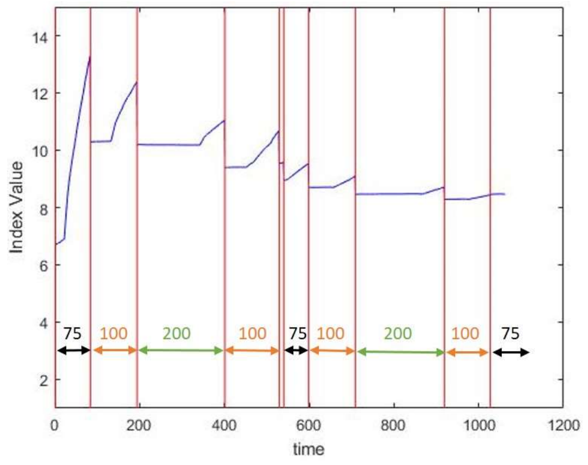
Figure 3-5 a) Effect of membership exponent on DB index on S2 dataset where the arrow pointing up is the increasing trend as m→1 and the arrow pointing down is the decreasing trend for m ≥ 2, (b) Zoom of the region containing the line pair

3.8.2 Effect of Cluster size in the Knee Location

To study the time elapsed when EROLSC starts a new cluster to the appearance of the next knee point, we set the m = 2 and vary the cluster sizes. Instead of having 100 samples/cluster in the two datasets, the clusters sizes are modified to [100, 75, 100, 200, 100, 75, 100, 200, 100, 75, 100]. The last cluster in modified S2 is the one at the center. **Figure 3-6** shows the iiDB for the modified S1 and S2 datasets where the arrows show the time between consecutive clusters and the numbers show the cluster sizes, keeping in mind the calculation of the index started from cluster 2. We notice that it takes longer to reach the knee point for larger clusters because more data points fall into the larger clusters before EROLSC triggers a new cluster. This keeps the iiDB value from increasing. This observation is true for both datasets, as shown in **Figure 3-6** (a) and **Figure 3-6** (b). However, *the distance between the knee point and the next vertical line* (time when the next cluster is detected by the algorithm) is almost constant for both datasets. This reflects the anomaly detection procedure that the online clustering algorithm follows to detect new emerging structure in the data stream. In addition, this might tell us that the clusters are coming from the same Gaussian distribution and/or the data streams are in an ordered fashion. We can see the case of S2 dataset where the middle cluster has shorter duration between the knee point and the next cluster due to the location of its data samples in the streaming data. This suggests that a different change detection mechanism could change the knee location as will be shown in the next section. While we can directly observe the behavior on these 2-D datasets, we note that the patterns shown in **Figure 3-4** hold for similar datasets of high dimension that we tested, allowing us to understand the cluster formation in data that cannot be visualized.



(a)



(b)

Figure 3-6 Effect of different cluster sizes on DB index, a) modified S1, b) modified S2. Compare these to Fig. 4(a) and Fig. 5(a).

3.8.3 Effect of Detection Mechanism in Knee Location

To study the starting time of the knee point and how long it takes to reach the sudden drop in the iiDB value (formation of next cluster), the procedure of spawning a new cluster at the data point is changed in three different ways. The original S1 and S2 datasets are used in this experiment, where all clusters have the same size. To create a new cluster in

EROLSC, the dispersion of the candidate cluster, which is a group of data points from the anomaly list, is computed. If the dispersion value of the candidate cluster is less than a scaled version of the maximum dispersion of the existing clusters, it is accepted as a new emerging structure. The scale value used in EROLSC is set to 3. Three different dispersion scales (1.5, 2 and 3) are used in this experiment, shown in **Figure 3-7** and **Figure 3-8**.

Figure 3-7 (a) shows the results on the S1 dataset where the distance between the knee point and the beginning of the next cluster is larger for a smaller dispersion scale. The reason is that the tolerance of the algorithm to create a new cluster compared to the existing prototypes increases as the dispersion scale increases because it accepts less compact clusters. However, for small values of the dispersion scale, the algorithm requires clusters with smaller dispersion, and hence, it takes a longer time accumulating data points to create the new cluster. This is particularly clear for the case with a dispersion of 1.5 in **Figure 3-7** (b). We can notice the same relationship in the S2 dataset for all clusters except the middle cluster, as in **Figure 3-8**. **Figure 3-8** (b) shows an exploded view of the knee point locations of the middle cluster where there is a noticeable time difference between them for each dispersion scale value.

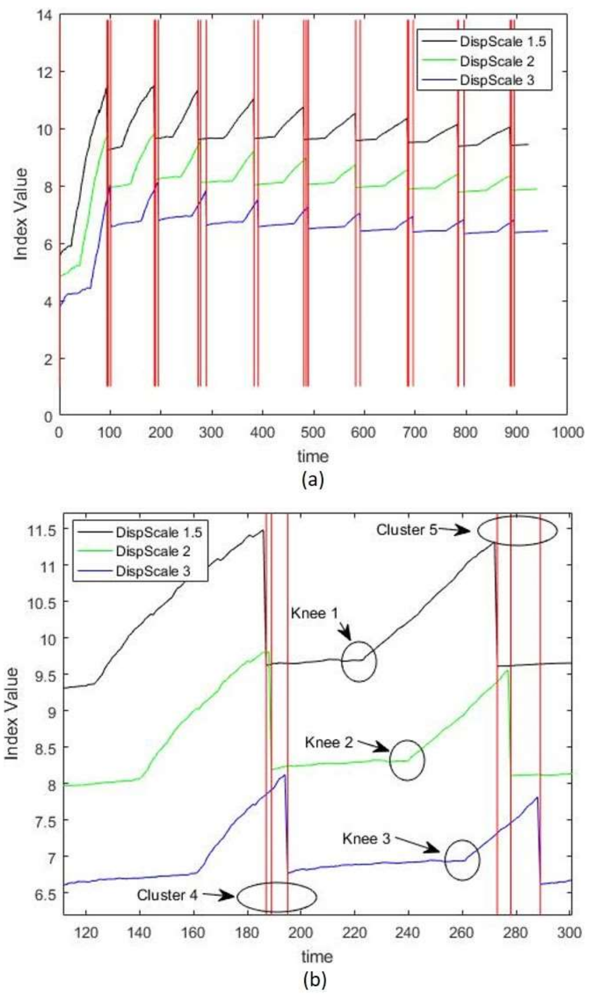


Figure 3-7 a) Effect of dispersion scale on the formation of the knee point for S1 dataset, b) Exploded view of (a) from time 110 to time 300 which shows the effect of dispersion scale on the formation of the knee point.

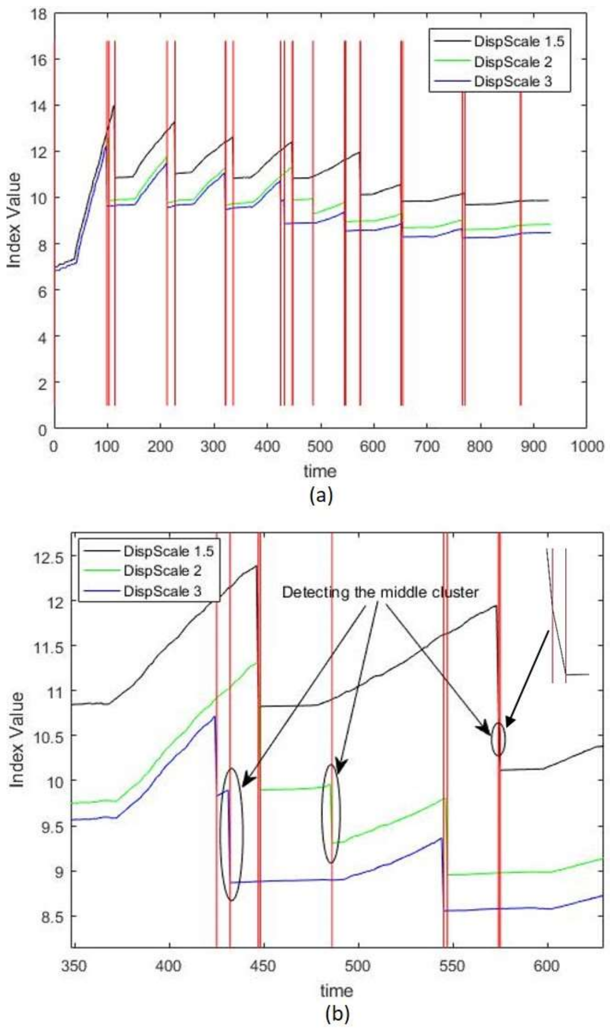


Figure 3-8 a) Effect of dispersion scale on the formation of the knee point for S2 dataset, b) Exploded view of (a) from time 350 to time 625 which shows the effect of dispersion scale on detecting the middle cluster.

The same thing holds for higher dimensions datasets. The iiDB behaves in the same manner regardless of the dataset dimension where it shows the times where new cluster is detected by the algorithm, the distance to the knee location refers to cluster size [27].

3.9 Conclusions

In this paper, we analyzed the information that an incremental form of a relative of the DB index could provide about the behavior of the EROLSC online clustering algorithm.

First, we demonstrated that the version of the iiDB exhibited in [6] at m=2 could be generalized to any value of m≥1. To avoid the increasing trend in the min-optimal index (DB), the exponent of the membership value should be greater than or equal to 2. We discussed how incremental iCVIs can be used to understand the performance of online clustering algorithms with respect to: (i) the appearance of new emerging clusters; (ii) how the clustering algorithm reacts to outlier data samples and evolving clusters, and (iii) the effect of clusters size on the iiCVIs results. The knee point is the point when EROLSC starts accumulating data points for a possible new cluster, but the cluster has yet to be detected by the algorithm. The duration between the knee point and the next cluster is related to the anomaly mechanism used by the online clustering algorithm to detect new emerging structures. In addition, the longer the duration between the cluster detection and the next knee point, the larger the cluster size. The way that data samples in each cluster are streamed has a direct effect on the cluster detection and the knee point location. Clusters that form intermittently along with other clusters are more difficult to see with an iiCVI until before the algorithm detects it (middle cluster in S2 dataset). Therefore, if the source of the data is known a priori (domain knowledge), one can understand the order of the data stream with respect to the clusters. This can be important for applications such as activity recognition or elder adult monitoring.

3.10 References

[1] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. Gama, "Data stream clustering: A survey," ACM Computing Surveys, vol. 46, no. 1, pp. 13–31, Jul. 2013.

[2] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," IEEE Transaction on Knowledge and Data Engineering, vol. 15, no. 3, pp. 515–528, 2003.

[3] M. Ackerman and S. Dasgupta, "Incremental clustering: The case for extra clusters," in Neural Information Processing Systems (NIPS), Montreal, Canada, December 2014.

[4] P. Angelov, Evolving Takagi-Sugeno Fuzzy Systems from Streaming Data (eTS+). John Wiley & Sons, Inc., 2010, pp. 21–50.

[5] M. Moshtaghi, J. Bezdek, and C. Leckie, "Online clustering of multivariate time-series," in Proceedings of SIAMConference on Data Mining, Florida, USA, May 2016.

[6] M. Moshtaghi, J. Bezdek, S. Erfani, C. Leckie, J. Bailey, "Online Cluster Validity Indices for Streaming Data," submitted, TETCI.

[7] O. Arbelaitz, I. Gurrutxaga, J.Muguerza, J.M. Perez, and I. Perona, "An extensive comparative study of cluster validity indices," Pattern Recognition, vol. 46, no. 1, pp. 243 – 256, 2013.

[8] O. Ibrahim, Y. Du, J. Keller, "Extended robust on-line streaming clustering (EROLSC)," acepted at IPMU 2018.

[9] D. L. Davies and D. W. Bouldin, "A cluster separation measure," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 1, no. 2, pp. 224–227, Feb. 1979.

[10] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, On Clustering Massive Data Streams: A Summarization Paradigm. Springer US, 2007, vol. 31, pp. 9–38.

[11] N. Ailon, R. Jaiswal, and C. Monteleoni, "Streaming k-means approximation," in Neural Information Processing Systems 2009, vol. 22, 2009, pp. 10–18.

[12] M. Salehi, C. Leckie, M. Moshtaghi, and T. Vaithianathan, "A relevance weighted ensemble model for anomaly detection in switching data streams," in Proceedings of PAKDD, 2014, pp. 461–473.

[13] M. Moshtaghi, J. Bezdek, and C. Leckie, "Online clustering of multivariate time-series," in Proceedings of SIAMConference on Data Mining, Florida, USA, May 2016.

[14] P. Angelov and X. Zhou, "Evolving fuzzy-rule-based classifiers from data streams," IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pp. 1462–1475, 2008.

[15] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in SIAM Conf. on Data Mining, 2006, pp. 328–339.

[16] P. Kranen, I. Assent, C. Baldauf, and T. Seid, "The clustree: Indexing micro-clusters for anytime stream mining," Knowledge and Information Systems, vol. 29, no. 2, pp. 249–272, 2011.

[17] N. Mozafari, S. Hashemi, and A. Hamzeh, "A statistical approach for clustering in streaming data," Artificial Intelligence Research, vol. 3, pp. 38–45, 2014.

[18] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1, no. 14, pp. 281-297. 1967.

[19] G. W. Milligan and M. C. Cooper, “An examination of procedures for determining the number of clusters in a data set,” Psychome- trika, vol. 50, no. 2, pp. 159–179, 1985.

[20] K. Schermelleh-Engel, M. Moosbrugger, and H. Helfried, “Evaluating the fit of structural equation models: Tests of significance and descriptive goodness-of-fit measures,”Methods of Psychological Research, vol. 8, pp. 23 – 74, 2015.

[21] J. C. Bezdek. A Primer on Cluster Analysis: Four Basic Methods that (Usually) Work, First Edition Design Publ., Sarasota, FL, 2017.

[22] X. L. Xie and G. A. Beni, "A validity measure for fuzzy clustering," IEEE Trans. PAMI, vol.13, no.8, pp. 841-846.

[23] O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.

[24] Krishnapuram R., Keller 1.M., "A possibilistic approach to clustering," IEEE Trans. Fuzzy Syst., vol. I, no. 2, pp. 98-1 10, May 1993.

[25] Miin-Shen Yang; Chien-Yo Lai, "A Robust Automatic Merging Possibilistic Clustering Method," IEEE Transactions on Fuzzy Systems, vol.l9, no.l, pp.26,41 , Feb. 201 1.

[26] S. Araki, H. Nomura, and N. Wakami, “Segmentation of thermal images using the fuzzy c-means algorithm,” in Proceedings of the Second IEEE International Conference on Fuzzy Systems, San Francisco, USA, April 1993, pp. 719–724, 2005.

[27] O. Ibrahim, J. Keller, J. Bezdek "Evaluating Evolving Clusters in Streaming Data with Modified Dunn's Indices," submitted to IEEE Transaction on Evolving Topics in Computational Intelligence, 2018.

Chapter 4: Evaluating Evolving Structure in Streaming Data with Modified Dunn's Indices

Omar A. Ibrahim*, student member, IEEE,* James M. Keller, *life fellow, IEEE*, and James C. Bezdek, *life fellow, IEEE*

Abstract—Dunn's internal cluster validity index is used to assess partition quality and to identify a "best" crisp c-partition of n objects built from static data sets. This index is quite sensitive to inliers and outliers in the input data, so a subsequent study developed a family of 17 generalized Dunn's indices that extend and improve the original measure in various ways. This article presents online versions of two modified generalized Dunn's indices that can be used for dynamic evaluation of evolving (cluster) structure in streaming data. We argue that this method is a good way to monitor the ongoing performance of streaming clustering algorithms and we illustrate several types of inferences that can be drawn from such indices. Streaming clustering algorithms are incremental, process incoming data points only once and then discard them, adapt as the data stream evolves, flag outliers, and most importantly, spawn new emerging structures. We compare the two new indices to the incremental Xie-Beni and Davies-Boudin indices, which to our knowledge offer the only comparable approach, with numerical examples on a variety of synthetic and real datasets.

Index Terms—internal cluster validity, Dunn's index, generalized Dunn's indices, streaming clustering, incremental modified Dunn's indices, incremental Xie-Beni index, incremental Davies-Bouldin index.

4.1 Introduction

Cluster analysis has a long history in pattern recognition. The goal is to find meaningful structure within a group of objects. The ambiguity of the terms “find”, “meaningful”, “structure” and “objects” have given rise to many hundreds (probably thousands) of approaches to address this problem. The traditional view of clustering is that the entire group of objects is available for analysis. Most approaches utilize this fact in their search for structure. With the river of continuous information generated through the internet, or more importantly for us, the ubiquity of low-cost sensors, an alternate view of clustering has appeared, one that acknowledges the temporal aspect of data collection and strives to build and interpret structure as the data arrives. We call this mode streaming clustering. Again, multiple algorithms have been developed to search for evolving structure [1-3], but this paper deals with an emerging topic relative to streaming clustering. What do we mean by streaming cluster analysis? Data, usually vectors in p-dimensional Euclidian space, arrive sequentially. Time should matter, as well as spatial location. For the most part, structure must be formed on the fly as data arrives. This limits the use of classical iterative algorithms where data is revisited. This happens for two reasons. First, decisions need to be made in a timely fashion. In trying to monitor elderly residents, for example, you can’t wait until all the data is available to determine if the resident is behaving in a normal manner or if there is a problem developing. Secondly, in the era of Big Data, it becomes impossible to store all the data for a given problem. Hence, streaming cluster analysis must take on a new and different meaning. So, cluster structure, such as the parameters that define a cluster, must be incrementally updated, outliers/anomalies must be identified, and emerging clusters need to be recognized and formed, all without keeping the data for iterative processing.

Traditional cluster analysis consists of three problems [4]: (i) pre-clustering assessment of tendency (do the data have clusters? If yes, at what value of c, the number of clusters?); (ii) partitioning the data (finding the c clusters?); and (iii) post-clustering cluster validity (are the c clusters found useful?). There are a plethora of papers that deal with the questions for traditional cluster analysis [4, 5-7]. Do these questions apply to the streaming case? If so, have they been addressed? If not, what takes their place?

Problem (ii) is the "easy" part, the energetic activity – find the structure. This is what essentially all of the work on streaming clustering focuses on. It's what we like to do – build algorithms that produce a partition of the data. We will focus on one such algorithm here [8-9] for illustration, but there are several in the literature. Just as we can't (by definition) keep all the data, we also can't keep the partitions, i.e., the crisp or soft assignment of data to the existing clusters. If the premise is that these approaches must scale to big data, then the partition assignments also may be too large to hold and manipulate.

In our opinion, question (i) is moot. How can we predetermine if there is cluster structure in the data, with a guess as to how many, if the data are arriving and must be processed sequentially? The only apriori help for a streaming clustering technique would be problem specific knowledge, but even that should be limited in unsupervised settings. For a streaming clustering algorithm, here are some new questions that we believe must be addressed in problem (ii):

A. How do you start the process?

Initialization takes on a different meaning.

B. How much of the stream do you need in memory?

Could limit application to big data.

C. Do “soft” partition assignments make sense?

Are they useful?

D. Can you deal with outliers?

Can explode the number of clusters; otherwise

one man’s (cluster’s) outlier is another man’s prototype.

E. How are new clusters recognized and formed?

This requires a change detection mechanism

F. Should you merge existing clusters

In feature space? and In time?

G. What about cluster “drift”?

Data comes in sequentially and original cluster can move around.

H. Should we call this type of processing “clustering”?

No groups of data points are saved, so there are no clusters to evaluate. The only remnants of streaming clustering are the entities retained in a cluster footprint.

All streaming clustering algorithms address these questions somehow, many times implicitly.

The real problem is defining a sensible notion to replace problem (iii). How do you tell if the streaming process is working when the input data lie in a high dimensional vector space? This is known for traditional cluster analysis of batch input data as the cluster validity problem. And in traditional clustering, there are many approaches to view the results of a run of your favorite algorithm to determine if a given partition is “good”. But they all require the partition information, and in many cases, the data itself, to be available.

Our axioms for streaming clustering prohibit this situation. What we study in this paper is a new approach to problem (iii), development and analysis of streaming "equivalents" to a class of cluster validity measures. While they don't, and can't really, directly answer question (iii) in a traditional sense, we show that these incremental measures can provide a unique view of the online clustering process and give new insights into the evolving structure seen by the processing algorithm.

Here is an outline of the rest of this article. Section 4.2 discusses the background. Section 4.3 displays related work, which is almost exclusively confined to evaluation in the static data case. Dunn's index and its generalizations are defined and discussed in Section 4.4. Section 4.5 describes the incremental update formula for within cluster dispersion that is essential to Section 4.6, which presents incremental versions of modified forms of DI_{43} and DI_{53}. Section 4.7 describes the streaming clustering algorithm we will use in this study. Section 4.8 is devoted to computational protocols and the data sets used for the numerical experiments that are presented in Section 4.9. Section 4.10 offers our conclusions, and some suggestions for further research.

4.2 Background

In order to be complete, we first discussed the need for streaming clustering and incremental cluster validity indices as an emerging topic in computational intelligence field. Now we discuss the background information about traditional clustering and cluster validity indices.

Let $O = \{o_1,\ldots,o_n\}$ denote n objects (soccer teams, boats, cancer patients, credit reports, etc.). Two kinds of numerical data are used to represent O. Numerical *object* data (feature vector data) $X = \{\mathbf{x}_1,\ldots,\mathbf{x}_n\} \subset \Re^p$, where the coordinates of $\mathbf{x}_i$ are feature values or attributes (e.g., weight, length, intensity, etc.) describing various properties of object o_i, $1 \le i \le n$. A second form of data is relational data, viz., pair-wise dissimilarities represented by values in an $n \times n$ matrix $D = [d_{ij}] = \left[\text{diss}(o_i, o_j) : 1 \le i, j \le n\right]$.

Partitions of the objects are built from sets of label vectors. Let integer c be the number of classes, $1 < c < n$. The crisp, fuzzy/probabilistic, and possibilistic label vectors in $\Re^c$ are:

$$N_{pc} = \{\mathbf{y} \in \Re^c : y_i \in [0,1] \forall i; y_i > 0 \exists i\} \quad ; \text{(possibilistic)} \qquad (1a)$$

$$N_{fc} = \{\mathbf{y} \in N_{pc} : \sum_{i=1}^{c} y_i = 1\} \quad ; \text{(fuzzy/probabilistic)} \qquad (1b)$$

$$N_{hc} = \{\mathbf{y} \in N_{fc} : y_i \in \{0,1\} \forall i\} \quad ; \text{(crisp)} \qquad (1c)$$

Crisp (or hard) c-partitions of n objects are represented as $c \times n$ matrices whose kth column ($\mathbf{U}^{(k)}$) comes from N_{hc} in (1c). Specifically, the hard c-partitions of O are:

$$\begin{aligned} M_{hcn} = \{U \in \Re^{cn} : \mathbf{U}^{(k)} \in N_{hc}, 1 \le k \le n; \\ ; \sum_{i=1}^{c} u_{ik} = 1 \forall k; \sum_{k=1}^{n} u_{ik} > 0 \forall i\} \end{aligned} . \qquad (2)$$

For feature vector data, we will use an equivalent way to represent a partition $U \in M_{hcn}$ of X in terms of the c crisp subsets $\{X_i \subset \Re^p\}$ that form the partition, written as $U \leftrightarrow X = \bigcup_{i=1}^{c} X_i ; \varnothing = X_i \bigcap_{i \ne j} X_j$. The cardinalities (sizes) of the c clusters are $|X_i| = n_i, i = 1, \ldots, c$, so that $\sum_{1 \le i \le c} n_i = n$.

We identify a "best partition" U using values of scalar measures of partition quality called *cluster validity indices* (CVIs). The most important distinction for CVIs is whether the index is *internal* (uses only the information available from the algorithmic clustering outputs and possibly the input data); or *external* (uses the algorithmic clustering outputs, the input data, plus additional "outside" information such as a ground truth partition that labels subsets in the data). *Dunn's index* (DI, [10]) and the *generalized Dunn's indices*, (GDIs, [11]) are internal CVIs designed to select a best partition amongst a set of *candidate partitions*,

$$CP = \{U \in M_{hcn} : c_m \le c \le c_M\} \tag{3}$$

where c_m and c_M are the minimum and maximum values of the integer c associated with partitions in CP. Let $\mathcal{V} : M_{hcn} \to \Re$ denote any crisp, internal CVI. Each element of CP is evaluated by $\mathcal{V}$, resulting in the set of values

$$\mathcal{V}_{CP} = \{\mathcal{V}(U) \in \Re : U \in CP\} \tag{4}$$

The "best" partition in CP *in the sense of measure* $\mathcal{V}$ is the one that yields the maximum or minimum value in $\mathcal{V}_{CP}$, according as $\mathcal{V}$ is *max-optimal* or *min-optimal*. There are some CVIs that operate a bit differently by instead indicating a preferred partition with a "jump" in successive values, but these are not used in the sequel.

Many clustering algorithms produce soft clusters that in the most general case are *possibilistic c-partitions*, which are $c \times n$ matrices in

$$\begin{aligned} M_{pcn} = \{U \in \Re^{cn} : \mathbf{U}^{(k)} \in N_{pc}, 1 \le k \le n; \\ ; \sum_{i=1}^{c} u_{ik} \le c \,\forall\, k; \sum_{k=1}^{n} u_{ik} > 0 \,\forall\, i\} \end{aligned} \tag{5}$$

The *fuzzy/probabilistic c-partitions* are a subset of M_{pcn}

$$M_{fcn} = \{U \in M_{pcn} : \sum_{i=1}^{c} u_{ik} = 1 \forall k\} \quad . \qquad (6)$$

Evidently $M_{hcn} \subset M_{fcn} \subset M_{pcn}$. The set CP in (2) and the domain of CVIs in (3) for the 17 GDIs is M_{hcn} , the set of crisp c-partitions of n objects.

4.3 Related Work

Many books on cluster analysis contain at least one chapter on static cluster validity [4, 5-7]. Dunn's index [10] is an internal CVI for assessing cluster validity for crisp c-partitions of a data set X. Surveys on crisp cluster validity indices (CVIs) that compare various validation schemes in one way or another began to appear in the 1980s [12]. Milligan and Cooper [13] compared 30 validity tests using partitions generated by four hierarchical clustering methods, and their paper is considered the classic reference on "best-c" studies of internal CVIs. Gurrutxaga et al. [14] present a very thorough critique of Milligan and Cooper's "best c" methodology.

Dimitriadou et al. [15] presented a survey of 15 internal CVIs in 2002. Arbelaitz et al. [16] published a very ambitious comparison of 30 internal CVIs for crisp c-partitions that channels the Milligan-Cooper style. Three crisp clustering algorithms were used to populate CP in their study. Hubert and Arabie [17] initiated the standardization of classical paired-comparison indices for external validation of crisp clusters. This seminal work also provides a general introduction to statistical normalization of CVIs to remove bias

Nguyen et al. [18] presented a "best-c" study of similarity measures and distance-based functions that compare pairs of crisp partitions using external information-theoretic CVIs. They identify a total of 26 measures that are subdivided into 10 similarity measures and 16 distance measures.

Dunn's index has been related to visual assessment methods such as the improved visual assessment of tendency (iVAT) algorithm in [18]. A recent study using neuron spike data relates Dunn's index to both iVAT and SL [19]. And various commercial software packages such as the spike extraction and sorting software (Offline Sorter, Plexon Inc, Dallas, TX) report Dunn's index as part of their default cluster validity statistics [20].

The use of any of the 17 GDIs is restricted to hindsight processing of partitions built from static data sets. While the recent trend towards online cluster analysis in data streams to detect anomalies, drift, and evolving cluster structure has led to many streaming clustering algorithms [1], there has been very little work in developing strategies that parallel this growth in the area of cluster validity. Reference [21] is a recent paper that develops the theory and some numerical examples that illustrate incremental versions of the Davies-Bouldin [DB, 22] and Xie-Beni [XB, 23] validity indices. In [24], the authors extended the incremental DB (iDB) index to a more general version and illustrated how the iDB can be used to analyze and understand the performance of a streaming clustering algorithm. Recently, the use of iCVIs with big data was introduced in [25] which show the need of iCVIs for high dimensional streaming data monitoring. These incremental CVIs can be used to monitor cluster formation and algorithmic performance of streaming algorithms. This article develops a similar method for modifications of two of the GDIs, viz., DI_{43} and DI_{53}, and compares their performance to the existing incremental CVIs discussed in [21].

4.4 Generalized Dunn's Indices

Dunn's index is based on the geometrical premise that "good" sets of clusters are compact and well separated. To quantify this index Dunn let X_i and X_j be non-empty subsets of $\Re^p$, and let $d:\Re^p \times \Re^p \to \Re^+$ be any metric on $\Re^p \times \Re^p$. Dunn based his index on the standard definitions of the diameter (dia) of X_k and the set distance (δ) between X_i and X_j.

$$\mathrm{dia}(X_k \mid d) = \underbrace{\max}_{\mathbf{x},\mathbf{y}\in X_k} \{ d(\mathbf{x},\mathbf{y})\} \tag{7}$$

$$\delta(X_i, X_j \mid d) = \delta_{SL}(X_i, X_j \mid d) = \min_{\substack{\mathbf{x}\in X_i \\ \mathbf{y}\in X_j}} \{d(\mathbf{x},\mathbf{y}\} \tag{8}$$

The notation (*|d) for δ and dia indicate that these depend only on d. For any partition $U \leftrightarrow X = X_1 \cup \ldots X_i \cup \ldots X_c$, Dunn defined the separation index of U as follows:

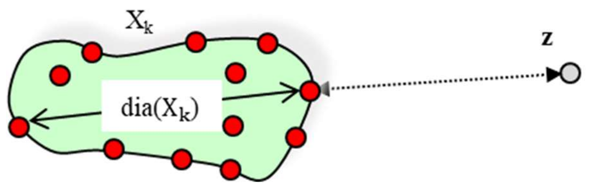
$$\mathrm{DI}(U \mid d, \delta, \mathrm{dia}) = \frac{\underbrace{\min}_{i\neq j} \{\delta(X_i, X_j \mid d)\}}{\underbrace{\max}_{1\leq k\leq c} \{ \mathrm{dia}(X_k \mid d)\}} \quad . \tag{9}$$

So DI depends on three functions, $\{d, \delta, \mathrm{dia}\}$. The most common metric for (d) in the numerator and denominator of DI ~ i.e., in (7), (8) and (9) ~ is Euclidean distance d_E, but Dunn formulated his theory for an arbitrary metric on the input space. The diameter $\mathrm{dia}(X_k \mid d)$ at (6) which appears in the denominator of (8) is a measure of scatter volume for cluster X_k. Compact clusters will have smaller diameters than ones that are more dispersed about their mean vectors. A set of clusters is relatively compact when the largest of its c diameters and hence, the denominator in (7), is small.

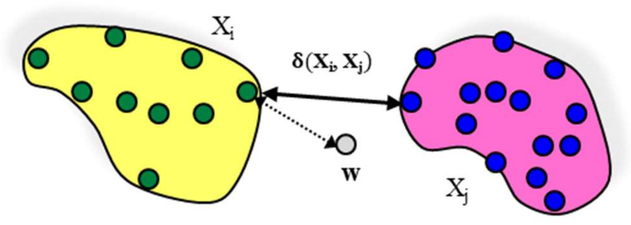
The set distance $\delta(X_i, X_j \mid d)$ shown in equation (7) between the crisp clusters X_i and X_j in U is used by the hierarchical *single linkag*e (SL) clustering algorithm [4-5], so is often called the single linkage set distance. The larger $\delta(X_i, X_j \mid d)$ is, the better separated are X_i and X_j. Taking the double minimum in the numerator identifies the pair of clusters that are *least* well separated so as the c clusters become more separated, the numerator in (4) grows.

So, the geometric objective of Dunn's index is to maximize intercluster distances (big numerators = good separation) whilst minimizing intracluster distances (small denominators = compact clusters). Larger values of DI intuitively correspond to better clusters in the sense of this measure. The partition U* that maximizes DI over a set of candidate partitions in CP is taken as the (Dunn's index) optimal set of clusters. Consequently, DI is called a *max-optimal* internal CVI. The range of DI is $(0, \infty)$, and DI is undefined when c = 1 ($U_{1\times n} = \mathbf{1}_n$) and c = n ($U_{n\times n} = I_n$).

Figure 4-1 depicts the classical meaning of equations (6) and (7) when d is chosen as *Euclidean distance*, d = d_E.



(a) The classical diameter $\mathrm{dia}(X_k)$ of X_k



(b) The classical (SL) distance $\delta(X_i, X_j)$ between X_i and X_j

Figure 4-1. SL set distance and classical diameter for d = d_E

Dunn's index has a well-known flaw, viz., sensitivity to anomalies, which can render it ineffective when clusters have outliers and/or inliers. **Figure 4-1** (a) illustrates that the addition of the point **z** to X_k can double its diameter, $\mathrm{dia}(X_k \cup \{\mathbf{z}\}) = 2\mathrm{dia}(X_k)$. Similarly, adding inlier **w** to X_i in **Figure 4-1** (b) significantly decreases the distance between X_i and X_j. Thus, a single anomaly can alter the numerator and/or denominator of Dunn's index by orders of magnitude.

To address this issue, a family of 17 *generalized Dunn's indices* (GDIs) were defined and analyzed in [11]. These indices take the general form

$$\mathrm{GDI}_{ab}(U \mid d, \delta_a, \mathrm{dia}_b)) = \frac{\underbrace{\min}_{i \neq j} \{\delta_a(X_i, X_j \mid d)\}}{\underbrace{\max}_{1 \le k \le c} \{\mathrm{dia}_b(X_k \mid d)\}}, \tag{10}$$

where $a \in \{1, \ldots, 6\}, b \in \{1, 2, 3\}$. Generally speaking, the geometric objective the 17 GDIs is preserved by the definition at (10). Our notation for the 17 indices represented by (10) is $\mathrm{GDI}_{ab}(U \mid d)$. Equation (10) reduces to (9) when a = b = 1, i.e., $\mathrm{DI} = \mathrm{GDI}_{11}$.

Some of the crisp GDIs have done well in comparative studies [7, 18, 19]. For example, the study in [17] ranked GDI_{33} 8-th among 40 competing internal CVIs, scoring 389 correct instances in 432 clustering scenarios, whereas GDI_{11} (the original Dunn's index at (9)) was ranked 29 out of 40 in this same study. The indices that are the object of this investigation are GDI_{43} and GDI_{53}, which ranked 10th and 12th in [19]. These are the only two GDIs that have a term which is similar to the core equation for cluster dispersion (equation (21) below).

GDI_{43} and GDI_{53} are selected in this paper because they can be extended to the incremental fashion. The indices GDI_{43} and GDI_{53} are defined by the following set distances and diameter function:

$$\delta_4(X_i, X_j) = d(\mathbf{v}_i, \mathbf{v}_j) \tag{11}$$

$$\delta_5(X_i, X_j) = \frac{\sum_{\mathbf{x} \in X_i} d(\mathbf{x}, \mathbf{v}_i) + \sum_{\mathbf{y} \in X_j} d(\mathbf{y}, \mathbf{v}_j)}{n_i + n_j} \tag{12}$$

$$\mathrm{dia}_3(X_k) = \left(\frac{2 \sum_{\mathbf{x} \in X_k} d(\mathbf{x}, \mathbf{v}_k)}{n_k} \right) \tag{13}$$

In (11), (12) and (13), $\forall k : \mathbf{v}_k = \sum_{\mathbf{x} \in X_k} \mathbf{x}/n_k$, are the means of the crisp clusters. The diameter at (13) is, for Euclidean distance, the diameter of a hyperball centered at the mean vector of X_k whose length is the average distance between the points in X_k and its cluster center.

The common factor in these choices for δ and dia is the term for the sum of distances between the points in a cluster and its cluster center, $\sum_{\mathbf{x}\in X_k} d(\mathbf{x},\mathbf{v}_k)$, or, when d is a norm metric, $\sum_{\mathbf{x}\in X_k} \|\mathbf{x}-\mathbf{v}_k\|$. This is *almost* the key term for incremental updates of the Xie-Beni and Davies-Bouldin CVIs that was derived in [2]. The formulation in [2] was based on the *squares* of *Euclidean* distances, $\sum_{\mathbf{x}\in X_k} \|\mathbf{x}-\mathbf{v}_k\|_2^2$. To accommodate this difference, we will use Euclidean distance in (10), and we replace the distances shown in (12) and (13) with squared Euclidean distances. For convenience, we drop the subscript "2" which indicates this as the Euclidean norm. We call the new CVIs based on these modifications *modified Dunn's indices* (MDIs), notated as MDI_{43} and MDI_{53}:

$$\delta_4(X_i,X_j) = \|\mathbf{v}_i - \mathbf{v}_j\| \qquad (14)$$

$$\delta M_5(X_i,X_j) = \frac{\sum_{\mathbf{x}\in X_i} \|\mathbf{x}-\mathbf{v}_i\|^2 + \sum_{\mathbf{y}\in X_j} \|\mathbf{y}-\mathbf{v}_j\|^2}{n_i + n_j} \qquad ;(15)$$

$$\mathrm{dia}(M_3(X_k)) = \left(2\sum_{\mathbf{x}\in X_k} \|\mathbf{x}-\mathbf{v}_k\|^2 \big/ n_k\right) \qquad ; \quad (16)$$

The modified Dunn's indices corresponding to these choices for set distance and diameter are

$$\mathrm{MDI}_{43} = \frac{\underbrace{\min}_{i\neq j}\{\delta_4(X_i,X_j)\}}{\underbrace{\max}_{1\leq k\leq c}\{\mathrm{dia}(M_3(X_k))\}} \qquad ; \quad (17)$$

$$\mathrm{MDI}_{53} = \frac{\underbrace{\min}_{i\neq j}\{\delta M_5(X_i,X_j)\}}{\underbrace{\max}_{1\leq k\leq c}\{\mathrm{dia}(M_3(X_k))\}} \qquad . \quad (18)$$

Indices MDI_{43} and MDI_{53} are new internal validity indices for crisp clusters. The only difference between these two MDIs and the original GDIs that precede them is the squaring of distances between the input data and the cluster centers. We believe ~ but will not test here ~ that evaluation of any set of crisp partitions of static data made by MDI_{43} and MDI_{53} will almost always be identical to results obtained with GDI_{43} and GDI_{53}. The main objective of this article is the adaptation of (17) and (18) for incremental evaluation of evolving cluster structure in streaming data. Our notation for the incremental versions will be iMDI_{43} and iMDI_{53} (the "i" meaning "incremental," so that iMDI stands for i*ncremental modified Dunn's index*).

4.5 Incremental form of Cluster Dispersion

Suppose we have streaming vector data inputs, say $\mathbf{x}_k \in \Re^p, k = 1, 2, \ldots, n$, beginning at k=1 (e.g., time 1), and we have chosen Euclidean distance for the norm metric in the input space. As each $\mathbf{x}_k$ arrives, let us assume that a streaming clustering algorithm assigns crisp membership labels to it in c clusters, $\mathbf{U}^{(k)} = \{u_{i,k} : 1 \le i \le c\} \in N_{hc}$. After n inputs, we have:

(i) c, the number of current crisp clusters; (19a)

(ii) Cluster prototypes $V_n = \{\mathbf{v}_{1,n}, \ldots, \mathbf{v}_{c,n}\} \subset \Re^p$; (19b)

(iii) a $c \times c$ distance matrix $D(V_n) = [\|\mathbf{v}_{in} - \mathbf{v}_{jn}\|]$; (19c)

(iv) cardinalities, $|X_i| = n_i, i = 1, \ldots, c$; $\sum_{1 \le i \le c} n_i = n$; (19d)

(v) current values of $\mathrm{iMDI}_{43,n}$ and $\mathrm{iMDI}_{53,n}$. (19e)

We do NOT retain the historical input data. After observing $\mathbf{x}_{n+1}$, we update the current values of $\mathrm{iMDI}_{43,n}$ and $\mathrm{iMDI}_{53,n}$ by finding incremental changes,

$$\mathrm{iMDI}_{43,n+1} = \mathrm{iMDI}_{43,n} + \Delta(\mathrm{iMDI}_{43,n}) ; \tag{20a}$$

$$\mathrm{iMDI}_{53,n+1} = \mathrm{iMDI}_{53,n} + \Delta(\mathrm{iMDI}_{53,n}) ; \tag{20b}$$

We will find the increments in (20) using a method developed by Moshtaghi et al. [2]. They defined the *fuzzy within cluster dispersion* of the i-th cluster in fuzzy partition $U \in M_{fcn}$ after n streaming inputs as

$$\hat{C}_{i,n} = \sum_{j=1}^{n} (u_{ij})^2 \left\| \mathbf{x}_j - \mathbf{v}_{i,n} \right\|^2 . \tag{21}$$

The quantity in (21) appears in many objective functions and CVIs. It is a measure of the compactness of the i-th fuzzy cluster about its centroid. Ibrahim et al. [24] noticed that the factor $(u_{ij})^2$ in (21) behaves like a constant in the derivation given in [21], i.e., it factors out of all terms in the derivation. Consequently, they demonstrated that a more general form of (21), viz.,

$$C_{i,n} = \sum_{j=1}^{n} (u_{ij})^m \left\| \mathbf{x}_j - \mathbf{v}_{i,n} \right\|^2 ; m \geq 1 . \tag{22}$$

could be used in Algorithm 2 of [21]. We will use this more general form of the update method in our Algorithm 1 by specifying its pseudocode for any value of $m \geq 1$.

Mostaghi et al. [21] wrote the incremental form of the compactness function at (21) as $\hat{C}_{i,n+1} = \hat{C}_{i,n} + \Delta\hat{C}_{i,n}$. We reproduce their algorithm (with the modification that $m \geq 1$) as our Algorithm 1, which yields, for all i, the updated value of the function $C_{i,n+1}$ in (22) as a function of $\mathbf{x}_{n+1}$. After processing the data points in the initial wondow S, we have (m), the number of clusters c, a label vector $\mathbf{U}^{(1)} \in N_{hc}$ and initial prototypes $V_1 = \{\mathbf{v}_{1,1}, \ldots, \mathbf{v}_{c,1}\}$.

First, we compute initial values for three quantities needed to initialize Algorithm 1:

```
Input: $\mathbf{U}^{(1)}, V_1, m \in [1, \infty); c \in \{1, \ldots, n\}$
For i =1 to c
    $G_{i,1} = (0, 0, \ldots, 0)^T = \boldsymbol{\theta} \in \Re^p$
    $M_{i,1} = |v_{1,i}|$
    $C_{i,1} = 0$
Next i
```

Beginning with the new arriving data point, streaming inputs can be sent to Algorithm 1, which uses $\mathbf{x}_{n+1}$ to produce the inputs shown:

```
Data: $m, c, V_n, \mathbf{U}^{(n+1)}, V_{n+1}, \forall i : G_{i,1}, M_{i,1}, C_{i,1}$
Input: $\forall i : G_{i,n}, M_{i,n}, C_{i,n}$
Output: $\forall i : G_{i,n+1}, M_{i,n+1}, C_{i,n+1}$
For i = 1 to c
    $Q_{i,n+1} = (\mathbf{v}_{i,n} - \mathbf{v}_{i,n+1})^T G_{i,n}$
    $B_{i,n+1} = \|\mathbf{v}_{i,n} - \mathbf{v}_{i,n+1}\|^2$
    $A_{i,n+1} = (u_{i,n+1})^m \|\mathbf{x}_{n+1} - \mathbf{v}_{i,n+1}\|^2$
    $C_{i,n+1} = C_{i,n} + A_{i,n+1} + M_{i,n} B_{i,n+1} + 2Q_{i,n+1}$
    $G_{i,n+1} = G_{i,n} + M_{i,n}(\mathbf{v}_{i,n} - \mathbf{v}_{i,n+1}) + (u_{i,n+1})^m (\mathbf{x}_{n+1} - \mathbf{v}_{i,n+1})$
    $M_{i,n+1} = M_{i,n} + (u_{i,n+1})^m$
Next i
```

Algorithm 1. Incremental compactness for $\mathbf{x}_{n+1}$ [2]

In the next section we develop incremental versions of the two iMDIs that were in (20).

4.6 Incremental Modified Dunn Indices

If the label vectors formed by the streaming clustering algorithm are crisp as in equation (1c), every value of m in (22) will result in the same weights, so there is no loss in using the generalized form of the modified indices. After n inputs have been processed, producing the current (not ordinarily saved, but it could be) partition $U_n \in M_{hcn}$ and current prototypes V_n, we have, substituting (22) into (17) and (18) the values:

$$iMDI_{43,n} = \frac{\underbrace{\min}_{i \neq j}\{\|\mathbf{v}_{i,n} - \mathbf{v}_{j,n}\|\}}{\underbrace{\max}_{1 \le k \le c}\left\{\frac{2C_{k,n}}{n_{k,n}}\right\}} \tag{23}$$

$$iMDI_{53,n} = \frac{\underbrace{\min}_{i \neq j}\left\{\frac{C_{i,n} + C_{j,n}}{n_{i,n} + n_{j,n}}\right\}}{\underbrace{\max}_{1 \le k \le c}\left\{\frac{2C_{k,n}}{n_{k,n}}\right\}} \tag{24}$$

We don't have the matrix U_n nor the data $\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$, but we do have the current prototypes V_n, and the current values of $iMDI_{43,n}$ and $iMDI_{53,n}$ at (23) and (24). Now $\mathbf{x}_{n+1}$ arrives. Without loss of generality, assume that the online clustering algorithm produces a crisp label vector $\mathbf{U}^{(n+1)} \in M_{hcn}$. Suppose $u_{j,n+1} = 1$; then $\mathbf{v}_{j,n} \to \mathbf{v}_{j,n+1}$ is updated, and the other c-1 prototypes are unchanged, resulting in the new prototypes V_{n+1}. This is the case, for example, in hard c-means (aka k-means, [4]). Algorithm 1 provides us with, $\{C_{i,n+1} : 1 \le i \le c\}$. This set, along with V_{n+1}, is all we need to compute the incremental updates to (23) and (24):

$$\Delta(iMDI_{43,n}) = \frac{\underbrace{\min}_{i \neq j}\{\|\mathbf{v}_{i,n+1} - \mathbf{v}_{j,n+1}\|\}}{\underbrace{\max}_{1 \le k \le c}\left\{\frac{2C_{k,n+1}}{n_{k,n+1}}\right\}} \tag{25}$$

$$\Delta(iMDI_{53,n}) = \frac{\underbrace{\min}_{i \neq j}\left\{\frac{C_{i,n+1} + C_{j,n+1}}{n_{i,n+1} + n_{j,n+1}}\right\}}{\underbrace{\max}_{1 \le k \le c}\left\{\frac{2C_{k,n+1}}{n_{k,n+1}}\right\}} \tag{26}$$

Combining (20a) and (25) yields the incremental form of $iMDI_{43,n+1}$; combining (20b) with (26) yields $iMDI_{53,n+1}$. We will compare the performance of these two max-optimal iMDIs to the incremental min-optimal DB and max-optimal XB indices developed in [21]:

$$iDB_{n+1} = \frac{1}{c}\sum_{i=1}^{c}\underbrace{\max}_{j,j\neq i}\left\{\frac{\frac{C_{i,n+1}}{M_{i,n+1}}+\frac{C_{j,n+1}}{M_{ij,n+1}}}{\left\|v_{i,n+1}-v_{j,n+1}\right\|^2}\right\} \tag{27}$$

$$iXB_{n+1} = \frac{\sum_{i=1}^{c}C_{i,n+1}}{(n+1)\underbrace{\min}_{i\neq j}\{\left\|\mathbf{v}_{i,n+1}-\mathbf{v}_{j,n+1}\right\|\}}. \tag{28}$$

The incremental updates shown at (27) and (28) depend on $C_{i,n+1}$ at (22) extracted from Algorithm 1. While the authors of [21] used $\hat{C}_{i,n+1}$, m=2 as in (21), we will use the more general form of C in (22) in our experiments. In the next section we describe the online clustering algorithm used for the experiments.

4.7 The streaming clustering algorithm

This section briefly describes the MUSC online clustering algorithm developed in [8] and extended in [9]. The main idea of MUSC is based on combining possibilistic and fuzzy clustering and *Gaussian Mixture Models* (GMM). The *Possibilistic C-Means* (PCM) [26] and the *Automatic Merging Possibilistic Clustering Method* (AMPCM) [27] are combined to cluster the buffered data stream in a window S to initialize the GMMs. Due to the tendency of PCM to search for a dense region in the dataset by producing coincident clusters, it is mainly employed in MUSC to detect outliers in the initialization window, S, and for creating a new cluster. The number of clusters that PCM looks for is set to a large number (for example if 100 data samples are in S, we look for 10 clusters). Data points with low typicalities to all existing prototypes are moved to the anomaly list and AMPCM is applied to the noise-free data in S. The clustering result of AMPCM is used to initialize a GMM. When a new data point x_{n+1} arrives at time n+1, its Mahalanobis distance to the means in the GMM is computed as in (29). If the minimum distance falls within a pre-specified threshold (T), x_{n+1} is incorporated into the winning Gaussian cluster. The mean and covariance of the winning Gaussian are incrementally updated using (30) and (31).

$$d = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{29}$$

$$\mu_{new} = \mu_{old} + \frac{x_{n+1} - \mu_{old}}{|\mu_{new}|} \tag{30}$$

$$\Sigma_{new} = \frac{(|\mu_{new}| - 1) * \Sigma_{old} + (x_{n+1} - \mu_{old})^T (x_{n+1} - \mu_{old})}{|\mu_{new}|} \tag{31}$$

where $|\mu_{new}|$ is the new cardinality of the winning cluster, μ and Σ are mean and covariance matrix of the cluster.

If the minimum distance exceeds the threshold, the new data point x_{n+1} is flagged as an outlier and saved in the anomaly list. The points in the anomaly history may or may not indicate the emergence of a new cluster. We track changes in the anomaly list in two different ways. First, we check if any point in the list could fit in one of the existing clusters by computing the Mahalanobis distance between the outliers and updated cluster centers. Points are assigned to their closest Gaussians if they are within a pre-specified threshold (T) (the cluster has “grown” to incorporate what was an outlier before). Second, we cluster the outliers following the same approach to initialize the GMM at the beginning, where we look for multiple emerging structures. See [8] for a more detailed description of the basic algorithm, along with details on initialization, selecting the threshold values, and new cluster formation. The historical footprint of the evolving structure is represented by the means, covariance matrices, and cluster cardinalities, as well as outlier data points. After updating the Gaussian parameters, x_{n+1} is discarded.

4.8 Data, Protocols, and Numerical Experiments

Synthetic and real-life datasets are used in the evaluation, all of which are presented in a streaming on-line fashion. The first synthetic dataset is S1, which has 1100 instances in 2-dimensional space generated from 11 Gaussian distributions distributed along a line as in **Figure *4-2*** (a). The static version of S1 has 11 clusters (100 samples each) as can be seen in **Table 4-1**. The dataset is ordered such that data vectors from cluster 1 arrive before data points from cluster 2 and so on. The second synthetic dataset is S2, which (in batch form) has 1100 data points with 11 clusters. Each cluster has 100 samples and the clusters are arranged in a circular shape with one cluster in the middle, as seen in **Figure *4-2*** (b). The data samples are ordered so that data points from cluster 1 arrive first. After that, 10 data points from the center cluster arrive, followed by vectors from cluster 2. A similar pattern is repeated for the remaining clusters. Data sets S1 and S2 are similar to two data sets used by the authors of [21, 24].

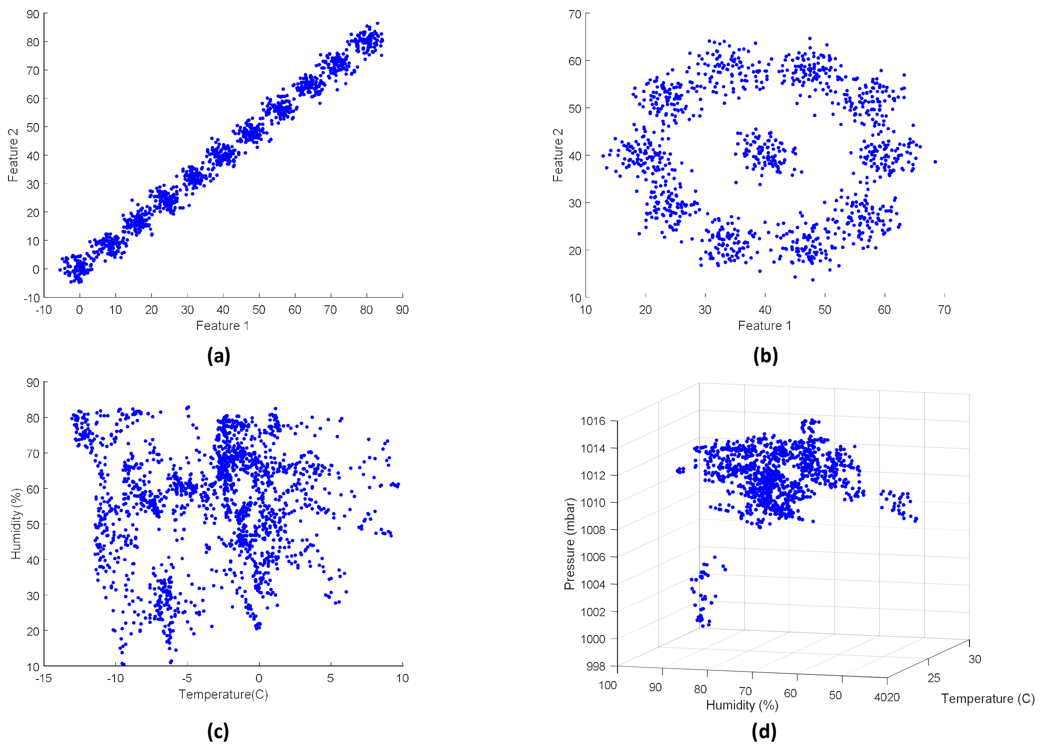


Figure 4-2. Scatter plots of four datasets: a) S1, b) S2, c) Le Genepi weather data, and d) Heron Island weather data

The first real-life dataset used in our evaluation is the LG dataset, which is a collection of weather station nodes in the Le Genepi (LG) region in Switzerland [28]. We use two weeks of data at node 18 starting from October 10, 2007. Average surface temperature (T) and humidity (H) readings over 10-minute intervals were used to create a two-dimensional feature vector dataset $\{x_i = \{(T_i, H_i)\}$. The scatter plot of the data can be seen in **Figure *4-2*** (c). The scatter plot does not provide clear visual evidence of clusters in the static, batch form of the LG data. But, the imagery information from the site shows that there is a snowy day during the two-week period. A windy and cold day precedes the snow. For that reason, we consider the LG data to have three different events: sunny days before and after the snow, cold front moving in, and the snowy day. We try multiple distance threshold values and select the best results (number of clusters) for comparative purposes. The last dataset is the Heron Island Weather station dataset. This is a real-life dataset collected from the Heron Island weather station on the Great Barrier Reef, Australia [29]. The Heron Island data has three features: air temperature (T), air humidity (H), and air pressure (P). The data were collected every 10min from 9:00 am to 3:00 pm each day for the 30 days beginning 21-Feb-2009 and ending 22-Mar-2009. This Heron Island dataset was studied in [29] using ellipsoidal chains for anomaly detection. In this dataset, there are two (visual) abnormal patterns (small clusters). The first abnormal cluster occurs from data points 445 to 476 (5th March 9.00 AM to 2.10 PM) and the second cluster starts from data points 593 to 629 (9th March). The anomalous data for March 9 correspond to the weather event "Cyclone Hamish" . The smaller anomalous cluster on March 5 is due to an unusual weather variation in T and H having regular P. The three-dimensional scatter plot in **Figure** ***4-2*** (d) shows the Heron Island dataset. It is possible that some observers would identify four visual clusters in this static view of the Heron Island data, where two of the four clusters represent the normal pattern and the other two (small clusters) represent the abnormal weather patterns. Table 1 summarizes the four datasets. The value of m used in (22) is m=2 except in the last experiment, where we vary it as $m \in \{1, 1.01, 1.1, 2, 2.5, 3\}$ to study its effect on the iCVIs involved.

4.9 Numerical Experiments

Rapid changes in the iCVIs are related to variations in the separation and cohesion of the clusters produced by the clustering algorithm because they measure cohesion and separation. Therefore, an abrupt change in the value of an iCVI value usually relates to the appearance of a new cluster in the streaming data. The behavior of the iCVI can be used for change-point detection in the streaming data. An iCVI monitors the labels produced by the online clustering algorithm each time a new data point arrives, and when the algorithm detects a new structure (significant change in the cohesion-separation values), the value of the iCVI spikes, indicating a change-point in the data stream. Due to the variation in the values of the four iCVIs, we display each index in a different figure where we highlight two aspects of the values. The first trait relates to the appearance of a new structure (cluster) in the data stream (sudden increase or decrease in the index value). The second aspect is the performance of the clustering algorithm after detecting the new cluster (the reduction of the index value after the abrupt increase).

Table 4-1. Summary characteristics of the datasets used in the evaluation.

Dataset	**n**	**p**	**c**	**# labeled subsets**
S1	1100	2	11	11
S2	1100	2	11	11
Weather	1817	2	3	Our Estimate
Heron	1110	3	4	Our Estimate

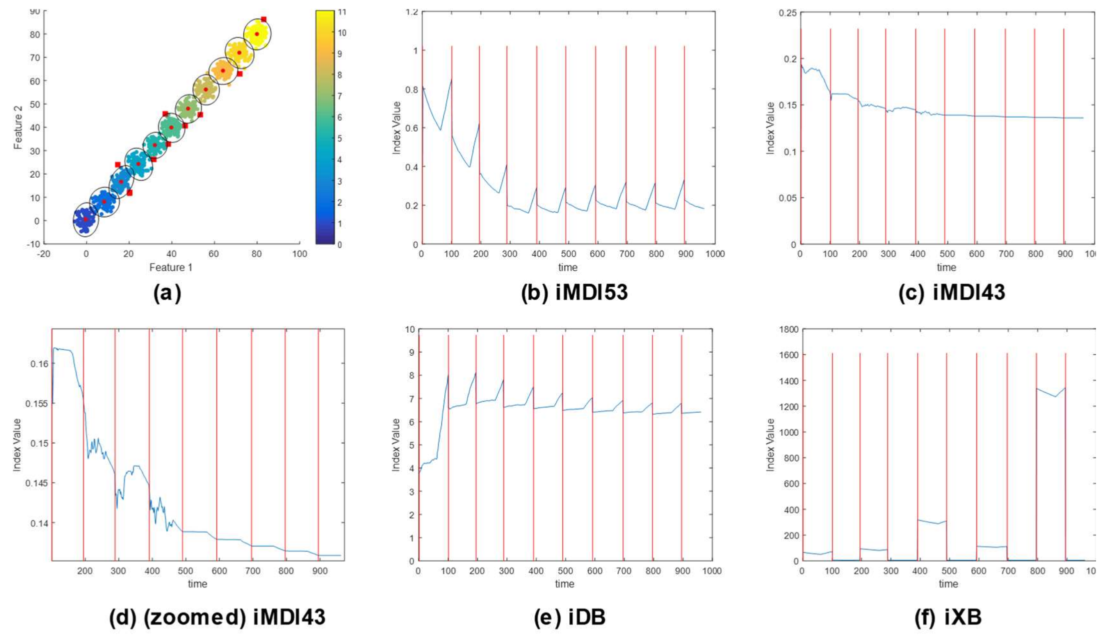


Figure 4-3. Results of the iCVIs on S1 dataset: a) MUSC hardened clustering result, b) iMDI53, c) iMDI43, d) Zoomed version of iMDI43, e) iDB, and f) iXB.

The online clustering result of S1 dataset is shown in **Figure *4-3*** (a) where MUSC detects the static clusters correctly. In all views of MUSC clusters, the red dots are the cluster centers, and the red squares are residual anomalies from MUSC processing (there are 9 hard to see anomalies in **Figure *4-3*** (a). **Figure *4-3*** (b, c, d, e, and f) and **Figure *4-4*** (b, c, d, e, and f) show the values of the four iCVIs on sequentially inputted data from the S1 and S2 datasets based on MUSC processing outputs. The red vertical lines indicate the times when the distribution of samples changes. There is usually a sudden jump in the indices at these times. There is a double vertical red line at the midpoint of the views in **Figure *4-4*** which corresponds to the creation of the cluster in the center of the ring in data set S2. Excluding iMDI43, there is usually a spike before the detection of a new cluster (red vertical lines) which is most pronounced in **Figure *4-3*** and **Figure *4-4***, (b) and (e). After the MUSC algorithm learns the prototype of the new structure, the iCVIs tend to decrease in value, even though only iDB is min-optimal. This is due to the saturation of the iCVI values as more and more points are processed. We know from experiments described in [2] that this effect can be overcome by adding a forgetting factor to the iCVIs. When online processing of many inputs is anticipated, it is important to add this correction to any iCVI. We omitted this while developing the new iMDI43 and iMDI53 indices here, but deem it an essential addition to these new indices for online deployment. The value of iMDI43 does not increase before the detection of a new cluster because its numerator (closest clusters) is not affected when creating a new cluster.

In iDB, and iMDI53, the increase in the iCVI value before the detection of a new cluster starts at the knee point before the vertical line. At the time of the knee location, data from the next cluster (next vertical line) start to arrive. These data are flagged as anomalies by MUSC until they become dense enough to form a new cluster. After a prototype of the anomaly data points is created, the iCVI value decreases sharply at the vertical line.

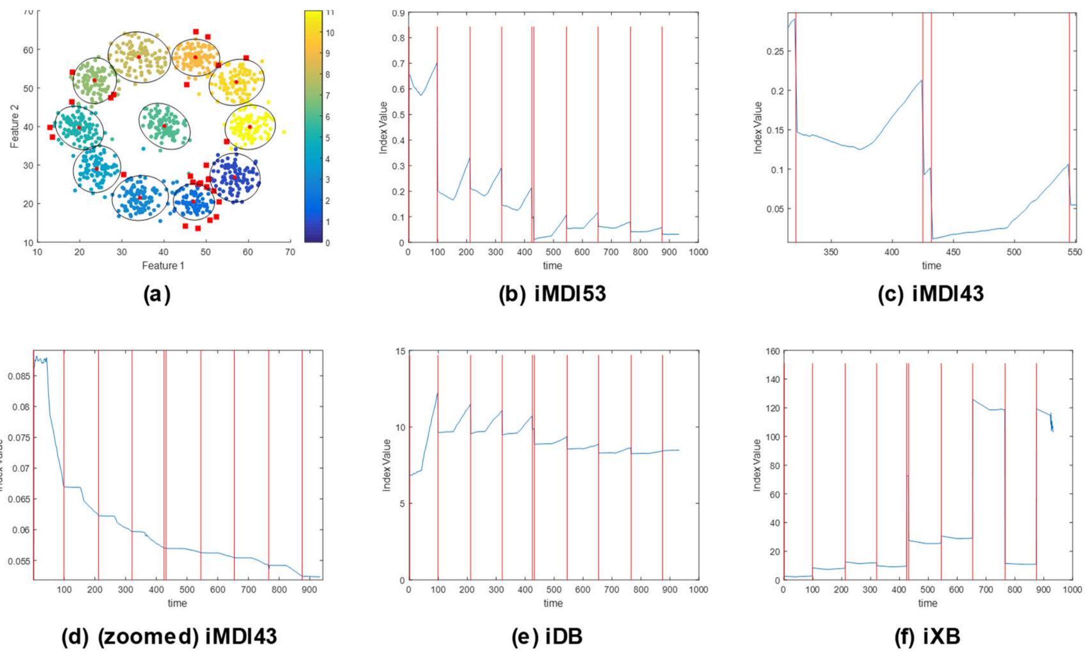


Figure 4-4. Results of the iCVIs on S2 dataset: a) MUSC hardened clustering result, b) iMDI53, c) Exploded view of (b) from t=300 to t=550, d) iMDI43, e) iDB, and f) iXB.

Since the data points from the middle cluster of S2 arrive in a non-sequential manner compared to the other clusters, this structure is detected right after the sixth cluster. The time at which MUSC detects this cluster can be clearly seen in the zoomed view of iMDI53 in **Figure *4-4*** (b) that is shown in **Figure *4-4*** (c). The index iMDI43 fails to show this in **Figure *4-4*** (d). iDB and iXB both detect the time where the middle cluster is formed: see **Figure *4-4*** (e) and **Figure *4-4*** (f).

After investigating the iCVIs performance on synthetic datasets, we turn to the real data sets. The first dataset is the LG weather dataset, which has three distinct event regimes as described earlier. The results of MUSC are shown in **Figure *4-5*** (a) where three possible clusters are detected. The iCVIs values are displayed in **Figure *4-5*** (c-f). All four iCVIs show the time when the third cluster is formed which is the cluster on the left. The third cluster is detected after the middle (small) cluster is formed, which makes it easier for the iCVIs to catch due to the rapid change in the cluster centers and the new cluster size compared to previous cluster. All the four iCVIs in views (c)-(f) indicate two changes in the streaming data because we need to have two clusters to compute the iCVI values. Therefore, in the display, we see only 2 structures, but MUSC detected all the three structures.

The results of MUSC on the Heron datset are displayed in **Figure *4-5*** (b) where four clusters are detected. Only iMDI53 and iDB show the time where the new structures (third and fourth clusters) are detected as in **Figure *4-5*** (g) and **Figure *4-5*** (i) because of the small size and short distance of the third and fourth clusters from others, which makes it harder for iMDI43 and iXB to detect the times when they are formed. This shows that various change detection metrics have different sensitivity to changes in the data stream, which is another clear indication of the need for multiple iCVIs just like the case of traditional CVIs.

Finally, we study the effect of the membership exponent (m) in equation (22) on two of the iCVIs used in this paper with the S1 and S2 datasets. The iXB index is not shown in the interests of brevity, and also because it was, in our experiments, the least reliable of the four indices tested. Please refer to [24] for the effect of the membership exponent on iDB. Different values of m ∈ {1, 1.01, 1.1, 2, 2.5, 3} are used in this experiment. **Figure *4-6*** (a-b) display the values of the two iCVIs for different values of (m) on S1, and **Figure *4-6*** (c-d) show the same thing for S2. From [20] and **Figure *4-6***, it is clear that changes to m do alter the values of all of the iCVIs. The families of graphs obtained with iMDI53 and iDB consistently show the same behavior and change detection capabilities for all values of m. The index iMDI43 is also consistent in the sense that it has the same shapes for S1 and S2 at all values of m, but this index flattens out more rapidly than the other two, and is less able to detect changes in the data stream. In [20], we concluded that m needs to be greater than or equal to 2. However, iMDI53 seems robust against the variations in its membership exponent.

4.10 Conclusions and Discussion

Incremental cluster validity indices provide an unsupervised method for monitoring the performance of online clustering algorithms. This paper presents online incremental versions of two new modified generalized Dunn's indices (iMDI43 and iMDI53) that can be used for dynamic evaluation of evolving cluster structure in streaming data. We used MUSC as our online clustering algorithm, and found that it provides a very reasonable cluster footprint for the four data sets used in our tests. The performance of the two new indices is compared with the incremental Xie-Beni (iXB) and Davies-Boudin (iDB) indices, which to our knowledge offer the only comparable approach, with numerical examples on a variety of synthetic and real data sets.

We investigated how these indices can be used to understand the performance of online clustering algorithms with respect to the appearance of new clusters; and how the clustering algorithm reacts to evolving clusters. We demonstrated that iCVIs could be used to send distress signals about evolving clusters to real time monitors. Our experiments indicate that iMDI53 is more reliable than iMDI43 and has comparable performance to iDB. We concluded that iMDI53 is robust against any value of the membership exponent (m).

From the results of the Heron Island dataset, only iMDI53 and iDB shows the times when the small clusters are detected. This is a clear reason for the need of multiple iCVIs similar

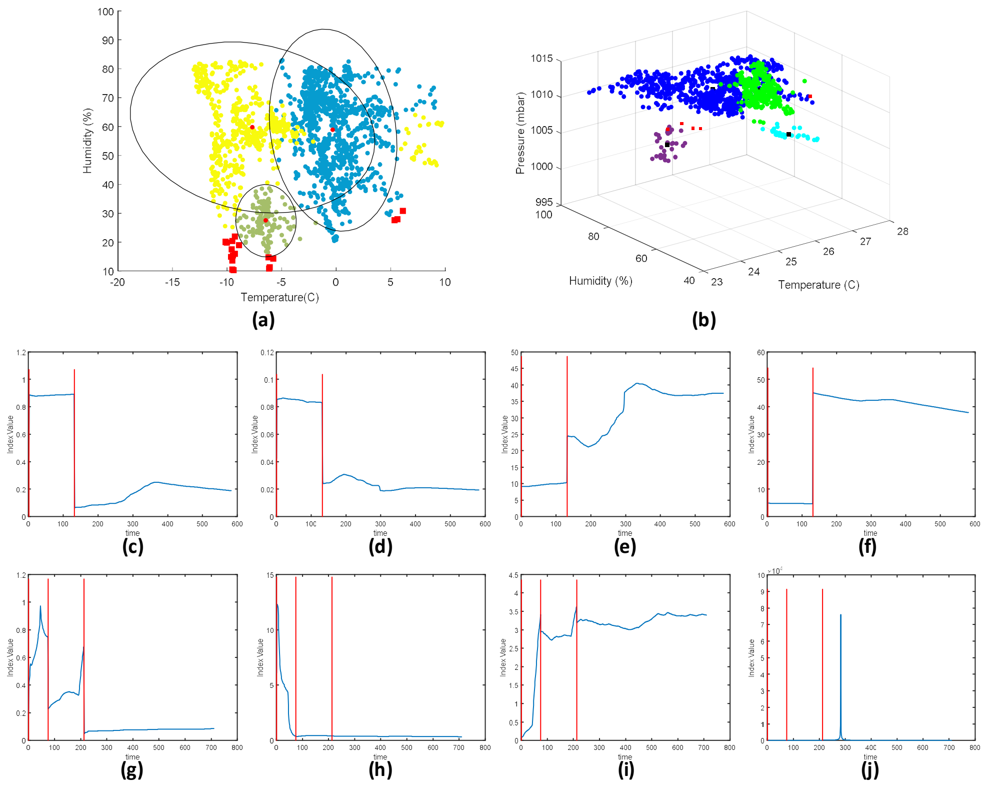


Figure 4-5. Results of the iCVIs on the LG and Heron datasets: a) MUSC hardened clustering result on the LG datset: b) MUSC hardened clustering result on the Heron Island dataset, c) iMDI53 on LG, d) iMDI43 on LG, and e) iDB iXB on LG, (f) iXB on LG, g) iMDI53 on Heron, h) iMDI43 on Heron, i) iDB on Heron, and i) iXB on Heron.

to the case of best use of traditional CVIs. Our next technical step is to derive other incremental validity indices and conduct comparative studies among different types of indices.

Finally, during the course of this work, we have come to believe that much of the terminology associated with classical batch clustering is very confusing and somewhat misleading when used in the streaming data context. Although we have shown scatterplots of batch clusters in **Figure *4-3*** (a), **Figure *4-4*** (a), and **Figure *4-5*** (a-b), these figures were built in hindsight. This type of visualization and the information about clusters it bears would not ordinarily be available in the online streaming environment, since neither the data nor the aggregated partition of it would be retained. Moreover, the classical meaning of cluster validity as described in equations (3) and (4) doesn't make much sense in the streaming case, since no partitions are retained to evaluate. We assert that some new terminology if needed for the streaming data case. Streaming clustering algorithms do not produce clusters in the sense of partitions in $\mathrm{M}_{\mathrm{hcn}}$ as defined in equation (2). Instead, such algorithms produce cluster footprints: in the case of MUSC, the footprint will be a sequence of times of creation, size of "clusters" built, the cluster centers, and the covariance matrices. This is not "clustering", it is much more understandably called something like change detection with a cluster footprint. We will devote a forthcoming paper to the ideas advanced in this paragraph in the near future.

4.11 References

[1] M. Ackerman and S. Dasgupta, “Incremental clustering: The case for extra clusters,” in Neural Information Processing Systems (NIPS), Montreal, Canada, December 2014.

[2] P. Angelov, Evolving Takagi-Sugeno Fuzzy Systems from Streaming Data (eTS+). John Wiley & Sons, Inc., 2010, pp. 21–50.

[3] M. Moshtaghi, J. Bezdek, and C. Leckie, “Online clustering of multivariate time-series,” in Proceedings of SIAMConference on Data Mining, Florida, USA, May 2016.

[4] Bezdek, J. C. (2017). A Primer on Cluster Analysis: Four Basic Methods that (Usually) Work, First Design Publ., Sarasota, FL.

[5] R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, 1973, Wiley Interscience, NY.

[6] S. Theodoridis and K. Koutroumbas, Pattern Recognition, 4th ed., 2009, Academic Press, NY.

[7] A. K. Jain, and R. C. Dubes, Algorithms for Clustering Data, 1988, Prentice Hall, Englewood Cliffs, NJ.

[8] O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.

[9] O. A. Ibrahim, Y. Du, and J. M. Keller, “Extended robust on-line streaming clustering (EROLSC),” in IPMU, 2018.

[10] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters", J. Cybernetics, 3(3), 1973, pp. 32-57.

[11] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," IEEE Trans. SMC, 28(3), pp. 301-315, 1998.

[12] R. Dubes, and A. Jain, "Validity studies in clustering methodologies," Pattern Recognition, 11, pp. 235-254, 1980.

[13] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," Psychometrika, 50(2), pp. 159-179, 1985.

[14] I. Gurrutxaga, J. Muguerza, O. Arbelaitz, J. M. Perez, and J. I. Martin, "Towards a standard methodology to evaluate internal cluster validity indices," Patt. Recog. Letters, 32, 505-515, 2011.

[15] E. Dimitriadou, S. Dolnicar and A. Weingessel, "An examination of indexes for determining the number of clusters in binary data sets," Psychometrika, 67(3), pp. 137-160, 2002.

[16] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Perez and I. Perona, "An extensive comparative study of cluster validity indices," Patt. Recog., 46(1), 2013, pp. 243-256.

[17] L. J. Hubert and P. and Arabie, "Comparing partitions," J. Classification, 2, 1985, pp. 193-208.

[18] V. X. Nguyen, J. Epps and J. Bailey, "Information theoretic measures for clusterings comparisons: variants, properties, normalization and correction for chance," J. Mach. Learning Research, 11, 2010, pp. 2837-2854.

[19] T. C. Havens, J. C. Bezdek, J. M. Keller and M. Popescu, "Dunn's cluster validity index as a contrast measure of VAT images," Proc. ICPR, 2008, pp.1-4.

[20] S. Mahallati, J. C. Bezdek, D. Kumar, M. R. Popovich and T. A. Valiante, "Interpreting Cluster Structure in Waveform Data with Visual Assessment and Dunn's

index", in Frontiers in Computational Intelligence, eds. S. Mostaghim, A. Nuernberger, C. Borgelt, 2017, Springer.

[21] M. Mostaghi, J.C. , Bezdek, S. Erfani, C., Leckie and J. Bailey, "Online Cluster Validity Indices for Streaming Data," Inter. Jo. Intell. Systems, in press.

[22] D. L. Davies and D. W. Bouldin, "A cluster separation measure", IEEE Trans. Patt. Anal. and Mach. Intell., 1(4), 224-227, 1979.

[23] X. L. Xie and G. Beni, “A validity measure for fuzzy clustering,”IEEE Trans. Patt. Anal. and Mach. Intell., 13(8), pp. 841–847, 1991.

[24] O. A. Ibrahim, J. M. Keller, and J. C. Bezdek, “Analysis of Streaming Clustering Using an Incremental Validity Index”, In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1-8, 2018.

[25] O. A. Ibrahim, Y. Wang, J. M. Keller, “Analysis of Incremental Cluster Validity for Big Data Applications,” accepted at the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2018.

[26] Krishnapuram R., and Keller J. M., "A possibilistic approach to clustering," IEEE Trans. Fuzzy Syst., vol. I, no. 2, pp. 98-1 10, May 1993.

[27] Miin-Shen Yang; Chien-Yo Lai, "A Robust Automatic Merging Possibilistic Clustering Method," IEEE Transactions on Fuzzy Systems, vol.l9, no.l, pp.26,41 , Feb. 201 1.

[28] SensorScope. http://lcav.epfl.ch/page-86035-en.html, 2007.

[29] Bezdek, J. C., Rajasegarar, S., Moshtaghi, M., Leckie, C., Palaniswami, M., and Havens, T. C (2011). Anomaly detection in Environmental Monitoring Networks, Computational Intelligence Magazine. 6(2), 52-58

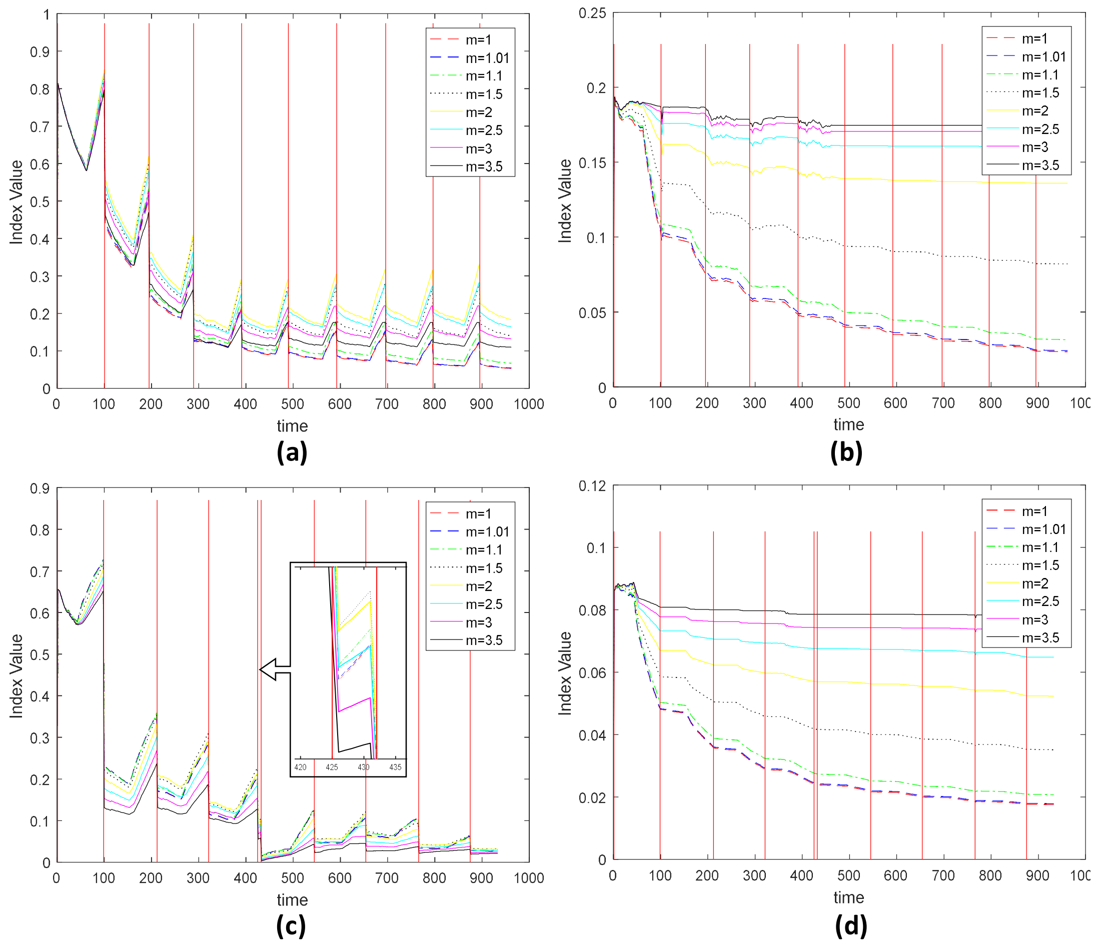


Figure 4-6. Effect of membership exponent for S1and S2 datasets on: a) iMDI53-S1, b) iMDI43-S1, c) iMDI53-S2, d) iMDI43-S2.

Chapter 5: Analysis of incremental cluster validity for big data applications

Omar A. Ibrahim, Yiqing Wang , James M. Keller

Abstract- Online clustering has attracted attention due to the explosion of ubiquitous continuous sensing. Streaming clustering algorithms need to look for new structures and adapt as the data evolves, such that outliers are detected, and that new emerging clusters are automatically formed. The performance of a streaming clustering algorithm needs to be monitored over time to understand the behavior of the streaming data in terms of new emerging clusters and number of outlier data points. Small datasets with 2 or 3 dimensions can be monitored by plotting the clustering results as data evolves. However, as the size and dimensions of streaming data increase, plotting the clustering result becomes unfeasible. Therefore, incremental internal Validity Indices (iCVIs) could be applied for monitoring the performance of an online clustering algorithm. In this paper, we study the internal incremental Davies-Bouldin (iDB) cluster validity index in the context of big streaming data analysis. Also, we study the effect of large number of samples on the values of the iCVI (iDB). Finally, we propose a way to project streaming data into a lower space for cases where the distance measure does not perform as expected in the high dimensional space.

***Keywords*: Big Data, Streaming Clustering, Change Detection, Incremental Cluster Validity.**

5.1 Introduction

Processing large amount of streaming data has gained more attention with the emergence of new applications such as the internet of things (IoT), monitoring systems, and mining content from social media. Data analytics methods need to be online, unsupervised, and adaptive with time (temporal) to be suitable for such applications.

Clustering is an unsupervised technique that searches for structures in the dataset and could be applied to detect anomalies in the dataset [1]. In most cases, all of the dataset should be available in the memory before running clustering algorithms because they require several iterations over the data to compute dissimilarity between data samples. Developing streaming clustering algorithms is a challenging task because the algorithm has to search for new structures as data evolves instead of waiting for the whole data points to apply traditional clustering algorithms.

Online clustering algorithms provide a way to analyze and extract knowledge from streaming data, which makes them desirable in applications such as environmental sensing and packet analysis [1, 2] and can be categorized into two groups [3]. The first category includes algorithms that do not rely on the assumption of any ordering in the data and can be applied to any streaming data. The number of clusters should be defined for such algorithms; an example of this category is sequential k-means (sk-means) [4]. The second group of algorithms assume temporal ordering on the streaming data where consequent data samples are highly related, such as time series [5, 6, 7, 8]. An algorithm of this category does not require number of clusters to be defined in advance. Therefore, a change detection mechanism must be developed to detect new structures in the streaming data. The algorithm studied here, based on possibilistic clustering and Gaussian mixtures, is an example of the second category.

Cluster validity analysis is an important aspect of clustering because it determines the quality of the clustering results. Cluster Validity Indices (CVIs) are computational models and algorithms that look for the best clustering results for a dataset. CVIs can be applied to clustering results of an algorithm by varying specific parameters (number of clusters) and selecting the best results among them. In online clustering, a data point is processed once to update the model and subsequently removed. Thus, monitoring the performance of the algorithm over time is more important in this case. There are several CVIs for static data [9]. However, only four CVIs have been extended for application to streaming data [3, 10]. Separation and compactness of the clusters are normally used in many CVIs [9]. Compactness is usually computed from the density of data points in each cluster, whereas the separation is measured using the distance between the clusters. Therefore, an incremental update needs to be developed for the streaming case because data points are processed only once.

In our previous work, the algorithm and the incremental cluster validity indices (ICVIs) were not tested with big datasets (high dimensions and/or large number of samples) [8, 10, 11]. In this paper, we analyze the performance of MU streaming clustering algorithm (MUSC) and iCVIs on big datasets. Also, we study the effect of large number of samples on the values of the iCVI. Finally, we propose a way to project streaming data into a lower space for cases where the distance measure does not perform as expected in the original high dimensional space (upspace). The next section describes background information and related work. In section 5.3, we describe our online clustering algorithm. Section 5.4 introduces the definition of Davies-Bouldin (DB) CVI. Random Projection formulation is discussed in section 5.5. Section 5.6 shows the datasets used in the evaluation process. The experimental results are presented in section 5.7. A summary and conclusions are given in Section 5.8.

5.2 Background

Clustering streaming data can be divided into two main categories [1]. The first group buffers a window of the streaming data and apply traditional clustering algorithms on that set of the data. After that, the clustering results of adjacent windows are combined to get the final clustering results [12, 13, 14]. The second group applies incremental learning methods and are known as incremental, online or streaming clustering algorithms [7, 15, 16, 17, 18, 19]. Algorithms of the second category process one data point at a time which reduces the complexity and makes them good candidates for big data applications. Online cluster algorithms that assume temporal order in the streaming data have two main parts. The first section is the change detection technique that is used to find new structures in the streaming data. The other part is the adaptation mechanism as new vectors arrive [7].

Cluster Validity Indices (CVIs) can be grouped into two main categories. The first group is the external CVIs. Indices in this group require external information about the data such as assessing the clustering results in comparison with ground truth labels of the data. The second category is internal CVIs (iCVIs) which use the unlabeled data and the clustering results of the algorithm. Comparison between iCVIs is investigated in [9, 20]. Most of iCVI models try to compute separation and cohesion of the clusters.

Internal CVIs (iCVIs) can also be divided into two groups based on the method of measuring the cohesion and separation. Indices in the first category assess the quality of the clustering produced by the algorithm using the partitions only. Partition entropy and partition coefficient are examples of measures in this group [21] where such indices appear in fuzzy clustering validity. The second category has most of the iCVIs where they use both the partition information and the data itself to evaluate the clustering result. In general, the majority of iCVIs assess the quality of each partition using the partition, the unlabeled data, and external parameters produced by the clustering algorithm, such as cluster centers.

In [3], two incremental internal CVIs (iiCVIs) are developed for two well-known iCVIs, the Xie-Beni (XB, [22]) and Davies-Bouldin (DB, [23]). In [11], these indices are generalized and used to study the relationship between the iiCVIs and cluster size as well as the effect of different change detection mechanisms on the index value. It was shown in [11] that incremental DB is more stable than incremental XB. Therefore, incremental DB (iDB) is used in this article as the iiCVI. Here, we test the ability of MUSC and incremental DB on big datasets (high dimensions and large number of samples. Furthermore, if the distance measure fails to perform reasonably in the upspace, we propose a way to project the streaming data to a lower space and perform the clustering in the lower space as data evolves.

5.3 MU Streaming Clustering

The streaming clustering algorithm used this study is based on possibilistic fuzzy clustering with a combination of Gaussian Mixture Models (GMM) is known as MU Streaming Clustering (MUSC) [8, 15]. Possibilistic C-Means (PCM) [24] and the Automatic Merging Possibilistic Clustering Method (AMPCM) [25] are used at the beginning to initialize the cluster system with a window size S of the streaming data. When a new data sample x_{n+1} arrives into the algorithm, the Mahalanobis distance to each cluster at that time is calculated using equation 1. The new data point x_{n+1} will be assigned to the closest cluster if it falls with the predefined threshold. The mean and covariance matrices of the winning cluster are incrementally updated using equation 2 and equation 3.

$$d = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{1}$$

$$\mu_{new} = \mu_{old} + \frac{x_{n+1} - \mu_{old}}{|\mu_{new}|} \tag{2}$$

$$\Sigma_{new} = \frac{(|\mu_{new}| - 1) * \Sigma_{old} + (x_{n+1} - \mu_{old})^T (x_{n+1} - \mu_{old})}{|\mu_{new}|} \tag{3}$$

where $|\mu_{new}|$ is the cardinality of the winning cluster, μ and Σ are mean and covariance of the cluster.

If the minimum distance does not fall within the predefined threshold, it is identified as an outlier and saved in an anomaly list. Data points in the anomaly list may or may not form a new structure. Therefore, the anomaly list is checked in two ways. First, the Mahalanobis distance between data samples in the anomaly list and the current clusters is computed. Data points are incorporated in their closest cluster if they are now within the predefined threshold (as the clusters evolve). Second, MUSC looks for multiple or single emerging structures from the anomaly list. See [8, 15] for more detailed description of the basic algorithm, along with details on initialization and new cluster formation. MUSC uses an incremental update, such that a new arriving data point is used to update the clustering parameters and then removed from memory. Only Cluster representatives such as means, covariance matrices, cluster cardinalities, as well as the outlier data points are kept.

5.4 Davies-Bouldin Index

For streaming data, internal CVIs are the only choice to assess the performance of online clustering algorithm because usually, there is no ground truth associated with the streaming data. The Davies-Bouldin (DB) index [23] is a well-known index which is applicable for both soft and hard partitions. The index is a relative of the Davies-Bouldin (DB) index introduced by Araki et al. [26]. Equations 4 and 5 below show the mathematical representation of DB index.

$$DB(U,V;X) = \frac{1}{k}\sum_{i=1}^{k} \max_{j,j\neq i} \frac{L_i + L_j}{\left\|v_i - v_j\right\|^2} \tag{4}$$

$$L_i = \frac{\sum_{j=1}^{n} u_{i,j}^2 \left\|x_j - v_i\right\|^2}{\sum_{j=1}^{n} u_{i,j}^2} ;1 \leq i \leq kdd \tag{5}$$

where k is number of clusters, v_i is the cluster center of cluster i, and $u_{i,j}$ is the membership of data sample j to cluster i.

The way of computing the within-cluster dispersion is usually a common factor between many indices. The fuzzy within cluster dispersion of cluster i at step n is calculated directly on the input streaming data as:

$$C_{i,n} = \sum_{j=1}^{n} u_{i,j}^2 \left\|x_j - v_{i,n}\right\|^2 \tag{6}$$

An incremental update of the within-cluster dispersion was proposed in [7] for the incremental DB index (iDB). The procedure followed to update the parameters used to compute iDB each time new data point x_{n+1} arrives is shown in figure 5-1 [7]. If DB(n) is the value of the Davies-Bouldin index after the n-th data sample, the incremental update values DB(n+1) when data sample x_{n+1} arrives is shown in equation 7.

$$DB(n+1) = \frac{1}{k}\sum_{i=1}^{k}\max_{j,j\neq i}\frac{L_{i,n+1}+L_{j,n+1}}{\left\|v_{i,n+1}-v_{j,n+1}\right\|^2} \tag{7}$$

where

$$L_{i,n+1} = \frac{C_{i,n+1}}{M_{i,n+1}} \tag{8}$$

$$M_{i,n+1} = M_{i,n} + u_{i,n}^2 \tag{9}$$

In MUSC, when a new data point (x_{n+1}) arrives, its distance to all clusters at the time is calculated using (1). The membership between the i-th cluster and x_{n+1} is computed using (10).

$$u_{i,n+1} = \frac{1}{\sum_{l=1}^{k}\left(\frac{d(x_{n+1},c_i)}{d(x_{n+1},c_l)}\right)^{\frac{2}{m-1}}};1\leq i\leq k;m>1 \tag{10}$$

The procedure to perform this incremental calculation is shown in Figure 5-1,

```
Data: m, c, $V_n$, $\mathbf{U}^{(n+1)}$, $V_{n+1}$, $\forall i: G_{i,1}, M_{i,1}, C_{i,1}$
Input: EMBED Equation.DSMT4 $\forall i: G_{i,n}, M_{i,n}, C_{i,n}$
Output: $\forall i: G_{i,n+1}, M_{i,n+1}, C_{i,n+1}$
For i = 1 to c
        $Q_{i,n+1} = (\mathbf{v}_{i,n} - \mathbf{v}_{i,n+1})^T G_{i,n}$
        $B_{i,n+1} = \|\mathbf{v}_{i,n} - \mathbf{v}_{i,n+1}\|^2$
        $A_{i,n+1} = (u_{i,n+1})^m \|\mathbf{x}_{n+1} - \mathbf{v}_{i,n+1}\|^2$
        $C_{i,n+1} = C_{i,n} + A_{i,n+1} + M_{i,n}B_{i,n+1} + 2Q_{i,n+1}$
        $G_{i,n+1} = G_{i,n} + M_{i,n}(\mathbf{v}_{i,n} - \mathbf{v}_{i,n+1}) + (u_{i,n+1})^m(\mathbf{x}_{n+1} - \mathbf{v}_{i,n+1})$
        $M_{i,n+1} = M_{i,n} + (u_{i,n+1})^m$
Next i
```

Figure 5-1. Incremental compactness for $\mathbf{x}_{n+1}$ [7]

5.5 Random Projection Formulation

Johnson – Lindenstrauss (JL) introduced the idea of random projection [27]. According to the JL lemma, with a predefined probability, a set of n samples $X = \{x_1, x_2, ..., x_n\} \in R^p$ (in the upspace) can be projected to the downspace $Y = \{y_1, y_2, ..., y_n\} \in R^q$ where the pairwise distances from the upspace (X) is nearly preserved in the downspace (Y). It is assumed that $q \ll p$.

There are different random projection variations produced [28] by varying the degree of approximation, the probability selection bounds, and the choice of the projection matrix R, but they are all based on the JL lemma. In this paper, we use the approach proposed by Achlioptas [29] and defined as follow:

Theorem 1. Assume X (n x p) is a set of n data samples in p-dimensional space. Given $\varepsilon, \beta > 0$, let

$$q_0 = \frac{4 + 2\beta}{\varepsilon^2/2 - \varepsilon^3/3} \log n \qquad (11)$$

For integers $q \geq q_0$, let R be a p x q random matrix with $R(i, j) = \{r_{ij}\}$ are independent random variables generated from the following probability distribution:

$$r_{ij} = \begin{cases} +1 & \text{with} \quad P=1/2 \\ -1 & \text{with} \quad P=1/2 \end{cases} \tag{12}$$

The projected data in the downspace is $Y = XR / q \in R^q$. With a probability $p > 1 - n^{-\beta}, \forall i, j \in [1, n]$, the pairwise distances in the downspace (R^q) are within $1 \pm \varepsilon$ from the distances in the upspace (R^p) as follow:

$$(1-\varepsilon)\left\|x_i - x_j\right\|^2 \leq \left\|y_i - y_j\right\|^2 \leq (1+\varepsilon)\left\|x_i - x_j\right\|^2 \tag{13}$$

We have found [30] that good conformance to the approximate isometry given in (13) can be achieved in downspaces well below that specified by the J-L Lemma.

5.6 Datasets

Synthetic and real-life datasets are used in the evaluation process, all of which are presented in a streaming on-line fashion. The datasets in this paper are divided into three categories. The first category has 7 datasets used to evaluate the performance of MUSC and iCVIs (iDB) on high-dimensional (d) datasets [31]. All the first six datasets in this category have 1024 instances generated from 16 Gaussian distributions, while the numbers of dimensions are varied. Dim032 is in 32-dimensional space, dim064 is in 64-dimensional space, dim128 is in 128-dimensional space, dm256 is in 256-dimensional space, dim215 is in 512-dimensional space, and dim1024 is in 1024-dimensional space, as shown in table 5-1. The last dataset in this category is the Synthetic Control Chart Time Series Dataset from [31] which has 600 samples in 60-diminsional space with six different categories: normal,

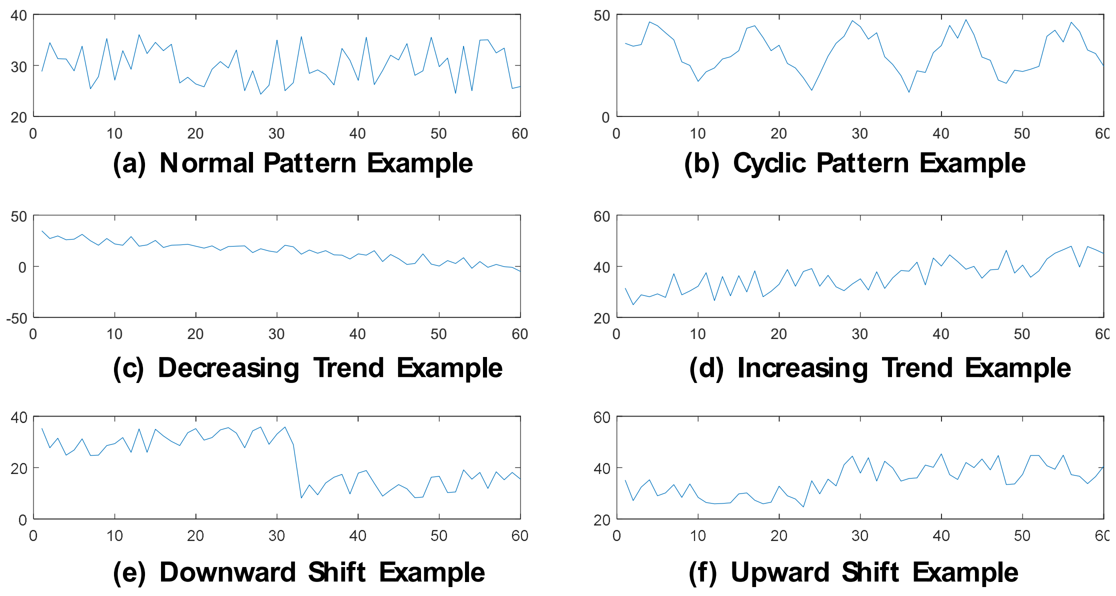


Figure 5-2. Synthetic Control Chart Time Series Dataset where we show a sample from each class [32].

cyclic, increasing trend, decreasing trend, upward shift, and downward shift. Figure 5-2 displays one-time series from each class.

The second group has 7 datasets that are used to test MUSC and ICVIs (iDB) on datasets with large samples (n). The first six datasets in this category have five clusters from Gaussian distributions in 2-diminsional space as displayed in figure 5-3 (a). The number of instances in each cluster is varied in each dataset. The datasets are ordered such that data vectors from cluster 1 arrive before data points from cluster 2 and so on. BD200, BD5000, BD10000, BD20000, and BD50000 have 200, 5000, 10000, 20000, and 50000 instances in each cluster respectively. The last dataset in this group is D31 which has 31 clusters in 2-diminsional space with 100 samples each [32] as seen in figure 5-3 (b).

Table 5-1 Datasets description

Dataset	n	d	c	# labeled subsets
dim032	1024	32	16	16
dim064	1024	64	16	16
dim128	1024	128	16	16
dim256	1024	256	16	16
dim512	1024	512	16	16
dim1024	1024	1024	16	16
BD200	1000	2	5	5
D31	3100	2	31	31
Control Chart Dataset	600	60	6	6
Eldercare	476	3	6	Our Estimate

The third group has a real-life dataset extracted from sensor recordings for a resident in TigerPlace [33]. The dataset has 476 instances in 30-diminsional space which is around 1.5 years of data. Based on the health record of the resident, we believe that the dataset has 6 clusters. Figure 5-3 (c) shows a 3-diminsional projection of the dataset.

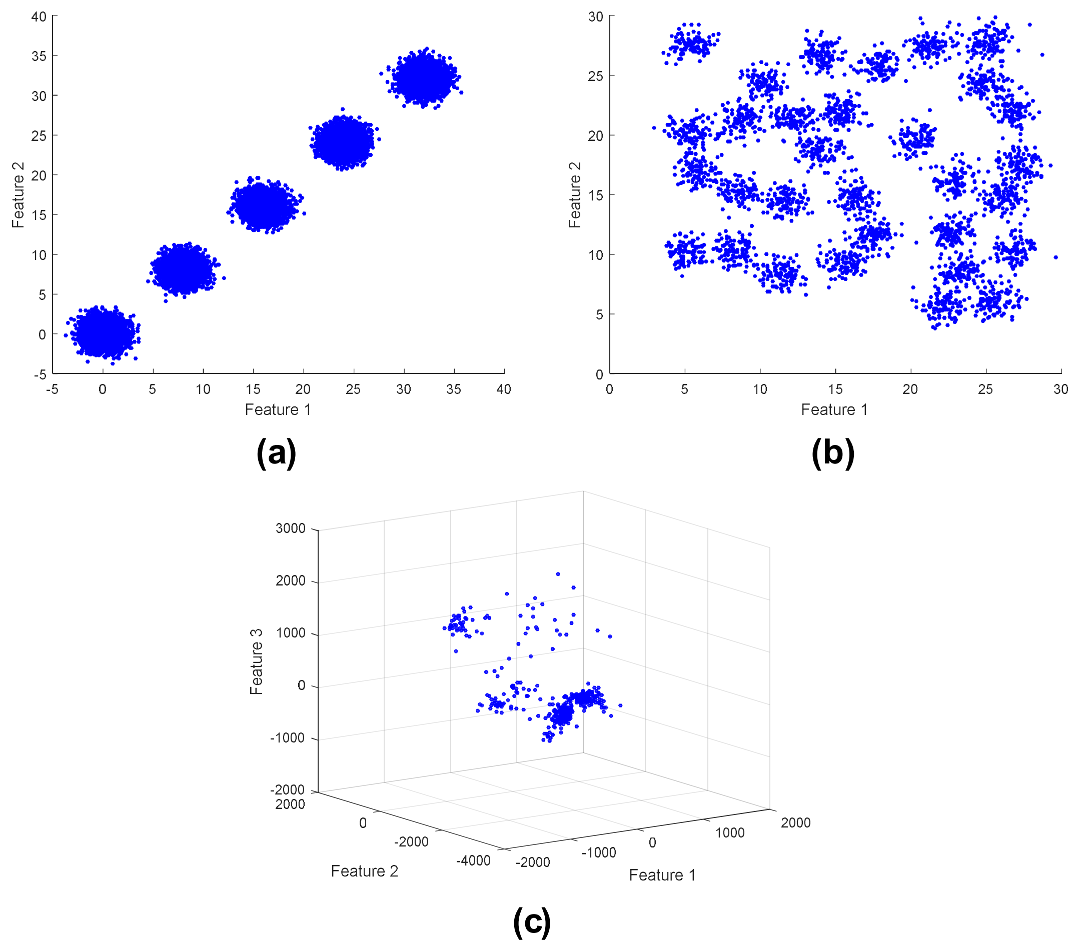


Figure 5-3. Scatter plots of three datasets: a) DB with 5000 samples in each cluster, b) D31 datasets, c) TigerPlace resident data.

5.7 Experimental Results

The experimental results are divided into three categories. The first category is for datasets with high dimension (d). The second group is for datasets with large number of samples (n). The last category is for applying random projections on datasets with high d.

5.7.1 High dimensions (d)

In this section, we test the algorithm performance and the iCVIs sensitivity to detect clusters in high dimensional datasets. High dimensional datasets can be more challenging for streaming clustering due to the issues with the distance measure as dimensionality increases. Also, the sensitivity of the iCVIs can be affected with the increase in the data dimensionality.

The first dataset is the dim032 in table 5-1 which is in the 32-dimensional space. MUSC detected all the clusters correctly with an accuracy of 99%. The iDB result is displayed in figure 5-4 (a). It can be noticed that iDB value changes each time MUSC detects a new structure in the data stream. The remaining datasets are dim064, dim128, dim256, dim512, and dim1024 from table 5-1. The iDB results are displayed in figure 5-4 (b, c, d, e, and f) which shows that iDB correctly determines the time when MUSC detects a new cluster for all datasets. For such high dimensional dataset, iDB can be used to monitor the performance of the online clustering algorithm since data visualization is not possible. The iDB values in figure 5-4 (d, e, f) look similar which indicates that the dataset distribution in the upspace is similar. It is possible to lose some of the properties of the iCVIs as dimensionality (d) increases due to the effect of large d on the distance measure.

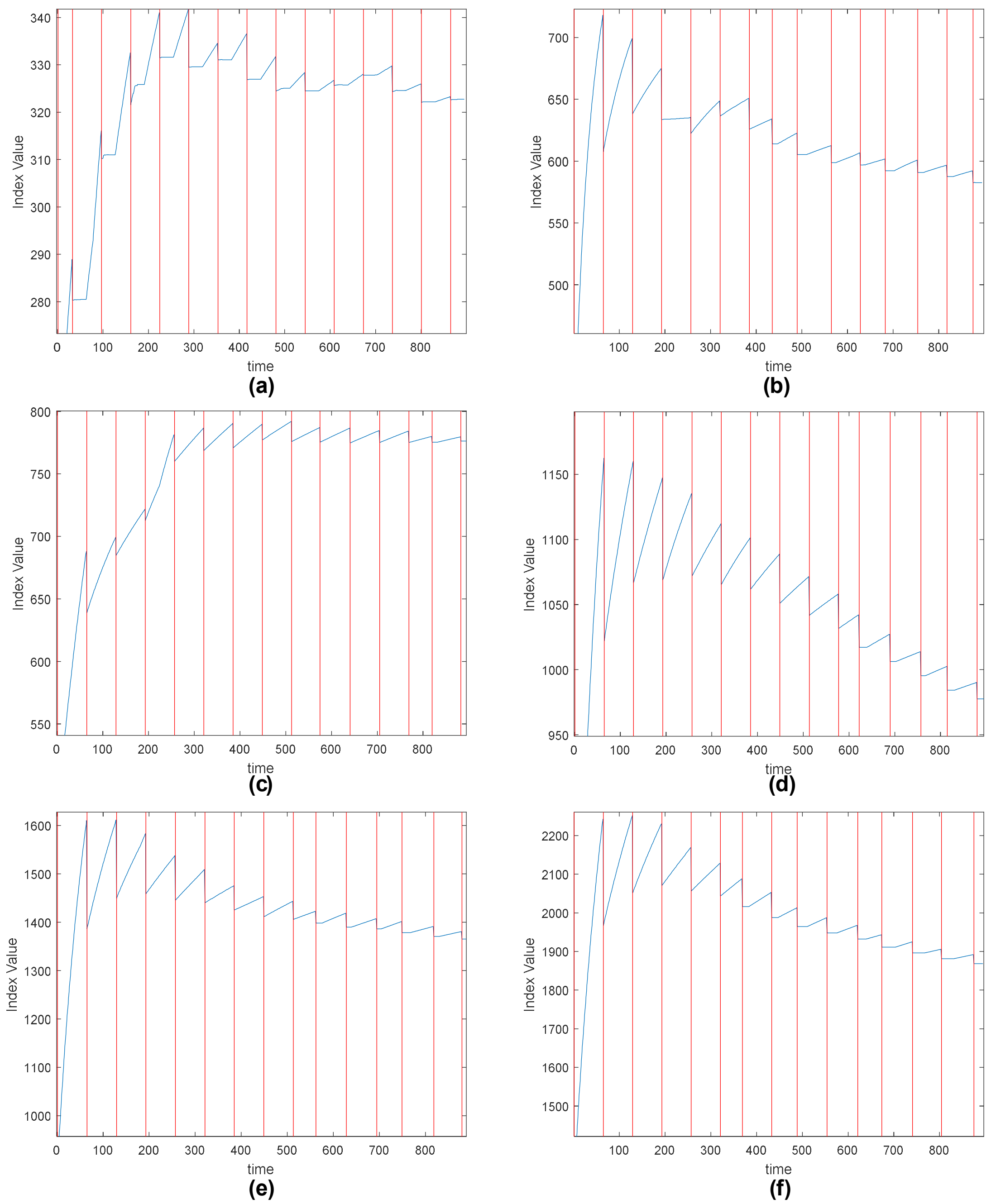


Figure 5-4. Results of the iCVIs on dim datasets: a) iDB on dim032 , b) iDB on dim064, c) iDB dim128, d) iDB dim256, e) iDB on dim 512, and f) iDB on dim1024.

The next experiment examines the ability of MUSC and iCVIs to detect normal trends, shifts, and cyclic behaviors in the data stream from [31]. Each time series in this dataset is in 60-diminsional space. Looking at figure 5-2, we can see the structures of this dataset are quite close to each other, which makes it more difficult for the algorithm to cluster them. MUSC detected all the six clusters with an accuracy of 98%. Figure 5-5 displays the iDB values where we can notice the times when MUSC spawns a new cluster in the data stream.

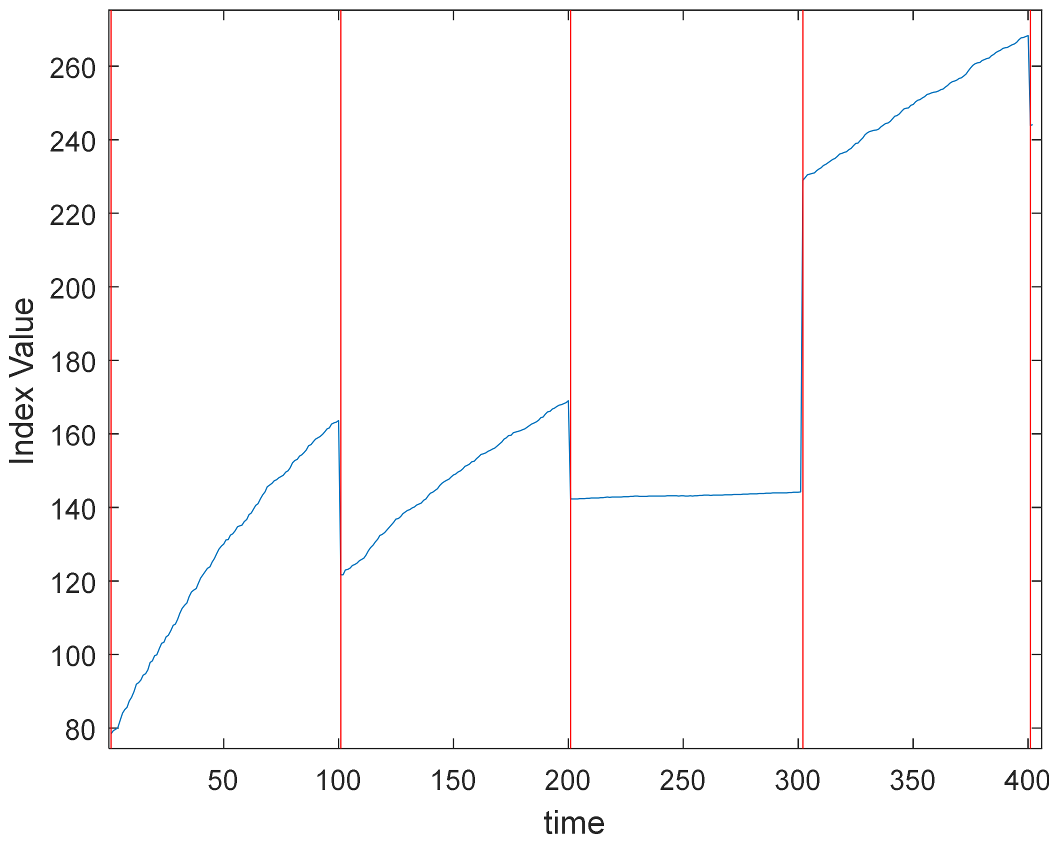


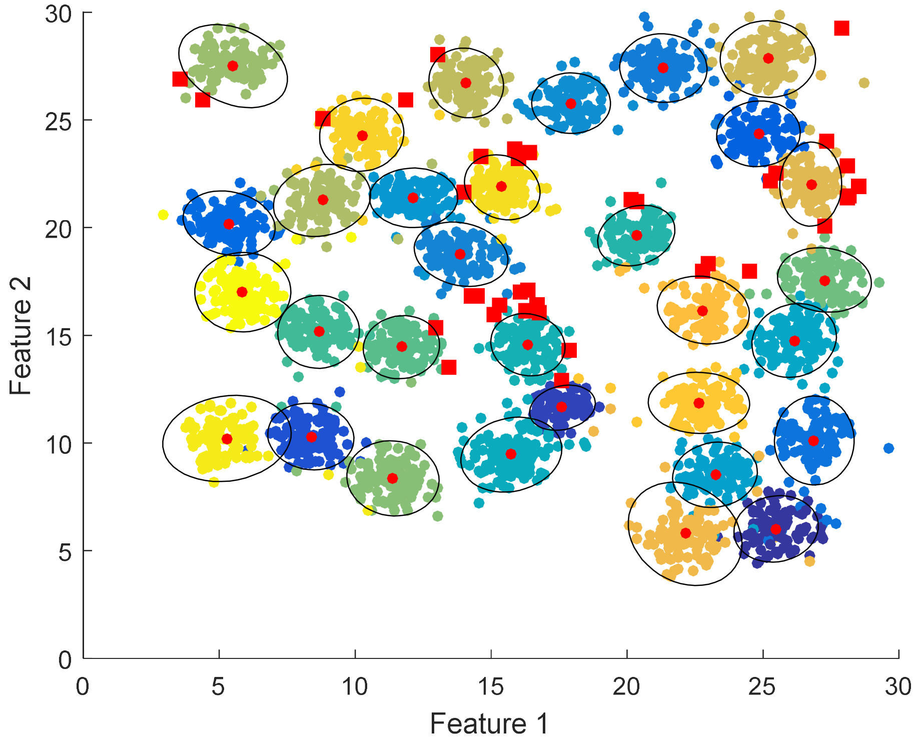
Figure 5-5. Results of the iDB index on Synthetic Control Chart Time Series Dataset.

5.7.2 Large number of samples (n)

In this experiment, we consider the effects of large number of samples (n). The first dataset used in this experiment is D31. This dataset has 31 clusters in 2-dimensional space. From figure 5-3 (b), we can notice that the clusters are close to each other which increases the chance for a clustering algorithm to merge relatively close clusters. MUSC clustering results are displayed in figure 5-6 (a) where it detected all the clusters with an accuracy of

96%. The iDB values are displayed in figure 5-3 (b) where it shows the times when MUSC (correctly) detects new structure in the streaming data.

The second part is to use the BD dataset in table 5-1 where we vary number of samples in each cluster to test the performance of MUSC and iDB as number of samples increases. In the first case, we set number of samples in each cluster to 200. MUSC clustering results and iDB values are displayed in figure 5-3 (a-b). After that, the number of samples in each cluster is increased to 5000, 10000, 20000, 50000, and 250000. The clustering results and iDB values are shown in figure 5-3 where MUSC clusters the data correctly in all cases. iDB values show the times when MUSC creates a new structure in all cases, and the value of the iCVI behaves consistently. Even though, these datasets can be visualized, monitoring such larger number of streaming data is not possible. Therefore, iDB can be used to monitor such type of datasets. Also, all the iDB values look almost identical which indicates that number of data points has no influence on the iCVI value.



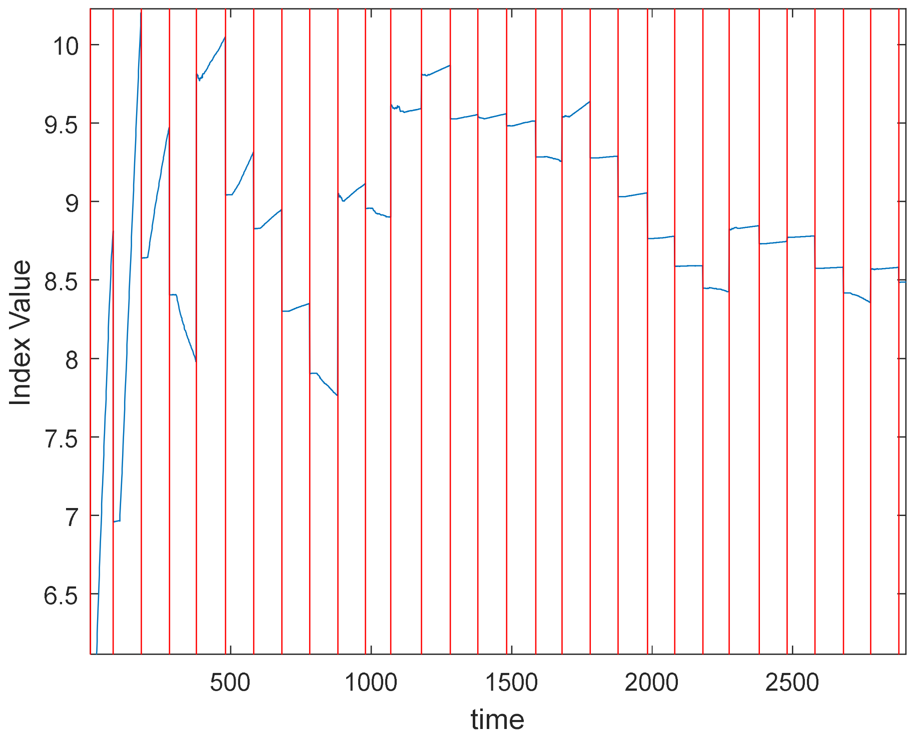


Figure 5-6. Results of the iCVIs on D31 dataset: a) MUSC hardened clustering result, b) results of the iDB index.

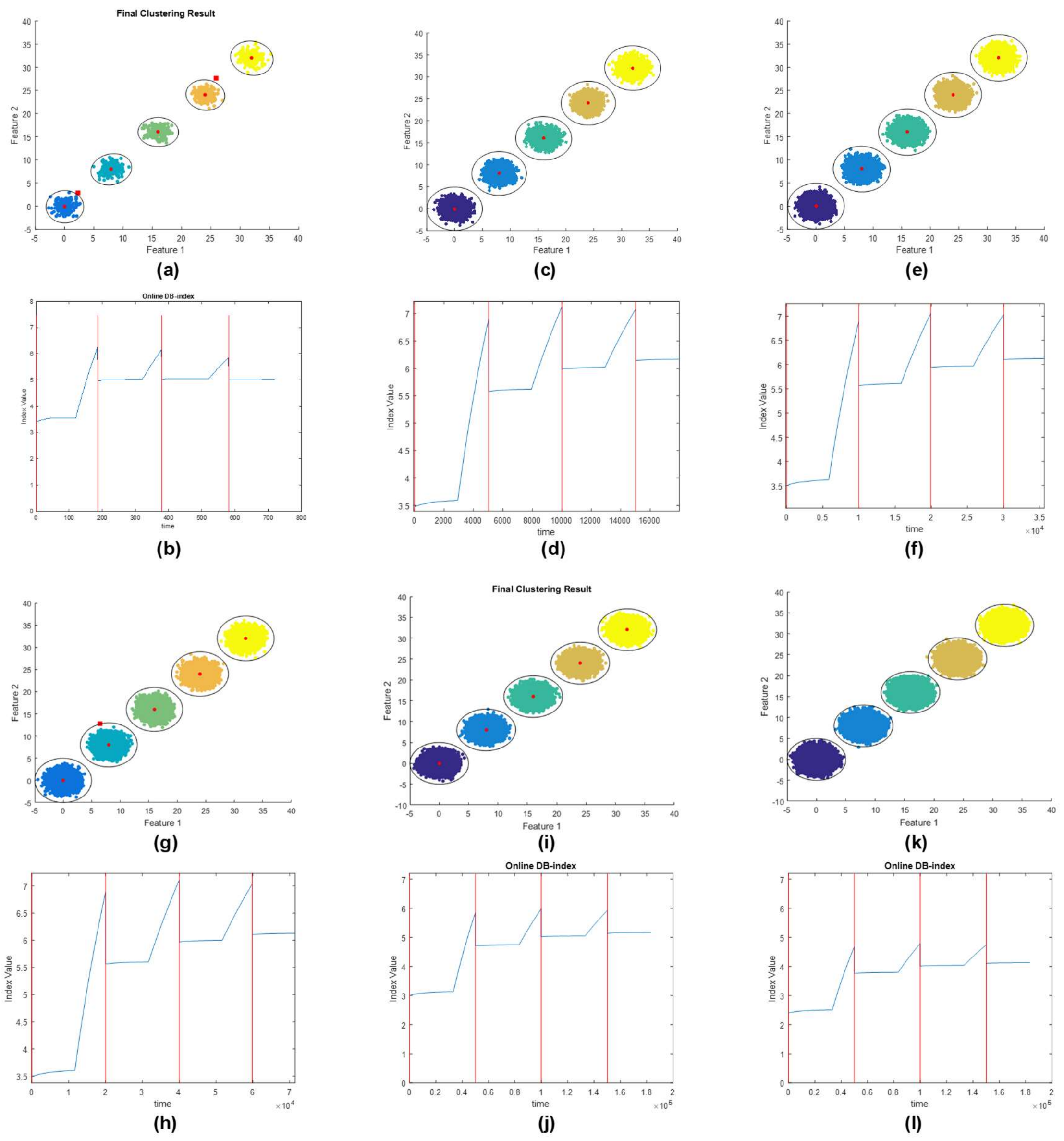


Figure 5-7. Results of the iCVIs on BD datasets: a) MUSC result on BD 200, b) iDB on BD200, c) MUSC result on BD5000, d) iDB on BD5000, e) MUSC result on BD10000, f) iDB on BD10000, g) MUSC result on BD20000, h) iDB on BD20000, i) MUSC result on BD50000, j) iDB on BD50000, k) MUSC result on BD250000, l) iDB on BD250000.

5.7.3 Random projection of streaming data

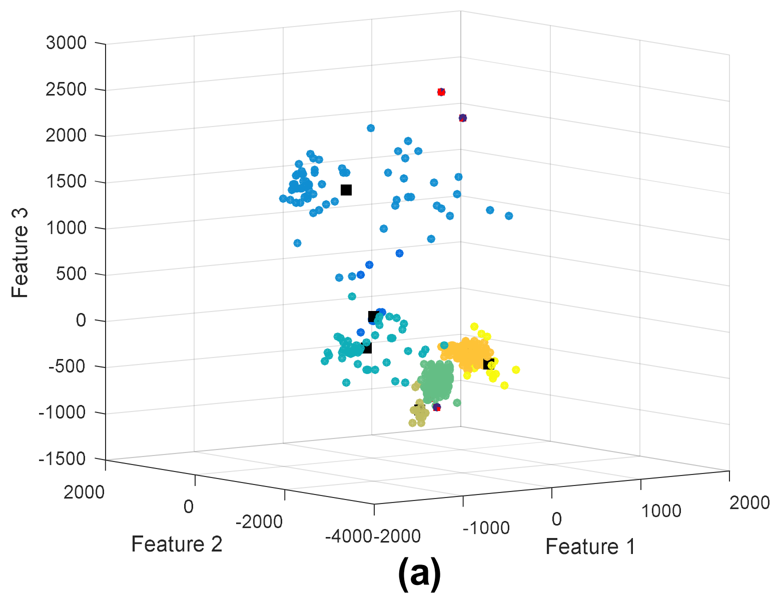
The Eldercare dataset in table 5-1 is used for this experiment. The dataset has 476 samples in 30-diminsional space. The features of this data are of different types. For example, the bathroom visit feature counts the number of bathroom visits per day (ranges from 2 to 20) while total bedroom activity counts the total bedroom motion sensor hits (ranges from 100-1000s). Having features of different types makes our distance measure (Mahalanobis distance) not to perform as required. As a result, the clustering results in the upspace did not match the health record history of the resident. Therefore, we need to either use a different distance measure or project the data to a lower space and compare with health record to make sure it matches.

We selected the second approach to deal with high dimensional datasets by projecting it to a lower space to avoid the issues with distance measures behaving strandely in the upspace. Methods such as PCA or random projection can be used to project the data to a lower space. The challenge with the data stream is that the whole dataset is not available in the memory at any point of the time. Therefore, PCA is not a viable approach. MUSC uses a window size of S (sxd) data points as an initialization process. Random projection is used as our dimensionality reduction technique. We use the initialization window S of MUSC to build the projection matrix (R) to project the data from d-space (sxd) to 3-space (sx3). Once the projection matrix is generated, we use it to project the remaining data stream using the following equation:

$x'_{n+1} = x_{n+1} * R$, where $x_{n+1} \in R^d$, $x'_{n+1} \in R^3$ and $R \in R^{dx3}$

We select a projection matrix, from a selection, that best preserves the pairwise distance in the downspace compared to the upspace. We generated multiple projection matrices and selected the one that best preserves the pairwise distances on the initial window. Figure 5-

8 (a) shows the clustering results where MUSC detected all the possible clusters in the dataset. The iDB values are plotted in figure 5-8 (b) and it shows the times where a new structure is detected in the streaming data.



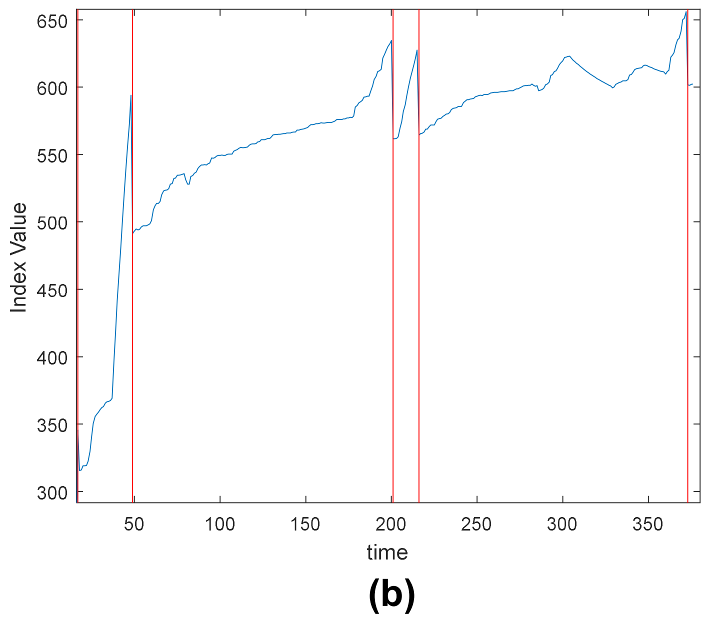


Figure 5-8. Results of the iCVIs on the eldercare dataset: a) MUSC hardened clustering result, b) results of the iDB index.

5.8 Conclusions

In this paper, we examined the ability of incremental internal cluster validity measures to monitor the process of a streaming clustering algorithm to correctly identify new structures, and to properly handle outliers, in data under situations with large dimensionality and large number of samples. The extremely good performance of both the MUSC algorithm and the iiCVI to correctly monitor conditions in controlled environments give us confidence to utilize them in truly unsupervised Big Data streaming data scenarios. Random projections are shown to provide an additional tool to ameliorate the problems caused by distance calculations in very high dimensional spaces. Clearly more work needs to be done to exploit this capability.

5.9 References

[1] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. ao Gama, “Data stream clustering: A survey,” in ACM Computing Surveys, vol. 46, no. 1, pp.13–31, July 2013.

[2] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O’Callaghan, “Clustering data streams: Theory and practice,” IEEE Transaction on Knowledge and Data Engineering, vol. 15, no. 3, pp. 515–528, 2003.

[3] M. Moshtaghi, J. Bezdek, S. Erfani, C. Leckie, and J. Bailey, “Online Cluster Validity Indices for Streaming Data,” arXiv preprint arXiv:1801.02937, 2018.

[4] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1, no. 14, pp. 281-297. 1967.

[5] M. Ackerman and S. Dasgupta, “Incremental clustering: The case for extra clusters,” in Neural Information Processing Systems (NIPS), Montreal, Canada, December 2014.

[6] P. Angelov, Evolving Takagi-Sugeno Fuzzy Systems from Streaming Data (eTS+). John Wiley & Sons, Inc., 2010, pp. 21–50.

[7] M. Moshtaghi, J. Bezdek, and C. Leckie, “Online clustering of multivariate time-series,” in Proceedings of SIAMConference on Data Mining, Florida, USA, May 2016.

[8] O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.

[9] O. Arbelaitz, I. Gurrutxaga, J.Muguerza, J.M. Perez, and I. Perona, “An extensive comparative study of cluster validity indices,” Pattern Recognition, vol. 46, no. 1, pp. 243 – 256, 2013.

[10] O. Ibrahim, J. Keller, J. Bezdek “Evaluating Evolving Structure in Streaming Data with Modified Dunn's Indices,” submitted to IEEE Transaction on Evolving Topics in Computational Intelligence, 2018.

[11] O. A. Ibrahim, J. M. Keller, and J. C. Bezdek, “Analysis of Streaming Clustering Using an Incremental Validity Index,” in IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2018.

[12] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, On Clustering Massive Data Streams: A Summarization Paradigm. Springer US, 2007, vol. 31, pp. 9–38.

[13] N. Ailon, R. Jaiswal, and C. Monteleoni, “Streaming k-means approximation,” in Neural Information Processing Systems 2009, vol. 22, 2009, pp. 10–18.

[14] M. Salehi, C. Leckie, M. Moshtaghi, and T. Vaithianathan, “A relevance weighted ensemble model for anomaly detection in switching data streams,” in Proceedings of PAKDD, 2014, pp. 461–473.

[15] O. Ibrahim, Y. Du, J. Keller, “Extended robust on-line streaming clustering (EROLSC),” in IPMU 2018.

[16] P. Angelov and X. Zhou, “Evolving fuzzy-rule-based classifiers from data streams,” IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pp. 1462–1475, 2008.

[17] F. Cao, M. Ester, W. Qian, and A. Zhou, “Density-based clustering over an evolving data stream with noise,” in SIAM Conf. on Data Mining, 2006, pp. 328–339.

[18] P. Kranen, I. Assent, C. Baldauf, and T. Seid, “The clustree: Indexing micro-clusters for anytime stream mining,” Knowledge and Information Systems, vol. 29, no. 2, pp. 249–272, 2011.

[19] N. Mozafari, S. Hashemi, and A. Hamzeh, “A statistical approach for clustering in streaming data,” Artificial Intelligence Research, vol. 3, pp. 38–45, 2014.

[20] G. W. Milligan and M. C. Cooper, “An examination of procedures for determining the number of clusters in a data set,” Psychome- trika, vol. 50, no. 2, pp. 159–179, 1985.

[21] J. C. Bezdek. A Primer on Cluster Analysis: Four Basic Methods that (Usually) Work, First Edition Design Publ., Sarasota, FL, 2017.

[22] X. L. Xie and G. A. Beni, "A validity measure for fuzzy clustering," IEEE Trans. PAMI, vol.13, no.8, pp. 841-846.

[23] D. L. Davies and D. W. Bouldin, “A cluster separation measure,” IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 1, no. 2, pp. 224–227, Feb. 1979.

[24] R. Krishnapuram, J. Keller, "A possibilistic approach to clustering," IEEE Trans. Fuzzy Syst., vol. I, no. 2, pp. 98-1 10, May 1993.

[25] Miin-Shen Yang; Chien-Yo Lai, "A Robust Automatic Merging Possibilistic Clustering Method," Fuzzy Systems, IEEE Transactions on, vol.l9, no.l, pp.26,41 , Feb. 2011.

[26] S. Araki, H. Nomura, and N. Wakami, “Segmentation of thermal images using the fuzzy c-means algorithm,” in Proceedings of the Second IEEE International Conference on Fuzzy Systems, San Francisco, USA, April 1993, pp. 719–724.

[27] W. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," Contemporary Mathematics, vol. 26, 1984, pp. 189-206.

[28] J. Matousek, “On variants of the Johnson-Lindenstrauss lemma," Random Structures & Algorithms Volume 33, Issue 2 (September 2008).

[29] Achlioptas D. Database-friendly random projections. In Proc. ACM Symp. on the Principles of Database Systems, pages 274–281, 2001.

[30] J. Bezdek, X. Ye, J. Keller, M. Popescu, A. Zare, “Random Projection below the JL Limit”, Proceedings of the IEEE International Joint Conference on Neural Networks, Vancouver, Canada, July 2016, pp. 2414-2423.

[31] Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[32] P. Fränti and S. Sieranoja Applied Intelligence, 1-17, 2018, https://doi.org/10.1007/s10489-018-1238-7.

[33] MJ Rantz, et al., "TigerPlace: A new future for older adults," Journal of Nursing Care Quality, vol 20, no. 1, 2005, pp. 1-4.

Chapter 6: A New Incremental Cluster Validity Index for Streaming Clustering Analysis

Omar A. Ibrahim[1,2]
[1]Dept of Electrical Engineering and Computer Science
Columbia, MO, USA
oai9bc@mail.missouri.edu
[2]EE Dept, Tikrit University, Tikrit, Iraq

James M. Keller
Dept of Electrical Engineering and Computer Science
Columbia, MO, USA
kellerj@missouri.edu

Mihail Popescu
Health Management and Informatics Department
Columbia, MO, USA
popescum@health.missouri.edu

***Abstract* — In this paper, we present an incremental version of the *Partition Coefficient and Exponential Separation* (PCAES) cluster validity index in the context of streaming data analysis. Incremental PCAES (iPCAES) can be used to monitor evolving structures in streaming data. We investigate the use of the proposed index to understand and analyze the performance of the *MU Streaming Clustering* (MUSC) algorithm. Synthetic and real-life streaming datasets are used to demonstrate the benefits that can be drawn from such indices such as the appearance of a new structure in the data stream, handling of outlier data samples, and the effect of the streaming sample order on the resultant cluster history. We compare the performance of iPCAES index with the incremental *Davies-Boudi*n index (iDB) because iDB was found to be the most stable among other incremental indices that offer comparable approaches.**

Keywords—Online clustering, change detection, incremental internal cluster validity, incremental Partition Coefficient and Exponential Separation index, incremental Davies-Bouldin index

6.1 Introduction

Developments in information technology have allowed large flows of data across IP networks. Some of these flows are time critical so that it is impossible to wait for the data to be available as a single file. These kinds of online data are referred to as data streams or streaming data. Moreover, even if the data can be stored locally, the size of the incoming samples could be large which makes it impractical to process individual record more than once. It is even more difficult from an algorithmic and computational point of view when data patterns evolve with time. Hence, it is crucial in the design the mining algorithms to take into account changes in underlying structure of the data stream [1]. Online clustering algorithms provide a way to study and mine streaming data. One type of online clustering is sequential clustering where data samples arrive one at a time, and the clustering representatives (means and covariance matrices) are updated with each sample [2, 3, 4].

One important problem in clustering is to assess the clustering output of a clustering algorithm. *Cluster validity Indices* (CVIs) are a scalar measure used to identify the best clustering results U (partition matrix) or C (number of cluster). For traditional CVIs [5], the clustering results (partitions) must be obtained from static datasets. Recently, many streaming clustering algorithms have been proposed in the literature to detect anomalies, drift, and evolving cluster structure in streaming data [6]. However, there has been very little work in developing cluster validity indices for streaming clustering algorithms. In [7], incremental versions of the *Davies-Bouldin* (DB) [8] and *Xie-Beni* (XB) [9] validity indices are developed. These incremental CVIs (iCVIs) can be used to with streaming clustering algorithms to monitor the algorithm performance and cluster formation.

In this article, we develop an incremental formula for the *Partition Coefficient and Exponential Separation* (PCAES) [10] to be used with streaming clustering algorithms. We investigate the relationship between the iCVI output and the partitions produced by our *MU Streaming Clustering algorithm* (MUSC) [11, 12]. We compare the performance of the proposed iCVI with the incremental DB (iDB) index because it is the most stable among other iCVIs [7, 13]. This paper has the following contributions: (1) we develop an incremental version of PCAES index (iPCAES); (2) we show the relationship between the iCVIs and MUSC in terms of: when a new cluster is formed (sudden drop in the iCVI value); and how to use the knee points in the graph of the iCVI to conclude information about the data stream with regard to clusters size, and streaming order of the data samples, (3) we show the robustness and stability of iPCAES compared to iDB. This is particularly important for large volume streaming datasets in high dimensional spaces.

The paper is organized as follow. In section II, a summarization of related work is given. MUSC online clustering algorithm used in this paper is discussed in section III. In section IV, background information and the definition of the PCAES index is provided. In Section V, derivation of an incremental form of cluster compactness is discussed. Incremental PCAES (iPCAES) is introduced in section VI. The datasets used in the evaluation is introduced in section VII. Section VIII shows initialization of the parameters. Experimental results and discussion are shown in section IX. Our conclusions are given in section X.

6.2 Background

In this section, we briefly describe related work in cluster validity analysis. We also, review related work in iCVIs. To validate the results of a clustering algorithm, many static cluster validity indices have been proposed in the literature [6, 8-9]. Internal CVIs are divided into two categories based on the measure of cohesion and separation. The first category uses only the partitions generated by the algorithm to assess the clustering quality such as the partition coefficient and partition entropy [6]. Most internal CVIs belong to the second category which assess the quality of each partition using the unlabeled data, the partition, and any secondary parameters produced by the clustering, such as cluster centers. In [14], a comparison was made between 30 different internal CVIs using four hierarchical clustering methods. A survey of 15 internal CVIs was presented in [15]. Another comparison of 30 internal CVIs for crisp clustering was investigated in [5]. *Partition Coefficient and Exponential Separation* (PCAES, [10]) index, *Dunn's* index (DI, [16]), *Davies-Bouldin* (DB) [8] and *Xie-Beni* (XB) [9] are internal CVIs designed to select a best partition amongst a set of *candidate partitions*.

With the rich literature in static cluster validity analysis, there has been little work in developing cluster validity indices from streaming clustering algorithms. In [7], incremental versions of two internal CVIs, the Xie-Beni (XB) and Davies-Bouldin (DB), were proposed by deriving an incremental formula for the cohesion term. In [13], the authors extended the incremental DB (iDB) index to a more general version and illustrated how the iDB can be used to analyze and understand the performance of a streaming clustering algorithm. The use of iCVIs with big data was introduced in [17] which show the advantage of iCVIs for high dimensional streaming data. Recently, an incremental version of two modified Dunn's indices was developed in [18].

6.3 MU Streaming Clustering Algorithm

In this section, we briefly describe the MUSC online clustering algorithm developed in [11] and extended in [12]. MUSC combines *Gaussian Mixture Models* (GMM) and possibilistic fuzzy clustering. *Possibilistic C-Means* (PCM) [19] and the *Automatic Merging Possibilistic Clustering Method* (AMPCM) [20] are used to initialize the GMMs in a window *S* of the streaming data. The reason behind using PCM is to detect anomalies in the initialization window (*n*-data points), *S*, and for spawning a new cluster due to its tendency to look for dense regions in a dataset. After removing outlier data samples in *S* (data points with low typicalities to all existing clusters), the remaining data samples are clustered using AMPCM, and the clustering result is used to initialize a GMM.

When a new data point x_{n+1} arrives at time n+1, its Mahalanobis distance is computed to the existing prototypes using equation (1). Then, x_{n+1} is assigned to the closest Gaussian cluster if the distance is within a pre-specified threshold (T). The mean and covariance of the winning Gaussian are incrementally updated using (2) and (3). After updating the Gaussian parameters, x_{n+1} is discarded.

$$d = \sqrt{(x-v)^T \Sigma^{-1} (x-v)} \tag{1}$$

$$v_{new} = v_{old} + \left(\left(x_{n+1} - v_{old} \right) / \left| v_{new} \right| \right) \tag{2}$$

$$\Sigma_{new} = \frac{(|v_{new}| - 1) * \Sigma_{old} + (x_{n+1} - v_{old})^T (x_{n+1} - v_{old})}{|v_{new}|} \tag{3}$$

where $|v_{new}|$ is the new cardinality of the winning cluster, v and Σ are mean and covariance matrix of the cluster.

If the distance to the closest prototype exceeds the threshold (T), the new data point x_{n+1} is saved in the anomaly list. The points in the anomaly history may or may not indicate the emergence of a new cluster. We follow two ways to monitor changes in the anomaly list. First, we compute the Mahalanobis distance between anomaly data points and existing prototypes to check if any of the clusters has grown to contain what was an anomaly data point before. Points are incorporated in their closest Gaussians if their distance are within a pre-specified threshold (T). Second, we follow the same procedure to initialize the GMM at the beginning by clustering the outliers to look for one or multiple emerging structures. For detailed description about MUSC and details on initialization, and threshold values, one can refer to [11, 12]. Means, covariance matrices, cluster cardinalities, and outlier data points are used to represent the historical footprint of the evolving structure.

6.4 Partition Coefficient and Exponential Separation Index

PCAES index was proposed in [10] for fuzzy clustering. Suppose, we have a dataset $X = \{x_1, x_2, ..., x_n\} \subset \Re^p$. Let $U = \{U^{(1)}, U^{(2)}, ..., U^{(c)}\}$ be a fuzzy c-partition from a fuzzy clustering algorithm such as FCM. Each cluster is validated based on two factors: a normalized partition coefficient and an exponential separation measure. These two terms are combined together to get the index value called a partition coefficient and exponential separation (PCAES) index. PCAES for the i-th cluster is computed as follows:

$$\mathrm{PCAES}_i = \sum_{j=1}^{n} \frac{u_{ij}^2}{\mu_M} - \exp\left(-\frac{\min_{k \neq i}\{\|v_i - v_k\|\}}{\beta_T} \right) \tag{4}$$

$$\mu_M = \min_{1 \leq i \leq c} \left\{ \sum_{j=1}^{n} u_{ij}^2 \right\} \tag{5}$$

$$\beta_T = \frac{\sum_{l=1}^{c} \left\| v_l - \bar{v} \right\|^2}{c} \tag{6}$$

where v_i is the cluster center of cluster i and $\bar{v}$ is average of all c-clusters.

A normalized portion coefficient (equation 7) is used to measure the compactness of cluster i relative to the most compact cluster (equation 5). An exponential-type separation measure (equation 8) computes the separation between cluster-i and its closest cluster relative to β_T (equation 6) which is the total average distance measure for all clusters.

$$\sum_{j=1}^{n} \frac{u_{ij}^2}{\mu_M} \tag{7}$$

$$\exp\left(-\frac{\min_{k\neq i}\{\|v_i - v_k\|\}}{\beta_T}\right) \qquad (8)$$

The compactness and separation values for each cluster are restricted between 0 and 1 as shown in equations (9) and (10). Therefore, PCAES_i is limited between 1 and -1 as in equation (11).

$$0 < \sum_{j=1}^{n} \frac{u_{ij}^2}{\mu_M} \leq 1 \qquad (9)$$

$$0 < \exp\left(-\frac{\min_{k\neq i}\{\|v_i - v_k\|\}}{\beta_T}\right) \leq 1 \qquad (10)$$

$$-1 < \mathrm{PCAES}_i < 1 \qquad (11)$$

From the cluster compactness and separation formulas, larger PCAES_i index value indicates that cluster i is compact and separated from other clusters. On the other hand, small or negative value means that the cluster is not well compact or separated. The total PCAES index for all clusters is found by adding PCAES_i values for all c-clusters as in equation (12). Larger PCAES(c) value indicates better clustering results (compact and separated clusters).

$$\begin{aligned}\mathrm{PCAES(c)} &= \sum_{i=1}^{c} \mathrm{PCAES}_i \\ &= \sum_{i=1}^{c}\sum_{j=1}^{n} \frac{u_{ij}^2}{\mu_m} \\ &\quad - \sum_{i=1}^{c} \exp\left(-\frac{\min_{k\neq i}\{\|v_i - v_k\|\}}{\beta_T}\right)\end{aligned} \qquad (12)$$

6.5 Incremental Form of Cluster Compactness

Consider we have streaming vector data inputs, for example $\mathbf{x}_k \in \Re^p, k = 1,2,\ldots,n$, where k=1 is at time 1. Euclidean distance is chosen as our distance measure in the input space.

Assume that a streaming clustering algorithm assigns crisp or fuzzy membership labels to $\mathbf{x}_k$ in *c* clusters, $\mathbf{U}^{(k)} = \{u_{ik} : 1 \le i \le c\}$. After n inputs, we have:

I. (i) *c*, the number of current clusters; (13a)

II. (ii) Cluster prototypes $V_n = \{\mathbf{v}_{1,n}, \ldots, \mathbf{v}_{c,n}\} \subset \Re^p$; (13b)

III. (iii) a $c \times c$ distance matrix $D(V_n) = [\|\mathbf{v}_{i,n} - \mathbf{v}_{j,n}\|]$; (13c)

IV. (iv) cardinalities, $|v_i| = n_i, i = 1, \ldots, c$; $\sum_{1 \le i \le c} n_i = n$; (13d)

V. (v) current value of $PCAES_n$. (13e)

Historical input data are not kept in memory. After observing $\mathbf{x}_{n+1}$, we update the current value of $PCAES_n$ by finding incremental changes,

$$PCAES_{n+1} = PCAES_n + \Delta(PCAES_n) \quad (14)$$

We will find the increments in (14) by computing equations (5)-(8) incrementally. The first part is the cluster compactness defined in function (7) which measures of the compactness of the i-th fuzzy cluster using its memberships. The numerator is the sum of the squared memberships of cluster i which can be computed incrementally using equations (15)-(16), where $M_{i,n}$ is the value after processing n data samples and $M_{i,n+1}$ is an updated value after $\mathbf{x}_{n+1}$ processed by the algorithm. The initialization and incremental computation of $M_{i,n}$ is shown in algorithm 1 and algorithm 2. The denominator $\mu_{M,n}$ is the most compact cluster among the existing c-clusters, which can be computed incrementally using the $M_{i,n}$ and

$M_{i,n+1}$ as in equations (18)-(19).

$$M_{i,n+1} = M_{i,n} + \Delta M_{i,n} \tag{15}$$

$$M_{i,n+1} = M_{i,n} + (u_{i,n+1})^2 \tag{16}$$

$$u_{i,n+1} = \frac{1}{\sum_{l=1}^{c} \left(\frac{d(x_{n+1}, v_i)}{d(x_{n+1}, v_l)}\right)^{\frac{2}{m-1}}} ; 1 \le i \le c; m > 1 \tag{17}$$

$$\mu_{M,n} = \min_{1 \le i \le c_n} \left\{ M_{i,n} \right\} \tag{18}$$

$$\mu_{M,n+1} = \min_{1 \le i \le c_n} \left\{ M_{i,n+1} \right\} \tag{19}$$

After processing the data points in the initial window S, we have the number of clusters *c*, a label vector $\mathbf{U}^{(1)}$ and initial prototypes $V_1 = \{\mathbf{v}_{1,1}, \ldots, \mathbf{v}_{c,1}\}$. From this configuration, we compute initial values as in algorithm 1 to initialize algorithm 2, where we use part of the initialization process proposed in [7]:

```
Input: $\mathbf{U}^{(1)}, V_1, c \in \{1, \ldots, n\}$
For i =1 to c
$M_{i,1} = |v_{1,i}|$, number of data points in cluster i
Next i
```

Algorithm 1: Parameters initialization

Beginning with the newly arriving data point, streaming inputs can be sent to algorithm 2, which uses $U^{(n+1)}$ to produce the inputs shown, also similar to part of the algorithm in [7]:

Algorithm 2: Incremental compactness for x_{n+1}

```
Data: $c, \mathbf{U}^{(n+1)}, \forall i : M_{i,1}$
Input: $\forall i : M_{i,n}$
Output: $\forall i : M_{i,n+1}$
For i = 1 to c
        $M_{i,n+1} = M_{i,n} + (u_{i,n+1})^2$
Next i
```

In the next section we develop incremental versions of the PCAES index.

6.6 Incremental Partition Coefficient and Exponential Separation Index

Since the data vectors are arriving in a streaming fashion, the iCVI needs to be computed incrementally. After n data samples have been processed, producing the current (not necessarily saved, but it could be) partition U_n, current prototypes V_n, sum of squared memberships of each cluster M_n, and total average of current clusters $\overline{v_n}$ which are substituted into (23) as follows:

$$\mathrm{iPCAES}_{i,n} = \frac{M_{i,n}}{\mu_{M,n}} - \exp\left(-\frac{\min\limits_{k \neq i}\{\|v_{i,n} - v_{k,n}\|\}}{\beta_{T,n}} \right) \tag{20}$$

$$\beta_{T,n} = \frac{\sum\limits_{l=1}^{c} \left\| v_{l,n} - \overline{v_n} \right\|^2}{c} \tag{21}$$

$$\overline{v_n} = \frac{\sum\limits_{i=1}^{c} v_{i,n}}{c} \tag{22}$$

$$\mathrm{iPCAES}_n = \sum_{i=1}^{c} \frac{M_{i,n}}{\mu_{M,n}} - \sum_{i=1}^{c} \exp\left(- \frac{\min_{k \neq i}\{\|v_{i,n} - v_{k,n}\|\}}{\beta_{T,n}} \right) \tag{23}$$

We do have the current prototypes V_n, and the current values of iPCAES_n at (23), but not the data samples $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and the membership matrix U_n. Now $\mathbf{x}_{n+1}$ arrives. Suppose that the online clustering algorithm generates a label vector $\mathbf{U}^{(n+1)}$. Then, the prototypes $\mathbf{v}_n$ are updated to $\mathbf{v}_{n+1}$ using (2). The average of the updated prototypes is computed using equation (27). Algorithm 2 provides us with $\{M_{i,n+1} : 1 \le i \le c\}$. $M_{i,n+1}$ and $\mathbf{v}_{n+1}$ are all we need to compute the incremental updates to (23) as in equation (24):

$$\Delta(\mathrm{PCAES}_n) = \sum_{i=1}^{c} \frac{M_{i,n+1}}{\mu_{M,n+1}} - \sum_{i=1}^{c} \exp\left(- \frac{\min_{k \neq i}\{\|v_{i,n+1} - v_{k,n+1}\|\}}{\beta_{T,n+1}} \right) \tag{24}$$

$$\mu_{M,n+1} = \min_{1 \le i \le c}\left\{M_{i,n+1}\right\} \tag{25}$$

$$\beta_{T,n+1} = \frac{\sum_{l=1}^{c} \left\| v_{l,n+1} - \overline{v_{n+1}} \right\|^2}{c} \tag{26}$$

$$\overline{v_{n+1}} = \frac{\sum_{i=1}^{c} v_{i,n+1}}{c} \tag{27}$$

Combining (14) and (24) produces the incremental form of PCAES_{n+1}. We will compare the performance of this max-optimal iPCAES to the incremental min-optimal iDB developed in [7] and generalized in [13]:

$$iDB_{n+1} = \frac{1}{c}\sum_{i=1}^{c}\underbrace{\max}_{j,j\neq i}\left\{\frac{\dfrac{C_{i,n+1}}{M_{i,n+1}}+\dfrac{C_{j,n+1}}{M_{ij,n+1}}}{\left\|v_{i,n+1}-v_{j,n+1}\right\|^2}\right\} \qquad (28)$$

where $C_{i,n+1}$ is the dispersion of cluster i after the arrival of data sample $\mathbf{x}_{n+1}$.

For more details about the incremental computation of iDB, one can refer to [7, 13].

6.7 Datasets

Synthetic and real-life datasets are used in the evaluation, all of which are presented in a streaming on-line manner. The first synthetic dataset is S1. It has 1100 data samples in 2-dimensional space generated from 11 Gaussian distributions distributed along a line as shown in the scatter plot in Fig. 6-1-a. The static version of S1 has 11 clusters with 100 samples each as shown in table I. The dataset is ordered such that instances from cluster 1 arrive before instances from cluster 2 and so on. The second synthetic dataset is S2, which has 1100 data points generated from 11 Gaussian distributions. In the static version, S2 contains 11 clusters (with 100 samples each) arranged in a circular shape with one cluster in the middle, as seen in Fig. 6-1-b. The data samples are ordered so that data points from cluster 1 arrive first. After that, 10 data points from the center cluster arrive, followed by vectors from cluster 2. A similar pattern is repeated for the remaining clusters. Datasets S1 and S2 are similar to two data sets used by the authors of [7, 13].

The first real-life dataset used in our evaluation is the LG dataset. It is a collection of weather station nodes in the Le Genepi (LG) region in Switzerland [21]. Two weeks' worth of data is selected from node 18 starting from October 10, 2007. Average humidity (H) and surface temperature (T) readings over 10-minute intervals were used to produce a two-dimensional feature vector dataset $x_i = \{(T_i, H_i)\}$. Fig. 6-1-c displays the scatter plot of the data which does not provide clear visual evidence of clusters in the static, batch form of the LG data. However, the imagery information from the site shows that there is a windy and cold day that precedes snowy day during the two-week period. Therefore, we assume the LG data to have three different events: sunny days before and after the snow, cold front moving in, and the snowy day. Different distance threshold values are used in the experiment and the best results (number of clusters) are selected for comparative purposes. The last real-life dataset is the Heron Island Weather station dataset, which is collected from the Heron Island weather station on the Great Barrier Reef, Australia [22]. The Heron Island data has three features: air temperature (T), air humidity (H), and air pressure (P). The data were collected every 10 minutes from 9:00 am to 3:00 pm each day for the 30 days beginning 21-Feb-2009 and ending 22-Mar-2009. In [22], ellipsoidal chains for anomaly detection was used to study Heron Island dataset. There are two (visual) abnormal patterns (small clusters) in this dataset where the first abnormal cluster occurs from data points 445 to 476 (5th March 9:00 AM to 2:10 PM) corresponding to the weather event "Cyclone Hamish". The second abnormal cluster starts from data points 593 to 629 (9th March) which is due to an unusual weather variation in T and H having regular P. Fig. 6-1-d displays a scatter plot of the Heron Island dataset. We can notice four visual clusters in this static view of the Heron Island data, two of which represent the normal pattern and the other two (small clusters) represent the abnormal weather patterns. Table 6-1 summarizes the four datasets.

Table 6-1. Summary characteristics of the datasets used in the evaluation.

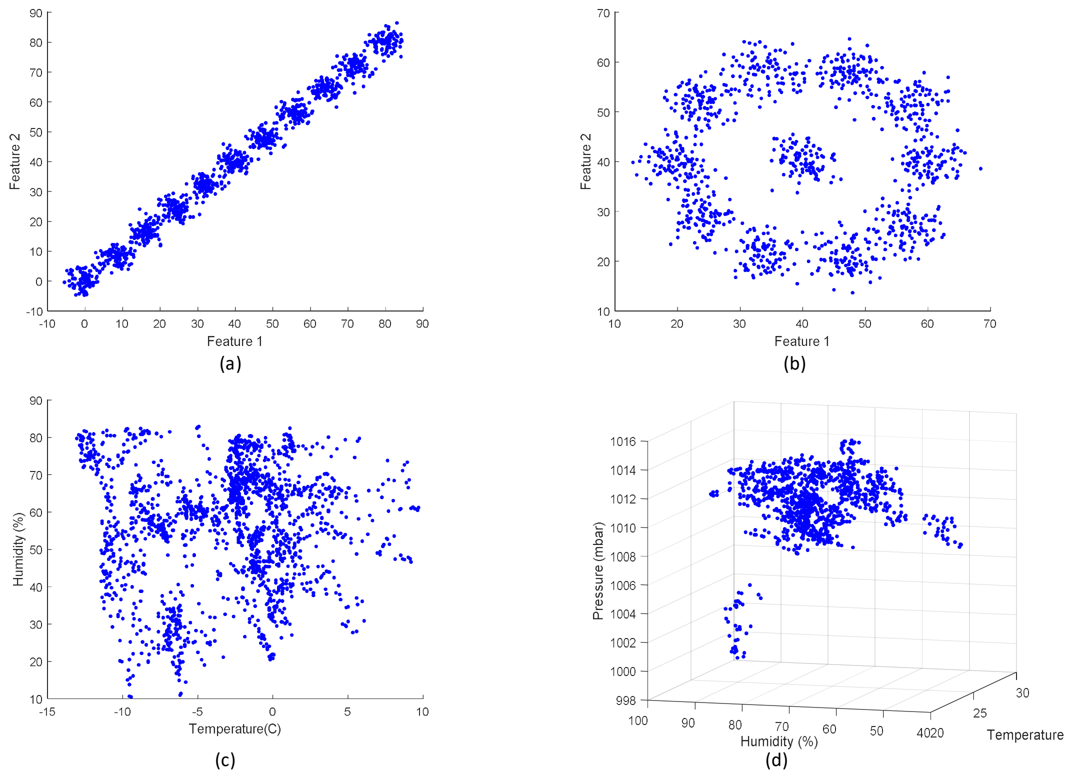


Figure 6-1. Scatter plots of four datasets: a) S1, b) S2, c) Le Genepi weather data, d) Heron Island weather data

Dataset	**# samples**	**# dimensions**	**# clusters**	**# labeled subsets**
S1	1100	2	11	11
S2	1100	2	11	11
LG Weather	1817	2	3	Our Estimate
Heron Island	1110	3	4	Our Estimate

6.8 Initialization

Computation of the iPCAES and iDB indices requires the existence of at least two clusters. Therefore, these indices start after MUSC finds the second cluster in the data stream. However, the parameters are initialized in the window size S (algorithm 1) and updated for each cluster after each data points arrives (algorithm 2). Each time MUSC forms a new cluster, its parameters are initialized with $M_{i,1} = |c_{new}|$ (number of elements in the new cluster) as can be seen in algorithm 1. This initialization is repeated each time a new cluster is created. After initialization, MUSC and iCVIs use each new data sample only once and removes it from memory.

6.9 Experimental results

S1and S2 datasets are used to test the performance of the iPCAES index in monitoring the MUSC algorithm. Hardened clustering results of MUSC are shown in Fig. 6-2-a and Fig. 3-a where the algorithm detected all the clusters in both datasets. The graphs in Fig. 6-2-b and Fig. 6-3-b show the values of iPCAES index for S1 and S2 datasets where the red vertical lines indicate the times when the algorithm detects a new emerging cluster. The iPCAES has an increasing trend in both S1 and S2 datasets, which satisfies the requirement of being max-optimal index. Sudden drop in the index value usually is associated with changes in the cohesion and separation of the clusters being produced by the clustering algorithm. In general, a sudden change in an online validity index is an indication that a new structure has appeared in the data stream (the algorithm learns the prototype that represents the new emerging cluster). The iDB values are shown in Fig. 6-2-c and Fig. 6-3-c for S1 and S2 datasets respectively. iDB is the most stable among existing iCVIs indices [13, 17-18] and both indices behave in a similar manner. Taking a closer look at the iPCAES graphs, we can see the index value decreases after formation of a new emerging cluster because the new coming data samples fall in the new cluster until we see a knee point in the graph (data point from the new cluster start to arrive as outliers). Before MUSC detects a new structure in the data stream, it flags its new arriving data points as outliers until they are dense enough to be designated as a new cluster (time from the knee point to the next red line). The distance between the red line and the next knee point is almost constant for both iDB and iPCAES, indicating that the dataset has clusters of similar size coming from the same distribution as shown in Fig. 6-4, a zoomed section of Fig. 6-2-b.

Fig. 6-3-d and Fig. 6-3-e are an exploded view of iPCAES and iDB for the middle cluster

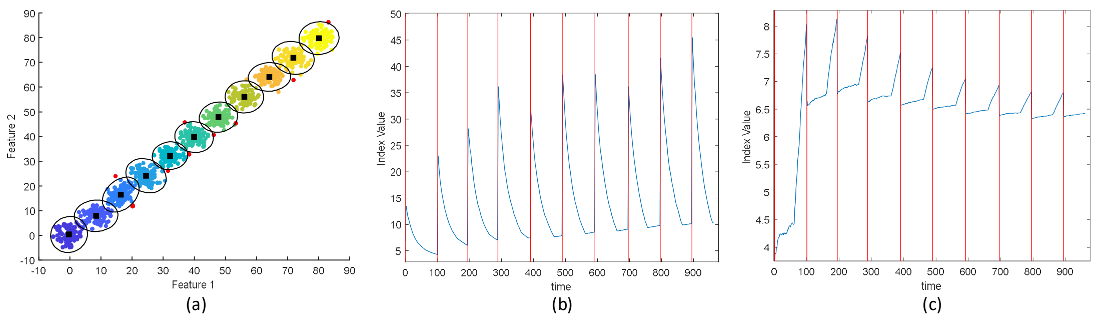


Figure 6-2. Clustering results on S1 dataset: a) MUSC hardened clustering results, b) iPCAES index, c) iDB index

detection in S2. The middle cluster is the hardest one to be detected among other clusters because portion of its data samples (10) arrive after each other cluster. iPCAES detects the middle cluster correctly as in Fig. 6-3-d.

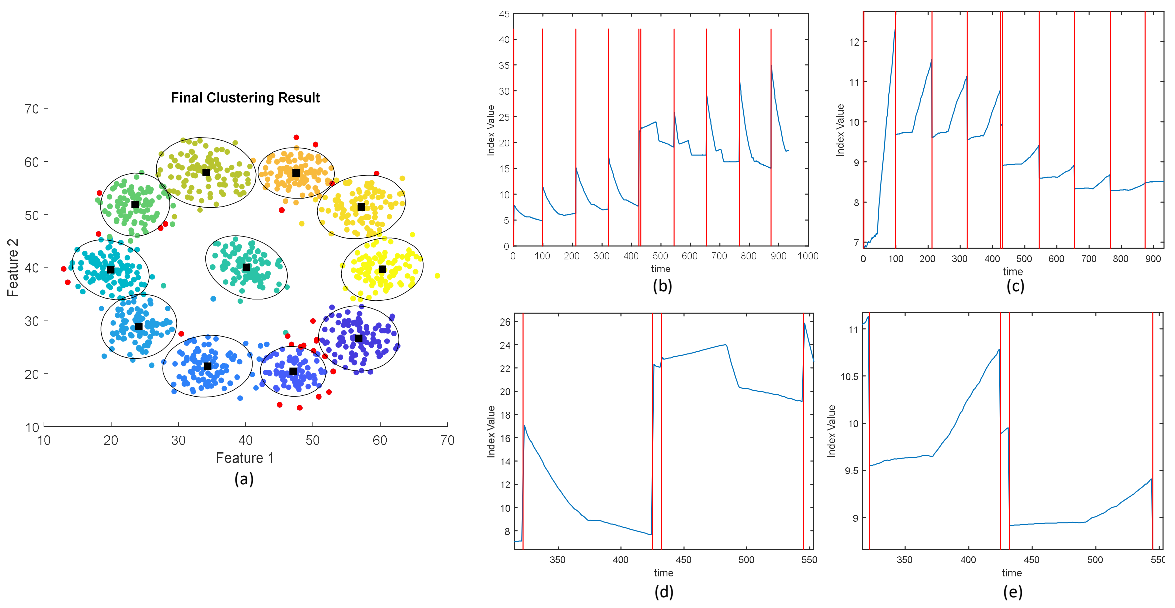


Figure 6-3. Clustering results on S2 dataset: a) MUSC hardened clustering results, b) iPCAES index, c) iDB index, d) Zoom of the region containing the line pair in (b), e) Zoom of the region containing the line pair in (c)

After investigating the performance of the proposed iCVI on synthetic datasets, we test it with two real datasets: LG and Heron. The LG weather dataset has three different events. MUSC (hardened) clustering results are displayed in Fig. 6-5-a where it detects all the event regimes. As we mentioned earlier, computations of iPCAES and iDB require the existence of two clusters. Therefore, we see only 2 structures (red lines) in Fig. 6-5-b and 6-5-c, but MUSC detected all the three structures (Fig. 6-5-a). iPCAES and iDB indicate the time when the third cluster is formed (cluster on the left). As we noticed earlier in the S1 and S2 datasets, the iPCAES value decreases after the formation of a new cluster (red lines in Fig. 6-2 and Fig. 6-3) while iDB stays constant. We see the same performance for iPCAES on the LG weather dataset (Fig. 6-5-b) compared to S1 and S2, but iDB appears to have an increase in its value after the second red line (Fig. 6-5-c). This could indicate that the new proposed index is more stable and consistent than iDB.

[30] The results of MUSC on the Heron dataset are shown in Fig. 6-5-d where all the possible events (four clusters) are detected. Both iCVIs show the time when each new structure detected on the data stream. The clusters are streamed in the order shown in Fig. 6-5-d where clusters 3 and 4 are smaller in size compared to 1 and 2. We can see that there is no drop in the iPCAES value after detecting the third cluster (second red line in Fig. 6-5-e). The reason behind that is the cluster is only detected after all of its points have arrived. Then the newly arriving data fits into clusters 1 and 2. The cardinality of cluster 4 is larger than the cardinality of cluster 3 which allows some newly arriving data points to fit into this cluster after it is formed. Therefore, we see a drop in the index value after cluster formation (last red line) in Fig. 6-5-e. However, the index value rises after the drop because data samples after that belong to cluster 1 and 2. Therefore, based on this observation, using iPCAES index can provide more information about the data stream order which cannot be noticed from iDB (Fig. 6-5-f). Furthermore, iPCAES has an increasing trend in both

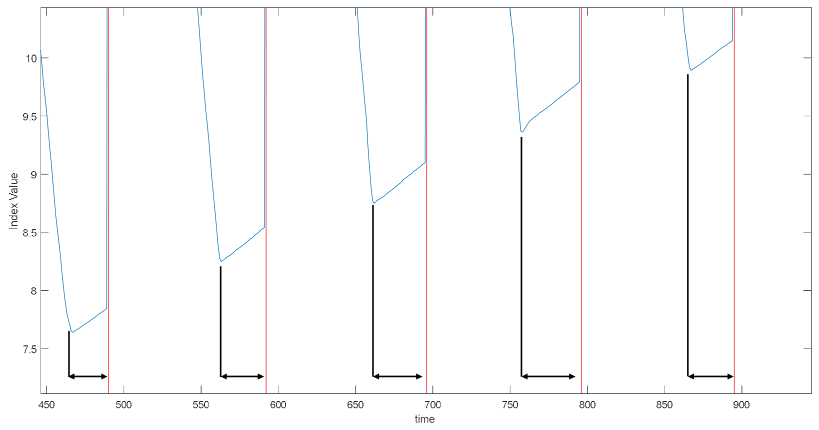


Figure 6-4 Exploded view of Fig. 2-b (iPCAES index on S1 dataset) from time 450 to time 900 which shows similar distance from the knee point and the next red line

synthetic and real-life datasets, which meets the requirements of max-optimal index and proves the robustness of the proposed measure. On the other hand, iDB is supposed to have a decreasing trend (min-optimal) which has not achieved in the real-life datasets.

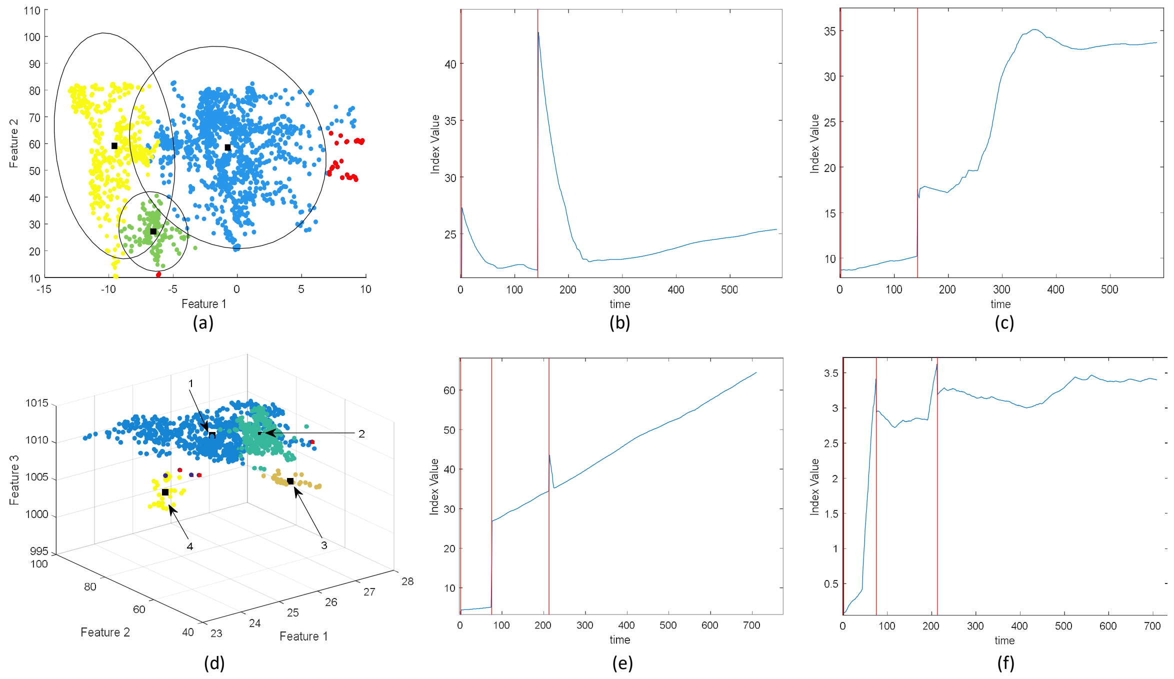


Figure 6-5 Clustering results on real-life datasets: a) MUSC hardened clustering results on weather dataset, b) iPCAES index on weather dataset, c) iDB index on weather dataset, d) MUSC hardened clustering results on Heron dataset, e) iPCAES index on Her

6.10 Conclusions

Incremental cluster validity indices can be employed as an unsupervised approach to monitor the performance of online clustering algorithms as a data stream evolves. In this article, we developed an online incremental version of the *Partition Coefficient and Exponential Separation* (PCAES) index. iPCAES can be used for dynamic assessment of evolving structures in streaming data. The MUSC algorithm was used as our online clustering algorithm, which performs well on all the four datasets used in this paper. The proposed iCVI was compared with the incremental Davies-Boudin (iDB) index because it has been the most stable among other iCVIs, and hence, it offers a comparable approach.

We investigated how iPCAES can be used to monitor the performance of online clustering algorithms with respect to the appearance of new emerging structures in the data stream, outlier data points and evolving clusters, and the effect of clusters size and streaming data order in the iCVIs value. We also studied the knee points in the iCVIs graphs, which indicates the time when MUSC starts accumulating data points for a possible emerging structure. We conclude that iCVIs are effective in real time monitoring of data stream, for example, to send distress signals.

From the results of the real-life datasets, iPCAES provides more information about the data samples streaming order and clusters size than does iDB, which is a clear reason for the need of multiple iCVIs. This is similar to the case of best use of traditional CVIs. Also, iPCAES satisfies the requirements of increasing trend in both synthetic and real-life datasets indicating the robustness of the proposed iCVI as compared to iDB.

In a current project, funded by the National Library of Medicine, we use sensor data to infer change of health status of elderly living alone in their home [23]. Using streaming clustering together with an incremental CVIs could help us detect a new health pattern and provide early illness recognition.

6.11 ACKNOWLEDGMENT

The authors were funded in part by the National Library of Medicine grant #R01LM012221. Omar Ibrahim would like to acknowledge the HCED-Iraq for funding him during his graduate study.

6.12 References

[1] Aggarwal, Charu C., ed. Data streams: models and algorithms. Vol. 31. Springer Science & Business Media, 2007.

[2] M. Ackerman and S. Dasgupta, “Incremental clustering: The case for extra clusters,” in Neural Information Processing Systems (NIPS), Montreal, Canada, December 2014.

[3] P. Angelov, Evolving Takagi-Sugeno Fuzzy Systems from Streaming Data (eTS+). John Wiley & Sons, Inc., 2010, pp. 21–50.

[4] M. Moshtaghi, J. Bezdek, and C. Leckie, “Online clustering of multivariate time-series,” in Proceedings of SIAMConference on Data Mining, Florida, USA, May 2016.

[5] O. Arbelaitz, I. Gurrutxaga, J.Muguerza, J.M. Perez, and I. Perona, “An extensive comparative study of cluster validity indices,” Pattern Recognition, vol. 46, no. 1, pp. 243 – 256, 2013.

[6] Bezdek, J. C. (2017). A Primer on Cluster Analysis: Four Basic Methods that (Usually) Work, First Design Publ., Sarasota, FL.

[7] M. Mostaghi, J. C. , Bezdek, S. Erfani, C., Leckie and J. Bailey, "Online Cluster Validity Indices for Streaming Data," Inter. Jo. Intell. Systems, in press.

[8] D. L. Davies and D. W. Bouldin, "A cluster separation measure", IEEE Trans. Patt. Anal. and Mach. Intell., 1(4), 224-227, 1979.

[9] X. L. Xie and G. Beni, “A validity measure for fuzzy clustering,” IEEE Trans. Patt. Anal. and Mach. Intell., 13(8), pp. 841–847, 1991.

[10] K. L. Wu, and M. S. Yang,”A cluster validity index for fuzzy clustering,” Pattern Recognition Letters, 26(9), 1275-1291, 2005.

[11] O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.

[12] O. A. Ibrahim, Y. Du, and J. M. Keller, “Extended robust on-line streaming clustering (EROLSC),” In IPMU, 2018.

[13] O. A. Ibrahim, J. M. Keller, and J. C. Bezdek, “Analysis of Streaming Clustering Using an Incremental Validity Index”, In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1-8, 2018.

[14] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," Psychometrika, 50(2), pp. 159-179, 1985.

[15] E. Dimitriadou, S. Dolnicar and A. Weingessel, "An examination of indexes for determining the number of clusters in binary data sets," Psychometrika, 67(3), pp. 137-160, 2002.

[16] J. C. Dunn, " fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters", J. Cybernetics, 3(3), 1973, pp. 32-57.

[17] O. A. Ibrahim, Y. Wang, J. M. Keller, “Analysis of Incremental Cluster Validity for Big Data Applications,” accepted at the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2018.

[18] O. A. Ibrahim, J. M. Keller, J. C. Bezdek “Evaluating Evolving Structure in Streaming Data with Modified Dunn's Indices,” submitted to IEEE Transaction on Evolving Topics in Computational Intelligence, 2018.

[19] R. Krishnapuram, J. Keller, "A possibilistic approach to clustering," IEEE Trans. Fuzzy Syst., vol. I, no. 2, pp. 98-1 10, May 1993.

[20] Miin-Shen Yang; Chien-Yo Lai, "A Robust Automatic Merging Possibilistic Clustering Method," Fuzzy Systems, IEEE Transactions on, vol. l9, no. l, pp.26-41, Feb. 2011.

[21] SensorScope. http://lcav.epfl.ch/page-86035-en.html, 2007.

[22] Bezdek, J. C., Rajasegarar, S., Moshtaghi, M., Leckie, C., Palaniswami, M., and Havens, T. C (2011). Anomaly detection in Environmental Monitoring Networks, Computational Intelligence Magazine. 6(2), 52-58.

[23] O. Ibrahim, J. Keller, J. and M. Popescu, "Context preserving representation of daily activities in elder care," In Bioinformatics and Biomedicine (BIBM), 2017 IEEE International Conference on (pp. 547-551). IEEE.

Chapter 7: Unsupervised Analysis of Activity Patterns in Eldercare Monitoring

Omar Ibrahim, Mihail Popescu, PhD, James Keller, PhD

University of Missouri, Columbia, MO

Abstract- Eldercare monitoring using non-wearable sensors is an emerging solution for improving care and reducing costs. Abnormal sensor patterns produced by certain resident behaviors could be linked to early signs of illness. We propose an unsupervised method for detecting abnormal sensor patterns based on a sensor sequence clustering approach. A preliminary analysis of the method was conducted on data collected in TigerPlace, an eldercare facility that promotes aging-in-place.

7.1 Introduction

Sensor networks have emerged in the last decade as a possible solution to reducing cost and improving quality of eldercare. Tiger Place is an aging-in-place facility for seniors located in Columbia, Missouri[1]. Each resident included in the study (37 as of March 2017) has a data logger in his/her apartment that collects data from a wireless sensor network. Each sensor network consists of several types of sensors mounted throughout the resident's apartment, including motion and bed sensors. The health data for each resident is stored in a home-grown nursing EHR (see more details at http://eldertech.missouri.edu/papers).

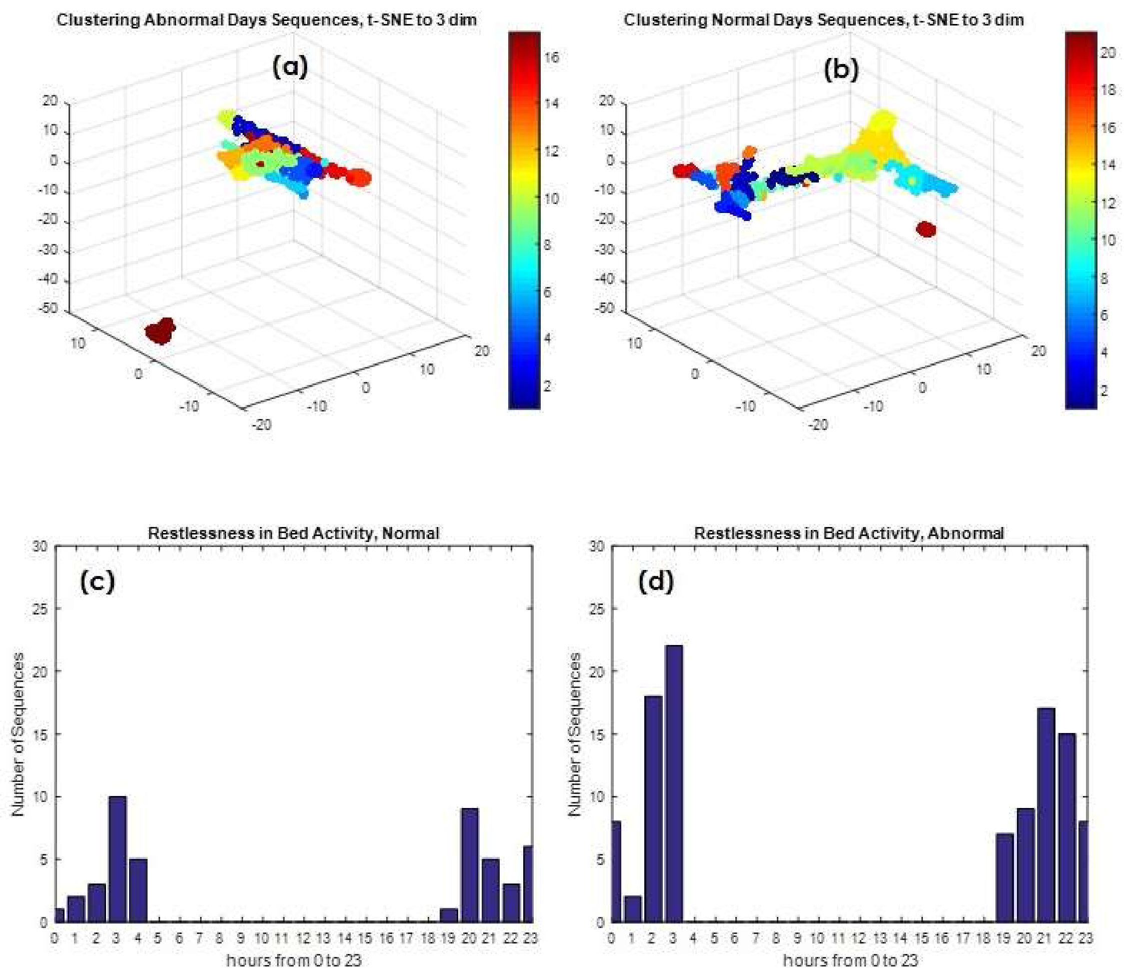


Figure 7-1. 3D visualization of abnormal (a) and normal (b) days using t-SNE projection; time histogram for occurrence of bed restlessness in normal days (c), abnormal days (d).

7.2 Methods

For each day, the sensors firings were converted into sequences of discrete symbols where each symbol represents one sensor type. The idea is to consider the activity (behavior) of a resident represented by a sequence of sensor firings as it would be represented by his/her genome. In this study, we used 11 different bed and motion sensors. We split the daily sensor sequence in subsequences using a separation threshold of 30 seconds which would provide enough granularity to capture daily activities. We used a bag-of-words approach to map each sequence into an 11-D Euclidean space representing the percentage of each symbol in a sequence. Our pilot dataset consisted of 28 days of resident sensor data. To ground truth our approach, we employed a normal/abnormal labeling of each day based on clinically-validated health alerts from our EHR. Days on which a health alert was generated by a fall or other health event were labeled “abnormal”, while days on which no alert was generated were labeled “normal”. The days preceding and following a health alert were excluded from analysis. To explore behavior patterns captured by our sensors, we clustered both the normal and the abnormal datasets using hierarchical clustering and used Calinski-Harabasz index to find the most probable number of clusters.

7.3 Results

In figure 7-1, we show the clusters obtained for normal (1.a) and abnormal (1.b) days projected in 3D using t-SNE[2]. First, there are more clusters (activities) on normal days (21 clusters) than on abnormal days (17). Second, the cardinality of some clusters, such as bed restlessness (figure 7-1 (c and d)) and bathroom visits is greater on abnormal days, while the cardinality of living room activity is greater on normal days. Finally, after generating the time histogram for each cluster, the timing of activity on normal days seems to be more consistent than that of abnormal days. We will use all above observations to differentiate normal from abnormal days and produce more meaningful health alerts.

7.4 Acknowledgment

This work was funded by the National Library of Medicine grant #R01LM012221.

7.5 References

1. Rantz MJ et al, "TigerPlace: A New Future for Older Adults," J. Nrs. Care Qlty., 20:1, pp. 1-4, 2005.
2. van der Maaten et al. “Visualizing High-Dimensional Data Using t-SNE”, J. Mchn. Lrn. Res.:2579-2605, 2008.

Chapter 8: Context Preserving Representation of Daily Activities in Elder Care

Omar A Ibrahim, James Keller

Electrical and Computer Engineering Department

University of Missouri-Columbia

Columbia, Missouri, USA

oai9bc@mail.missouri.edu, KellerJ@missouri.edu

Mihail Popescu

Health Management and Informatics Department

University of Missouri-Columbia

Columbia, Missouri, USA

popescum@health.missouri.edu

Abstract— Eldercare monitoring using non-wearable sensors is a candidate solution for improving care and reducing costs. Abnormal sensor patterns produced by certain resident behaviors could be linked to early signs of illness. We propose an unsupervised method for detecting abnormal behavior patterns based on a new context preserving representation of daily activities. A preliminary analysis of the method was conducted on data collected in TigerPlace, an eldercare facility that promotes aging-in-place. Sensors firings of each day are converted into sequences of daily activities. Using the proposed method, a day with hundreds of sequences is

converted into a single data point representing that day and preserving the context of the daily routine at the same time. We obtained an average Area Under the Curve (AUC) of 0.9 in detecting days where elder adults need to be assessed.

.Keywords— Eldercare monitoring; Wireless sensor networks; Early illness recognition.

8.1 Introduction

Elderly population of aged 65 and older is increasing from 13% in 2010 to 19% in 2030 whereas population of age 15- 65 is decreasing [1]. Usually, older adults prefer to live independently as long as they can despite serious conditions such as risk of falling, dementia, which makes eldercare more challenging. The fear of being institutionalized leads to late health assessments [2], which, in turn, could lead to poor quality of life [3]. Automatic health monitoring systems are a solution to preventing unreported health complications in independently living older adults.

Sensor networks have emerged in the last decade as a possible solution to reducing cost and improving quality of eldercare by monitoring the decline in their functional abilities as well as improving health care of housing, healthcare and social services [4, 5]. MIT's PlaceLab, Georgia Tech's Aware House, Honeywell's Independent Lifestyle Assistant, and University of Missouri’s TigerPlace are examples of monitoring environments [6, 7, 8, 9]. Different approaches for detecting activity and evaluate medication compliance have been reported in the literature [10, 11, 12, 13, 14]. Some illness recognition approaches are focused on either the detection of outliers such as too many bathroom visits [15] or the detection of a set of activities such as walks or falls [16]. Figure 8-1 (solid line) shows the trajectory of typical functional decline in elderly [17]. The typical decline has quasi-plateaus followed by sharp step-downs due to loss of functional ability such as ability to dress, ability to walk, etc. Some of the step-downs are temporary (that is why we used the term "quasi-plateau" above) such as the ability to walk after having a leg injury, before they become permanent. We believe that using sensor-based health assessment and early recognition and/or prediction of health problems we can reduce the functional decline (dotted line curve) and improve the quality of life.

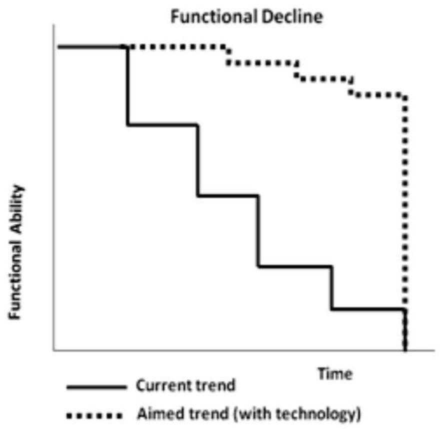


Figure 8-1. Trajectory of typical functional decline the goal with early illness recognition

In our previous work [18], we proposed an unsupervised approach to identify different daily activities. In this paper, we describe a context preserving technique to represent daily activities of older adults. This paper is organized as follows. In Section 8.2, we describe the system architecture and available sensors data. The background is discussed in section 8.3. Section 8.4 presents our method to represent each day in a single data point. Section 8.5 shows experiments and results. Finally, in section 8.6, we give conclusions and future work.

8.2 System Architecture

Tiger Place is an aging-in-place facility for seniors located in Columbia, Missouri [18] where sensor technology is utilized to help elderly manage their illness and stay as healthy and independent as possible. An integrated monitoring system was installed in 47 TigerPlace apartments with the University of Missouri IRB approval. Only non-wearable sensors were used for monitoring because they are more acceptable by older adults and unobtrusive [14, 17]. The monitoring started in fall 2005 with an average of two years' worth of data for each resident.

Figure 8-2 shows the architecture of our monitoring system. The main components of the monitoring system are a sensor network, a data logger, a reasoning system for decline detection and recognition, an electronic health record (EHR) system, an alert manager to inform clinicians of possible problems, and a secure Web-based interface to display the data for the clinicians and researchers. Each sensor network consists of several types of sensors mounted throughout the resident's apartment, including motion (PIR), a Microsoft Kinect and a bed sensor. In this paper, only bed PIR sensor data is used. Each PIR sensor sends an X10 signal that is logged together with a time stamp in our sensor database as can be seen in table 1. Similarly, the bed sensor used in this paper sends abnormal pulse, breathing and restlessness X10 signals. Each resident included in the study has a data logger in his/her apartment that collects data from the wireless sensor network. The computer from each apartment sends data to a main storage server via a secure wired network connection.

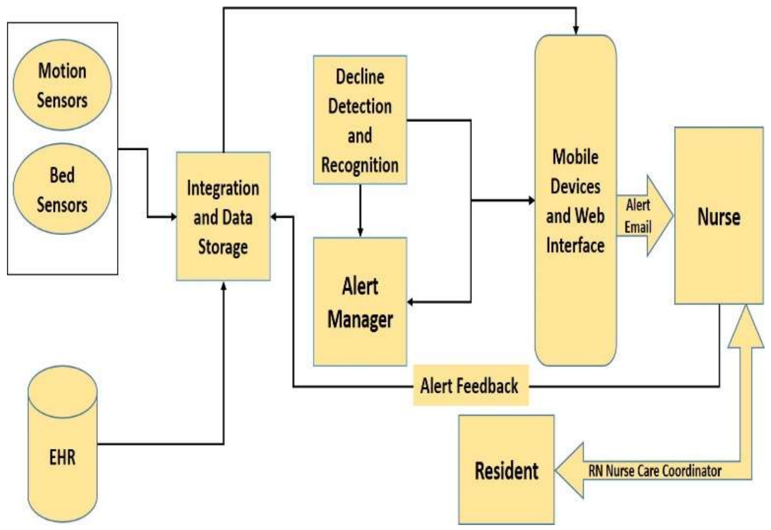

Motion Sensors
Bed Sensors
Integration and Data Storage
Decline Detection and Recognition
Alert Manager
Mobile Devices and Web Interface
Alert Email
Nurse
Alert Feedback
EHR
Resident
RN Nurse Care Coordinator

Figure 8-2. Tiger Place Sensor Network Architecture

Figure 8-3 displays a floor map of a typical TigerPlace apartment that shows the locations of some of the sensors, which are motion and bed sensors in this case.

8.3 Background

Imbalanced data is a common problem in many application domains. When data points of one class in a training data set extremely exceed data points of the other class or classes, traditional classifications algorithms tend to create a biased performance toward the dominant class or classes. To overcome this problem, several techniques have been introduced in the literature such as data sampling and boosting [19].

Oversampling and undersampling are two different ways of data sampling where they balance the class distribution by either adding more data points to the outnumbered class or removing data points from the dominant class. Random undersampling (RUS) is one technique to perform undersampling. RUS has the advantage of reducing the training time due to the reduction in the data size. However, removing data points from the dominant class leads to loss of information [20].

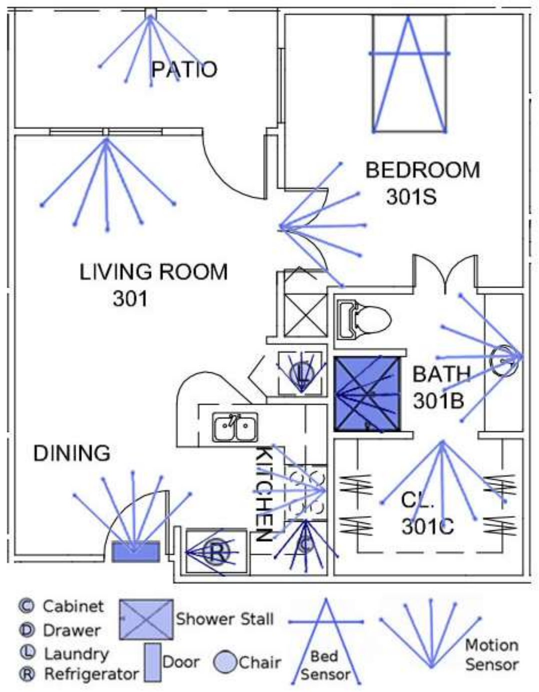


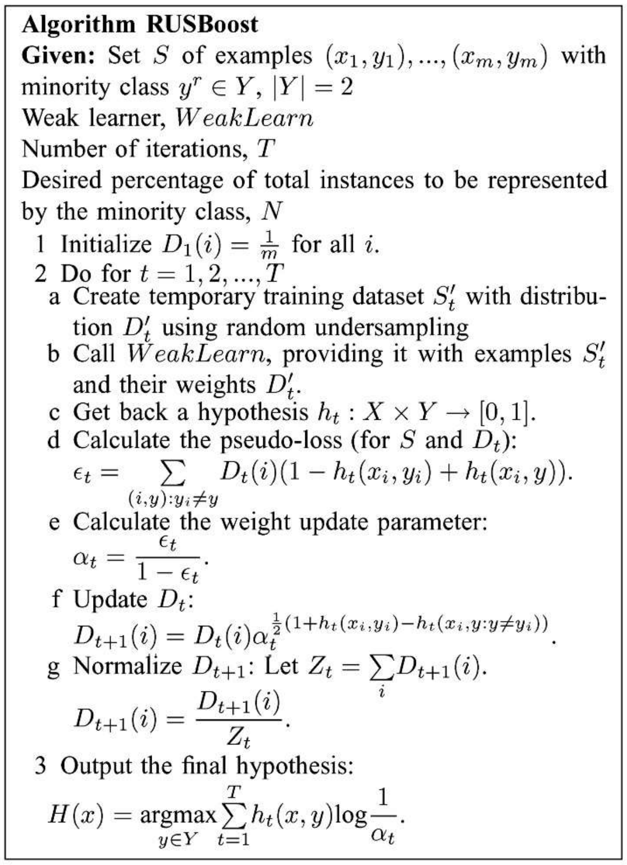
Figure 8-3. Typical apartment sensor layout in TigerPlace

Boosting is designed to improve the performance of weak learners. AdaBoost is the most common boosting techniques. Weights of each data point are adjusted in each iteration to correctly classify data points who were incorrectly classified in that iteration. After training, all weak learners participate in classifying new data points using a weighted vote. Boosting may improve imbalanced data classification because data points of the minority class could be assigned high weights if they were misclassified in previous iterations [21].

In this paper, we use RUSBoost,a combination of data sampling (RUS) and boosting (AdaBoost) algorithm, to improve the performance of models trained on imbalanced data [22] . Figure 8-4 shows the pseudo code of the RUSBoost algorithm. $D(i)_t$ is the weight of the *i*-th data point on iteration *t* and data points are removed from the dominant class until *N*% of the temporary training data set belongs to the outnumbered class. For instance, if the desired class ratio is 50:50, then the dominant class data points are randomly eliminated until majority and minority classes have equal number of data points.

8.4 Method

Table 8-1 shows the set of sensors in the resident's apartment that are used in this paper where (12-14) are coming from the bed sensors [10] and the rest are from motion sensors mounted in different places of the apartment. For each day, the sensor firings are converted into sequences of discrete symbols where each symbol represents one sensor type. The idea is to consider the activity (behavior) of a resident represented by a sequence of sensor firings as it would be represented by his/her genome. We split the daily sensor sequence in subsequences using a separation threshold of 30 seconds, which provides enough granularity to capture daily activities. Using a small separation threshold would break the activities into smaller sub activities, which is not of our interest. In addition, using a large separation threshold results in merging different activities into a single activity, also not desired for our approach. Therefore, 30 seconds worked best for our case after trying different thresholds.



```
Algorithm RUSBoost
Given: Set $S$ of examples $(x_1, y_1), ..., (x_m, y_m)$ with
minority class $y^r \in Y$, $|Y| = 2$
Weak learner, $WeakLearn$
Number of iterations, $T$
Desired percentage of total instances to be represented
by the minority class, $N$
 1 Initialize $D_1(i) = \frac{1}{m}$ for all $i$.
 2 Do for $t = 1, 2, ..., T$
   a Create temporary training dataset $S'_t$ with distribu-
     tion $D'_t$ using random undersampling
   b Call $WeakLearn$, providing it with examples $S'_t$
     and their weights $D'_t$.
   c Get back a hypothesis $h_t : X \times Y \rightarrow [0, 1]$.
   d Calculate the pseudo-loss (for $S$ and $D_t$):
     $\epsilon_t = \sum_{(i,y): y_i \neq y} D_t(i)(1 - h_t(x_i, y_i) + h_t(x_i, y))$.
   e Calculate the weight update parameter:
     $\alpha_t = \frac{\epsilon_t}{1 - \epsilon_t}$.
   f Update $D_t$:
     $D_{t+1}(i) = D_t(i)\alpha_t^{\frac{1}{2}(1+h_t(x_i,y_i)-h_t(x_i,y:y\neq y_i))}$.
   g Normalize $D_{t+1}$: Let $Z_t = \sum_i D_{t+1}(i)$.
     $D_{t+1}(i) = \frac{D_{t+1}(i)}{Z_t}$.
 3 Output the final hypothesis:
   $H(x) = \underset{y \in Y}{\text{argmax}} \sum_{t=1}^{T} h_t(x, y) \log \frac{1}{\alpha_t}$.
```

Figure 8-4. RUSBoost Algorithm [22]

Table 8-2 shows a snippet of the sensor firing data recorded in the log file for a resident of TigerPlace on September 27, 2005, around 01:45 pm where the person had most of the sensors firings on the living room. Sensor ids 1, 3 and 4 are motion sensors on the living room, bathroom and bedroom respectively. Note that the data in Table 3 is a sensor sequence of length 16. Using our threshold separation of 30 seconds (determined empirically) to break the sequence into subsequences of activities, we get three sub sequences of different length ('LLLL', 'LDT', and ‘DLLLLLL’) where L, D and T are symbols annotation of sensors ids 1, 4 and 3 respectively. The first sub sequence 'LLLL' represents a living room activity and 'LDT' is a bathroom visit, which represent a walk from living room to bathroom going through the bedroom. Finally, ‘DLLLLLL’ represents a living room activity after finishing the bathroom visit. In our experiment, sequences of length 2 symbols or less and those of length 50 symbols or more are pruned from each day because they are either not long enough to represent an activity or very long and could be due to nurses or staff visits.

Table 8-1. List of sensors and their IDs used in this paper

ID. Sensor name	6. Patio Motion	11. Breathing
1. Living room Motion	7. Closet Motion	12. Bed Movement
2. Front Door Motion	8. Den Motion	13. Shower Motion
3. Bathroom Motion	9. Office Motion	14. Pulse
4. Bedroom Motion	10. Kitchen Motion	

We used a bag-of-words approach to map each sequence into an M-Dimensional Euclidean space representing the percentage of each symbol in a sequence, where M represents the number of sensors in a particular apartment. For example, the sequences ('LLLL', 'LDT', and ‘DLLLLLL’) would be mapped to [1 0 0 0 0], [1/3 0 1/3 1/3 0], and [6/7 0 0 1/7 0]) if we assumed that there are five sensors in our system. Using this representation, all sensor sequences will have the same length, M.

Table 8-2. A sensor sequence snippet for TigerPlace resident #1

Year	Month	Day	Hour	Minute	Second	Sensor ID	Sequences
2005	9	27	13	44	9	1	Sequence 1
2005	9	27	13	44	15	1	
2005	9	27	13	44	22	1	
2005	9	27	13	44	32	1	
2005	9	27	13	52	35	1	Sequence 2
2005	9	27	13	52	40	4	
2005	9	27	13	52	45	3	
2005	9	27	13	54	51	3	Not a valid sequence
2005	9	27	13	54	54	4	
2005	9	27	13	55	30	4	Sequence 3
2005	9	27	13	55	36	1	
2005	9	27	13	55	46	1	
2005	9	27	13	55	53	1	
2005	9	27	13	56	1	1	
2005	9	27	13	56	9	1	
2005	9	27	13	56	16	1	

To explain the proposed method, we will use an example from table 8-2. Let day 09-27-2005 have three sequences from five different sensors as in table 3. We first generate the histogram for each feature using 5 bins, that is (0 - 0.2, 0.2 - 0.4, 0.4 - 0.6, 0.6 - 0.8, and 0.8 - 1) as in Table 8-3. Then, the histograms of all features were concatenated together to generate a single feature vector representing the entire day. Using this representation, we preserve activity context since the histograms are based on the sequences (activities) performed that day.

Table 8-3. Example explaining the process of converting sequences of a day into a single data point

Day X Numeric Data Representation				
Feature 1 (L)	Feature 2	Feature 3 (T)	Feature 4 (D)	Feature 5
1	0	0	0	0
1/3	0	1/3	1/3	0
6/7	0	0	1/7	0
Histogram 1	Histogram 2	Histogram 3	Histogram 4	Histogram 5
0 1 0 0 2	0 0 0 0 0	0 1 0 0 0	1 1 0 0 0	0 0 0 0 0
Concatenated Histograms to Get a Single Data Point Representation				
0 1 0 0 20 0 0 0 00 1 0 0 01 1 0 0 00 0 0 0 0				

The ground truth was provided by labeling each day as normal/abnormal based on clinically validated health alerts from our EHR. Days on which a health alert was generated by a fall or other health event were labeled “abnormal”, while days on which no alert was generated were labeled “normal”. The days preceding and following a health alert were excluded from analysis.

Using single data representation for each day enables us to use existing (numeric) classification algorithms. RUSBoost is used to validate the proposed representation by computing the area under the receiver operating characteristic (ROC) curve.

8.5 Experimental Results

To validate our approach, we used three different residents with different number of normal and abnormal day as shown in table 8-4. We conducted three separate experiments on each resident to test the performance of our algorithms for finding abnormal days. The reason behind person-based experiments is that, typically, mathematical models (such as classifiers, algorithm parameters, etc.) for early illness recognition based on non-wearable sensors are not transferrable from one person to another, due to different disease-behavior associations that could be quite different between people. For each resident, sensors firings were converted into sequences then into a single data vector representing that day. Then, RUSBoost was used with 10-fold cross validation to generate the ROC curves.

Table 8-4. Dataset used in this paper

Resident	Number of Sensor Days	Number of Alert Days
1	440	81
2	745	35
3	500	335

Figure 8-5 shows the results in terms of ROC curves. The area under the curve (AUC) for our approach is shown in table 8-5 with an average AUC of 0.9. Resident 3 has the lowest AUC because most of his data has consecutive abnormal days, which makes it harder to detect some of them. However, resident 1 is the youngest and the healthiest of the three, which makes detection of his abnormal days easier due to the large change in activity as compared to normal days. Furthermore, resident 1 does not use walker, which makes his routines different between normal and abnormal days. On the other hand, residents 2 and 3 use walkers that affect their walking, and hence, reducing the variety of their routines. Our detection of abnormal days outperforms [23] where they used the same dataset and achieved an average AUC of 0.79. In addition, these results show a great improvement over our previous approach where each day was represented using the sum of the firings of each sensor during the entire day. The resulting representation was also a vector of size M. However, in that representation, the identity of the daily activities was not preserved [15, 24 and 25].

Table 8-5. Area under the curve of the three residents

Technique	AUC # 1	AUC # 2	AUC # 3
Our approach	91	90	89
Approach [23]	79	78	79

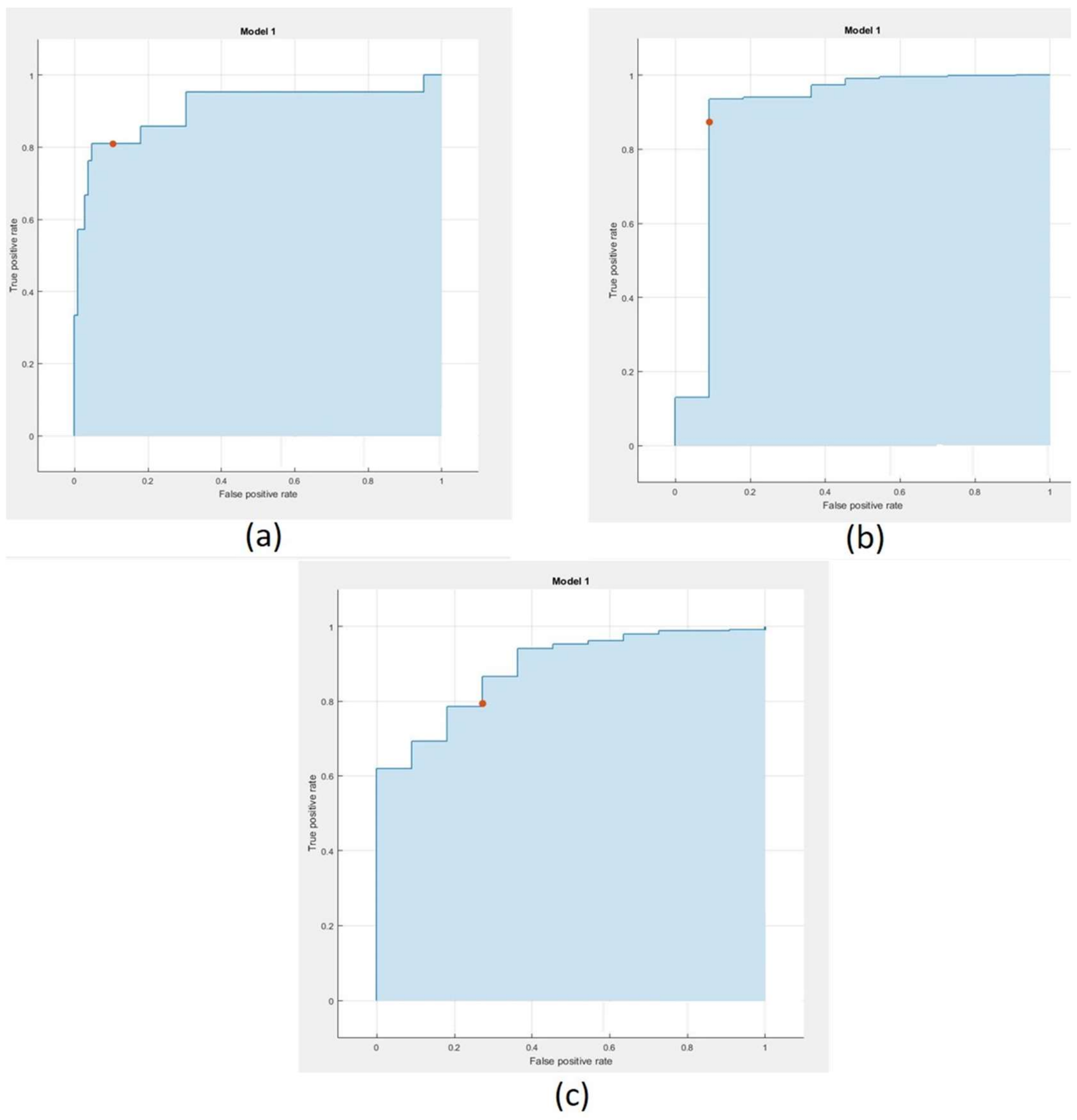


Figure 8-5 ROC Results for a) Resident 1, b) Resident 2, c) Resident 3

8.6 Conclusion

Our proposed computational algorithm successfully detected abnormal days of three TigerPlace residents by applying a new context preserving representation of daily activities. RUSBoost was used for training and testing abnormal day detection due to the issue of our imbalanced dataset. Our average, the detection rate on the three residents was around 80% with a false positive rate of 20%, which is an improvement over previous approaches. Our approach tackles the problem of data inhomogeneity associated with in-home monitoring systems, by converting a day with hundreds of sequences into a single data point representing that day and preserving the context of daily routine at the same time. The proposed algorithm will be integrated into the existing alert system in TigerPlace to detect abnormal days by identifying early functional changes reflected in deviations from daily routines.

8.7 Acknowledgment

This work was funded by the National Library of Medicine grant #R01LM012221.

8.8 References

[1] KV Grayson, AV Victoria, "The next four decades: the older population in the United States: 2010 to 2050", U.S. Department Commerce, Economics and Statistics Administration, U.S. Census Bureau, 2010, pp.I-14.

[2] TL Hayes, M Pavel, JA Kaye, "An unobtrusive in-home monitoring system for detection of key motor changes preceding cogniti ve decline", Proc. of the 26th Annual IntI. Conf. of the IEEE, vol. 1, 2004, pp.2480-2483.

[3] Z Hajihashemi , M Popescu, " Predicting Health Patterns Using Sensor Sequence Similarity and NLP", 2012 IEEE International Conference on BIBM, 2012, pp.948-950.

[4] MJ Rantz, et al., “Improving nurse care coordination with technology”. CIN:Computers, Informatics, Nursing, 2010. 28, p.325-332

[5] J Rowan, ED Mynatt, "Digital Family Portrait field trial: support for Aging in Place", Proc. of the SIGCHI Conf. on Human Factors in Computing Systems, New York,2005, pp. 521-530, ACM Press.

[6] S Intille, K Larson, E Munguia-Tapia, J Beaudin, P Kaushik, J Nawyn, et ai, "Using a live-in laboratory for ubiquitous computing research", Proc. of Pervasive Health, 2006.

[7] CD Kidd, RJ Orr, GO Abowd, IA Atkeson, B Essa, B Macintyre, et aI., "The Aware Home: A living laboratory for ubiquitous computing research", Proc of the 2nd International Workshop on Cooperative Buildings-CoBuild'9, 1999.

[8] KZ Haigh, LM Kift: G Ho, "Independent Lifestyle Assistant: Lessons earned", Assistive Technology, 2006, pp.87-106.

[9] MJ Rantz, et al., "TigerPlace: A new future for older adults," Journal of Nursing Care Quality, vol 20, no. 1, 2005, pp. 1-4.

[10] D Heise, M Skubic, "Monitoring pulse and respiration with a noninvasive hydraulic bed sensor", Proc 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society. Buenos Aires, Argentina. 2010.

[11] R Beckwith, "Designing for ubiquity: The perception of privacy", Pervasive Computing, 2003, pp.40-6.

[12] D Mack, M Alwan, B Turner, R Suratt, R Felder, "A passive and portable system for monitoring heart rate and detecting sleep apnea and arousals: Preliminary validation", Proceedings Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2). 2006.

[13] MJ Rantz, K Donnan-Marek, M Aud, HW Tyrer, M Skubic, G Demiris, et aI., "A technology and nursing collaboration to help older adults age in place", Nursing Outlook, 2005, pp.40-45.

[14] M Skubic, G Alexander, M Popescu, M Rantz, J Keller, "A Smart Home Application to Eldercare: Current Status and Lessons Learned", Journal of Technology and Health Care, 2009, pp.183-201.

[15] M Popescu, A Mahnot, "Early Illness Recognition in Older Adults Using In-Home Monitoring Sensors and Multiple Instance Learning," Methods of information in medicine, vol. 51, no. 4, 2012.

[16] Y Li, KC Ho, and M Popescu "Efficient source separation algorithms for acoustic fall detection using a Microsoft Kinect." IEEE Transactions on Biomedical Engineering, vol. 61, no. 3, 2014, pp.745-755.

[17] MJ Rantz, K Donnan-Marek, M Aud, HW Tyrer, M Skubic, G Demiris, et aI., "A technology and nursing collaboration to help older adults age in place", Nursing Outlook, 2005, pp.40-45.

[18] O Ibrahim, M Popescu, J Keller, " Unsupervised Analysis of Activity Patterns in Eldercare Monitoring," Accepted on American Medical Informatics Association (AMIA) Annual Symposium Proceedings, 2017.

[19] G Weiss, "Mining with rarity: A unifying framework," SIGKDD Explor., vol. 6, no. 1, 2004 pp. 7–19.

[20] G Batista, R Prati, and M Monard, "A study of the behavior of several methods for balancing machine learning training data," SIGKDD Explor., vol. 6, no. 1, 2004, pp. 20–29.

[21] Y Freund and R Schapire, "Experiments with a new boosting algorithm," in Proc. 13th Int. Conf. Mach. Learn., 1996, pp. 148–156.

[22] C Seiffert, T Khoshgoftaar, J Hulse, and A Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol 40, no. 1, 2010, pp.185-197.

[23] Z Hajihashemi , M Popescu, "Detection of abnormal sensor patterns in eldercare."E-Health and Bioengineering Conference (EHB), 2013. IEEE, 2013, pp. 1-4.

[24] M. Popescu, M. Skubic, M. Rantz, "Predicting abnormal clinical events using non-wearable sensors in elderly", proceedings of AMIA Fall Symposium, San Francisco, CA Nov. 14-18, 2009, pp. 1010.

[25] M. Popescu, A. Mahnot, "Early illness recognition in older adults using in-home monitoring sensors and multiple instance learning", IDAMAP-10, Washington DC, Nov 12, 2010, pp. 21-25.

Chapter 9: An Unsupervised Framework for Detecting Early Signs of Illness in Eldercare

Omar A Ibrahim, James Keller
Electrical Engineering and Computer Science Department
University of Missouri-Columbia
Columbia, Missouri, USA
oai9bc@mail.missouri.edu, KellerJ@missouri.edu

Mihail Popescu
Health Management and Informatics Department
University of Missouri-Columbia
Columbia, Missouri, USA
popescum@health.missouri.edu

Abstract—Using non-wearable sensors in eldercare monitoring is a promising solution for improving care and reducing healthcare costs. Abnormal sensor patterns produced by certain resident behaviors can be linked to early signs of illness. We propose an unsupervised framework for detecting abnormal sensor patterns based on clustering activity sensor sequences. We use a 30-day normal window to build a baseline model of an elderly resident by clustering the activity sequences from these days. Each cluster represents different daily activities that are performed in most (normal) days and correspond to normal routines. If a new day contains fewer routine activities, we flag it as abnormal and label the day as one with a possible sign of early illness. A preliminary analysis of the method was conducted on data collected in TigerPlace, an eldercare facility that promotes aging-in-place, with information from our electronic health records (EHR). On a pilot sensor dataset from three residents, with a total of 1902 days, we achieved an average abnormal events prediction of 0.75.

Keywords— Eldercare monitoring, Wireless sensor networks; Activity recognition, Early illness detection.

9.1 Introduction

The elderly population is growing rapidly in America, which causes concern for health care providers. The number of people aged 65 and older is increasing from 13% in 2010 to 19% in 2030 whereas the population of age 15 - 65 is decreasing [1]. Since older adults prefer to live independently, many of the health changes go undetected, such as dementia, frailty and urinary tract infections (UTI) [2]. As average life span in US is about 80 years, most people will live 15 years or more after retirement. A typical functional decline in elderly is shown in figure 9-1 (solid line curve) [3]. The solid line includes plateaus followed by precipitous step-downs due to the loss of functional ability such as ability to dress, ability to walk, etc. The goal of proactive care is to detect and predict health problems early, which could reduce the decline of the functional ability, extend the length of the plateaus and reduce the depth of the steps (dotted line curve in figure 9-1).

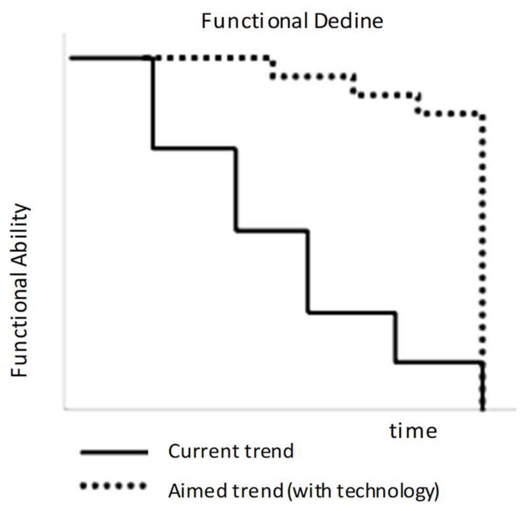


Figure 9-1 Trajectory of typical functional decline the goal with early illness recognition

Detecting health changes early is crucial for promoting health and controlling healthcare costs. On the other hand, late health assessments or unreported problems can lead to poor quality of life [4]. Therefore, automatic health monitoring systems are a possible solution to identify and assess problems in their early stages, which could give more time for the intervention to solve the problems before they become serious.

To promote this model for independent living, sensor networks have been used as a possible solution [5, 6]. MIT's PlaceLab, Georgia Tech's Aware House, Honeywell's Independent Lifestyle Assistant, and University of Missouri’s TigerPlace [7, 8, 9, 10] are few examples of technological solutions for aging-in-place. Different methods for assessing daily activities, walking patterns and medication compliance have been associated with these projects [11, 12, 13, 14, 15,16, 17].

The existing early illness alert system in TigerPlace is univariate, which means that it treats each sensor data stream as an independent variable. The alert system triggers an alert if there is an increase or decrease in the sensor data during a day as compared to the average and the standard deviation from W (a given window size) previous days. However, if there is an increase in multiple sensor values for a day but it is not significant to trigger an alert for any of the sensors, some health changes could potentially go undetected. In [18], it was demonstrated through a survey from our clinical team that some health issues (such as UTI and dementia) in older adults are captured by a combination of sensors. Therefore, we need an alert system that takes into account multiple sensor data changes at the same time to detect early signs of illness. Some illness recognition techniques are focused on either the detection of a set of activities such as walks or a fall [19] or the detection of anomalies such as too many bathroom visits [20]. These techniques are limited to detect abnormal days who have problems associated to these activities. In our previous work [21], we introduced a context preserving measure using daily sequences of activities, which is extended in [22] to annotate days with different health issues. In [23], we used an unsupervised approach to identify different daily activities. In this paper, we extend the idea in [23] and introduce a novel approach to detect (abnormal) days with early signs of illness based on daily activities.

This paper is organized as follows. In Section II we describe the monitoring system architecture and available sensors data. Section III presents our method to detect abnormal days. Section IV shows the metrics that are used to evaluate our approach. Section V describes the clustering results. In Section VI, we describe the experimental results. Finally, in section VII, we give conclusions and future work.

9.2 System Architecture

TigerPlace is an eldercare facility that promotes aging-in-place located in Columbia, Missouri [10]. Sensor technology is applied to help elderly residents manage their illness and stay as healthy and independent as possible. After several focus groups with TigerPlace residents in 2004, a decision was made to use only non-wearable sensors in the monitoring process because they are unobtrusive and more acceptable by older adults [15, 3]. With the University of Missouri IRB approval, our monitoring system was installed in 47 TigerPlace apartments. Data collection has started since fall 2005 with an average of two years' worth of data for each resident.

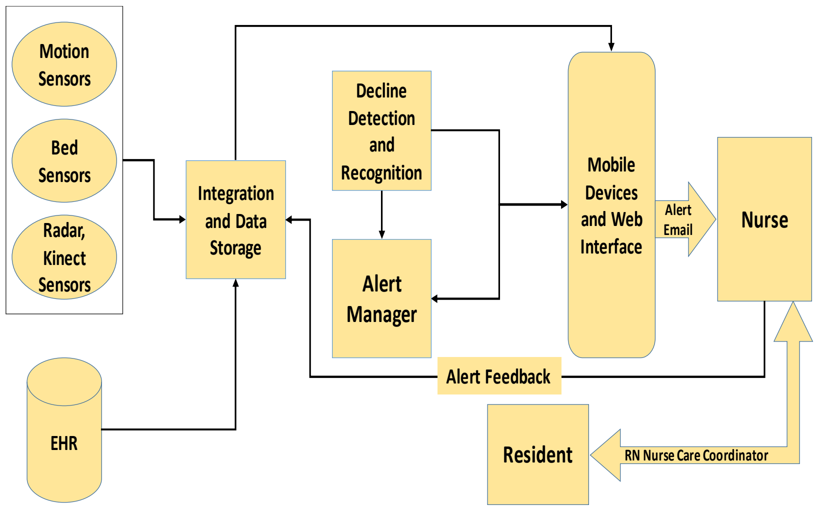

Motion Sensors
Bed Sensors
Radar, Kinect Sensors
EHR
Integration and Data Storage
Decline Detection and Recognition
Alert Manager
Mobile Devices and Web Interface
Alert Email
Nurse
Alert Feedback
Resident
RN Nurse Care Coordinator

Figure 9-2. Tiger Place Sensor Network Architecture.

The architecture of our monitoring system is displayed in figure 9-2. Our monitoring system has the following components: a sensor network, a data logger, a reasoning system for decline detection and recognition, an electronic health record (EHR) system, an alert manager to notify the clinical staff of possible health issues, and a secure Web-based interface to display the data for the researchers and clinicians. Sensor network contains various types of sensors mounted in the resident's apartment, which includes motion, bed, and depth-based video sensors. Each sensor sends a firing with the sensor ID and a timestamp that is logged in our sensor database as can be seen in table 9-2 below. A logger is assigned to each resident included in the study in his/her apartment to collect data from a wireless sensor network. The computer from each apartment is connected to a main storage server via a secure wired network connection. Sensor data of all apartments in TigerPlace are received, stored, and monitored for research purposes. In this study, we use motion and bed (pulse, breathing, and restlessness) sensor data. A typical TigerPlace apartment layout is shown in figure 9-3, which shows the locations of some of the sensors.

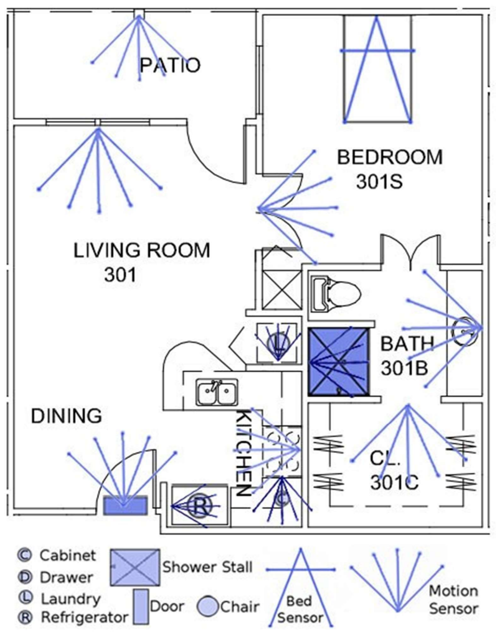


Figure 9-3. Typical apartment layout in TigerPlace

9.3 Method

The set of sensors used in this paper are displayed in table 9-1 where these sensors are invasive and placed in the resident's apartment. Sensor readings (1-3) are computed using the bed sensor [11], and the remaining are from motion sensors. The idea is to capture the resident's daily activities by converting the sensor hits (sensor ID and a timestamp) into a sequence of sensor firings, similar to the representation of his/her genome. Therefore, the sensors readings of each day are converted into discrete symbols where each symbol represents one sensor type. The daily sequence needs to be split into subsequences of different activities using a separation threshold. A small separation threshold results in breaking activities into smaller sub-activities, which is not desirable. Conversely, a large separation threshold merges different activities into a single activity, which is not desired for our approach. Therefore, after trying different thresholds, a 30-second separation provides enough granularity to capture daily activities as demonstrated in [21, 22].

Table 9-1 List of sensors in the resident's apartment

VI. **ID. | Sensors** VII. **firings**	VIII. 5. Patio Motion	IX. 10. Living room Motion
X. 1. Breathing	XI. 6. Closet Motion	XII. 11. Front Door Motion
XIII. 2. Bed Movement	XIV. 7. Den Motion	XV. 12. Bathroom Motion
XVI. 3. Pulse	XVII. 8. Office Motion	XVIII. 13. Bedroom Motion
XIX. 4. Shower Motion	XX. 9. Kitchen Motion	XXI.

A snippet of the sensor firings for a resident in TigerPlace is shown in table 9-2 on September 27, 2005, around 01:45 pm where the person had most of the sensor's firings in the living room. Sensor ids 10, 12, and 13 are motion sensors in the living room, bathroom, and bedroom, respectively. The data in table 9-3 shows a sensor sequence of length 16. Applying our threshold separation of 30 seconds to break the sequence into subsequences of activities, we end up with three sub-sequences of different length ('LLLL', 'LDT', and 'DLLLLLL') where L, D, and T are symbols annotation of sensors ids 10, 13 and 12 respectively. First subsequence 'LLLL' represents activity in the living room area and 'LDT' is a bathroom visit, which is a walk from living room to bathroom going through the bedroom whereas 'DLLLLLL' represents a living room activity after finishing the bathroom visit. Sequences of length 2 or less and those of length 50 or more are removed from each day because they are treated as noise (not long enough to represent an activity or very long and could be due to nurses or staff visits).

Due to the vast variation in activity sequence length, sequence-based similarity measures, such as Smith-Waterman (SWM) [24], will be affected by different sequence lengths. Therefore, we used a bag-of-words approach to map each sequence into an N-dimensional Euclidean space representing the percentage of each symbol in a sequence, where N represents the number of sensors. To explain the proposed method, we use the example from table 9-2. Assume day 09-27-2005 has three sequences from five different sensors as in table 9-3. Each sequence is mapped into a 5-dimensional space (N=5), where each component is the percentage of the occurrence of that symbol in the sequence. Thus, sequences ('LLLL', 'LDT', and 'DLLLLLL') are transformed to [1 0 0 0 0], [1/3 0 1/3 1/3 0], and [6/7 0 0 1/7 0]) as shown in table 9-3. After this process, all the sequences will have the same length. This representation enables us to preserve the activity context because it relates to the sequences (activities) on that day. In addition, this sequence mapping enables us to apply Euclidean distance or other measures to compute similarity/dissimilarity between sequences, which is a key point in any clustering algorithm. Also, dimensionality reduction techniques can be applied directly to the mapped sequences, which makes it easier to visualize the clustering results.

Our goal is to use the activities from normal days to build a base model, which can be used to find the number of activities performed by an elderly resident on normal days. We are looking for two things in a new day: the existence of a specific activity and the frequency of this activity on that day. One month of normal days is used to build the model by concatenating all the sequences together and clustering them using hierarchical clustering with the Ward distance. Calinski-Harabasz cluster validity index [25] is used to find the best number of clusters for our model where each cluster represents a specific activity of the elderly's daily routine. On this approach, the model is updated daily if the new arriving day is determined to be normal, which allows for maintaining the temporal information of the resident because it is expected that behaviors of elderly people change with time according to their health conditions. For example, routines before leg surgery are different from those after the surgery due to the limitations in movement.

Table 9-2. A sensor sequence snippet for a TigerPlace resident

Year	Month	Day	Hour	Minute	Second	Sensor ID	Sequences
2005	9	27	13	44	9	10	Sequence 1
2005	9	27	13	44	15	10	
2005	9	27	13	44	22	10	
2005	9	27	13	44	32	10	
2005	9	27	13	52	35	10	Sequence 2
2005	9	27	13	52	40	13	
2005	9	27	13	52	45	12	
2005	9	27	13	54	51	12	Not a valid sequence
2005	9	27	13	54	54	13	
2005	9	27	13	55	30	13	
2005	9	27	13	55	36	10	

2005	9	27	13	55	46	10	Sequence 3
2005	9	27	13	55	53	10	
2005	9	27	13	56	1	10	
2005	9	27	13	56	9	10	
2005	9	27	13	56	16	10	

Table 9-3. Example explaining the process of converting sequences of a day into a fixed-length data vector

Sequences	**Day X Numeric Data Representation**				
	Feature 1 (L)	**Feature 2**	**Feature 3 (T)**	**Feature 4 (D)**	**Feature 5**
'LLLL'	1	0	0	0	0
'LDT'	1/3	0	1/3	1/3	0
'DLLLLLL'	6/7	0	0	1/7	0

For each resident, the sensor firings of each day are converted into normalized sequences as depicted in table 9-3 (steps 1-2 in figure 9-4). Our approach requires the existence of X normal days (in this case we selected 30) to build the base model which captures number of activities and their frequences performeddaily for each resident as in figure 9-4-step 3. From the histogram of each cluster, the mean (m) and standard deviation (std) of the number of sequences in each cluster of the model (from each day) is computed to provide an idea of how many different activities are performed by that person usually in a normal day (see step 3-a, b, and c of the algorithm from figure 9-4). After a 30 day model is computed, the data from new days is fed in a streaming fashion. To flag a day as abnormal, sequences of that day are mapped to the clusters of the model by computing the Euclidean distance between the normalized sequences of that day and model cluster centers. A rule is built based on the results shown in figure 9-7. If a new day has total number of sequences in any cluster less than (mean - standard deviation) of the members of that cluster generated from the normal period of time, that cluster is flagged as a violated cluster (activity), which means the resident has not performed such activity enough compared to his routine on a normal day.

Also, if the sequences in any cluster are greater than (mean + standard deviation) of the members of that cluster generated from the normal period of time, that cluster is flagged as a violated cluster (activity), which means the resident has performed too many of such activity compared to his routine on a normal day. Furthermore, when a day has two violated clusters or more, it is flagged as an abnormal day (possible early illness) as displayed in step 7. A day that is labeled as normal is swapped with the oldest day (LIFO) in the model and the clustering step (figure 9-4-step 3) is repeated to update our model to adapt the recent resident's health changes (step 8 in figure 9-4).

```
Algorithm: Abnormal day detection using unsupervised approach.
Input: sensor hits of a new arriving day (day-i) and the previous 30 normal days
Output: Label of day-i as normal or abnormal.
Steps:
    1- Convert the sensor firings and time stamps into discretized sequences following the procedure
       in table 2.
    2- Generate the normalized sequences of all days following the approach in table 3.
    3- Build the model:
       a- Cluster the concatenated sequences of the 30 normal days using Ward Clustering
          Algorithm (number of clusters and Cluster representative).
                ✓ Find Best number of Clusters (activities) using Calinski-Harabasz index.
                ✓ Find Cluster representatives
       b- Compute the histogram of number of sequences from each day in each cluster.
       c- From the histogram in (b), find the mean (m) and std for each cluster.
    4- Compute the Euclidean distance between the clusters representatives and the new day's
       sequences, $D=\{d_{ij}\}_{i=1,\dots,n,j=1,\dots,k}$
    5- Map the sequences to their closest clusters
    6- Cluster i is flagged as a violated cluster if number of mapped sequences is 1-std from its mean
       $(\text{mean}_i \pm \text{std}_i)$
    7- If there are 2 or more violated clusters, day-i is labeled as Abnormal, otherwise label it as
       normal.
    8- Model Update:
If day-i is labeled as normal:
Swap it with the oldest day in the model
Else:
    Keep the same days for the model
```

Figure 9-4 Abnormal day's detection algorithm

To ground-truth our approach, we employed a normal/abnormal labeling of each day based on clinically validated health alerts and nursing notes from our EHR. Days on which a health alert was generated by a fall or other health event or if a nursing note is found, were labeled “abnormal”, while days with no alert or nursing note were labeled “normal”.

9.4 Dataset and Evaluation Metrics

In this section, we discuss the dataset used to evaluate our approach in detecting days where the resident needs to be assessed (referred as abnormal days) along with the evaluation metrics employed. We used three different residents with different number of normal and abnormal days, as shown in table 9-4, resulting in a total of 1403 sensor days.

Our goal is to detect days referred to as “abnormal” during which a resident need to be assessed. Considering a day as abnormal is a subjective matter that varies between elderly residents. Therefore, an event that is treated as abnormal for one resident could be a normal day for another resident. For instance, multiple bathroom visits during night for a resident who regularly sleeps soundly during the night is considered as an abnormal event, whereas for another resident, several midnight bathroom visits are considered normal. For these reasons, labeling of normal/abnormal events relies on the functional ability of residents, their health status, medication taken, etc. We use sensitivity and specificity to evaluate our proposed method as defined bellow.

Sensitivity = TP/(TP+FN),

Specificity = TN/(TN+FP).

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

Table 9-4. Pilot dataset of three residents from TigerPlace

Resident ID	Number of Sensor Days	Number of Abnormal Days
1	351	249
2	628	488
3	424	241

9.5 Clustering Results

As discussed earlier, our goal is to cluster the sequences in a window of 30 normal days (no health alerts were reported) to build the baseline model for each resident. Two approaches are applied to measure the distance between the sequences and find the possible clusters (activities): the first approach is to use SWM as a distance measure between the symbolic sequences (as in table 9-2), and the second approach is to map the sequences into the Euclidean space (as in table 9-3) and use Euclidean distance. Ward Hierarchical clustering is employed to cluster the data in both cases along with Calinski-Harabasz validity index [25] to find the best number of clusters (set of activities).

As an example of the effect of the two choices for determining distance, cluster distributions and histograms are shown in figure 9-5 for the case of using SWM as our distance measure on clustering 14 normal days of resident 1 in table 9-4. Figure 9-5-a (left graph) shows the distribution of the sensors in cluster 1, which is the percentage of each sensor in the sequences of that cluster. The x-axis of the distributions (numbers 1:11) corresponds to Front Door, Living Room, Closet, Kitchen, Patio, Shower, Bedroom, Bathroom, Office, Den, and Restlessness sensors and the y-axis is a percentage between 0 and 1. Figure 9-5-a (right graph) displays the histogram of the sequences in cluster 1, which can be used to differentiate between activities performed on daytime from those performed on nighttime. The x-axis of the histograms is a 24-hours period of the day, which corresponds to the occurrence of the sequences of each cluster during the day, while the y-axis is number of sequences in each hour. Cluster 1 in figure 9-5-a has sequences generated from front door, living room, closet, and kitchen sensors performed on the period 5:00 am – 7:00 pm. From the distribution of cluster 2 in figure 9-5-b (left graph), we can see that the sequences are from multiple activities such as bathroom, office, bedroom, shower, etc. It can be noticed from the cluster distributions in figure 9-5-c, d, and e that the clusters do not represent specific activity (sequences from different activities are grouped in the same cluster) except the cluster in figure 9-5-f which represents bathroom activity at night. Restlessness, an important activity that is usually monitored by our healthcare team to detect different health conditions, is missing in this case. The reason behind such poor clustering performance is the difference in sequence length, which makes it not feasible for SWM to generate a meaningful distance between sequences of the same type but with different length.

Therefore, we normalize the above sequences by mapping them to the N-dimensional space as in table 9-3 where N is total number of used sensors (11 in this example). Figure 9-8 shows the cluster distributions and histograms for clustering the same 14 normal days of resident 1, where we can clearly see similar activities are grouped in the same cluster because one or two sensors have the highest percentage in the distribution of each cluster. For example, bathroom visits sequences are grouped together (figure 9-8-b), and bed restlessness sequences are grouped in the same cluster as in figure 9-8-a. A tSNE projection [26] of the sequences is depicted in figure 9-6, which shows that sequences of the same activity are grouped in the same cluster. Therefore, we chose this approach in the next experiment to detect abnormal days.

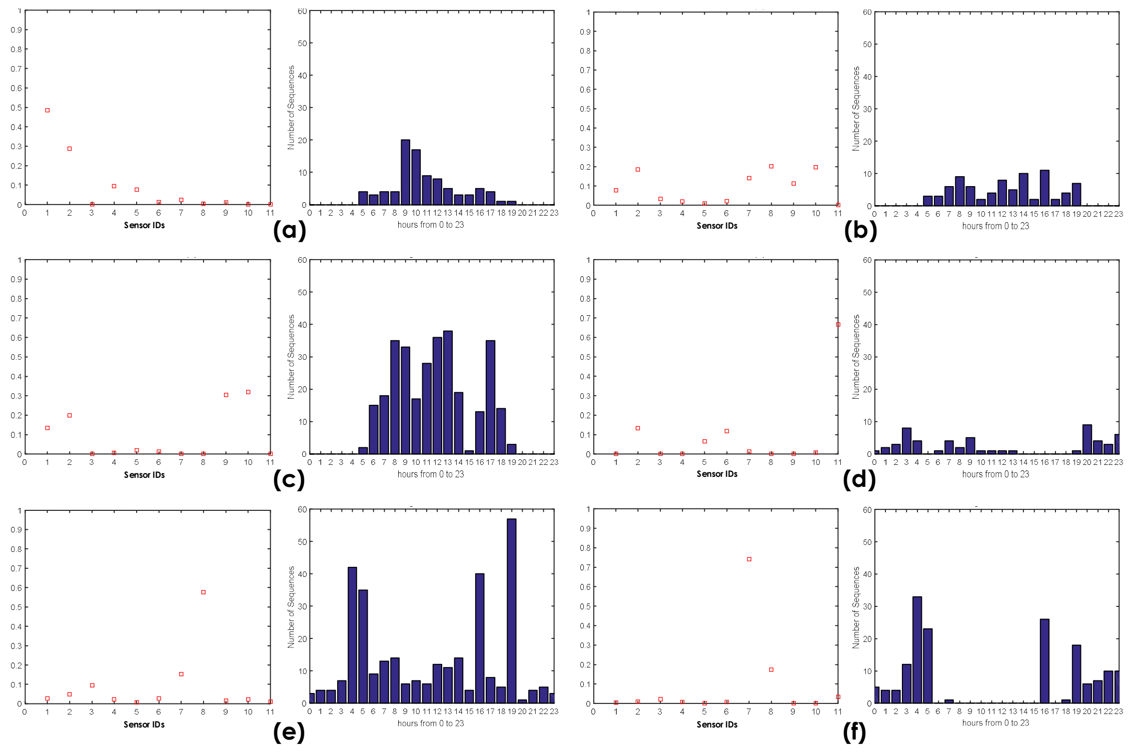


Figure 9-5. Cluster distributions and histograms when using SWM as the distance measure to compute the distance between symbolic sequences for 14 normal days of resident 1 in table 4 .

9.6 Experimental Results

We tested the performance of our algorithm for detecting abnormal days using the data shown in table 9-4. The idea is that, in normal days, elderly people do similar activities within the same period. On the other hand, in days where they are sick, either they do not do some activities as much compared to normal days such as living room activity, or they do them more, for example, bathroom visits.

For each resident, the sensor firings of each day are converted into normalized sequences, as depicted in table 9-3. The new data is fed in a streaming fashion. In order to decide the

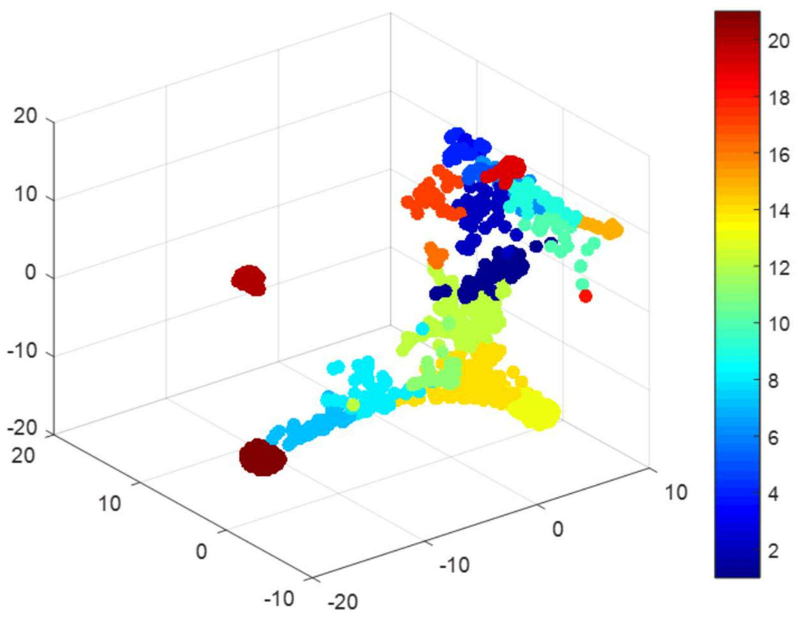


Figure 9-6. t-SNE of 14 normal days projected in 3-d space

number of violated clusters required to flag a day as abnormal, we vary number of violated clusters for resident 1 in table 9-4 and compute the accuracy for each choice of this parameter. As it can be seen in figure 9-7, 2 violated clusters gives the best performance and, hence, it is set at 2 for all experiments.

We run three separate experiments on each resident in table 9-4. Table 9-5 displays the results in terms of sensitivity/specificity with an average sensitivity of 75% and 70% of an average specificity. Our results suggest that, for residents with rigid routines such as resident # 1 and #2, the proposed methods provide good detection of abnormal days. Also, for younger residents, the detection of abnormal days is easier due to the large change in their activity on abnormal days compared to their normal routines. The result shows a significant improvement over our previous approaches [20, 27]. One possible reason behind misdetecting some of the abnormal days is due to the actual problem at that day where medications could help to reduce its effect on the resident's activity. In addition, a day with a health issue that is not reported in the nursing notes could lead not only to misclassification of that day but also misclassification of that “normal” day (because it is actually an abnormal day and flagged as abnormal by our algorithm).

In [20], the authors used bathroom visits as parts of the older adult’s daily routine in their approaches, which simplifies the problem and detect abnormal days with problems associated with such activities only. However, our approach is unsupervised, where we do not use specific activities but rather, a general approach that depends dynamically on the resident’s daily activities. Also, this approach enables us to detect abnormal days with

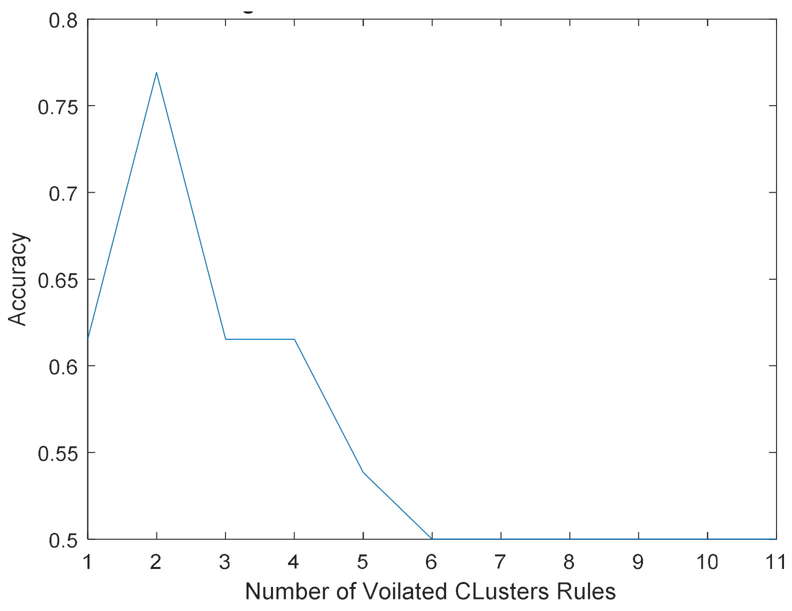


Figure 9-7. Selecting the best number of violated clusters to flag a day as abnormal

different problems that affect the resident’s daily patterns. In [27], the daily representation was a sum of total sonsor firings where the identity of the daily activities was not preserved compared to our approach. Our average accuracy is around 0.75 versus 0.84 in [20] and 0.70 in [27], which is comparable in performance, considering the generalization of our method. Our next step is to evaluate our approach with more residents and deploy it as part of our alert system in TigerPlace.

Table 9-5. Abnormal Day Detection Performance

Resident ID	1	2	3
Sensitivity	75	80	70
specificity	70	72	68

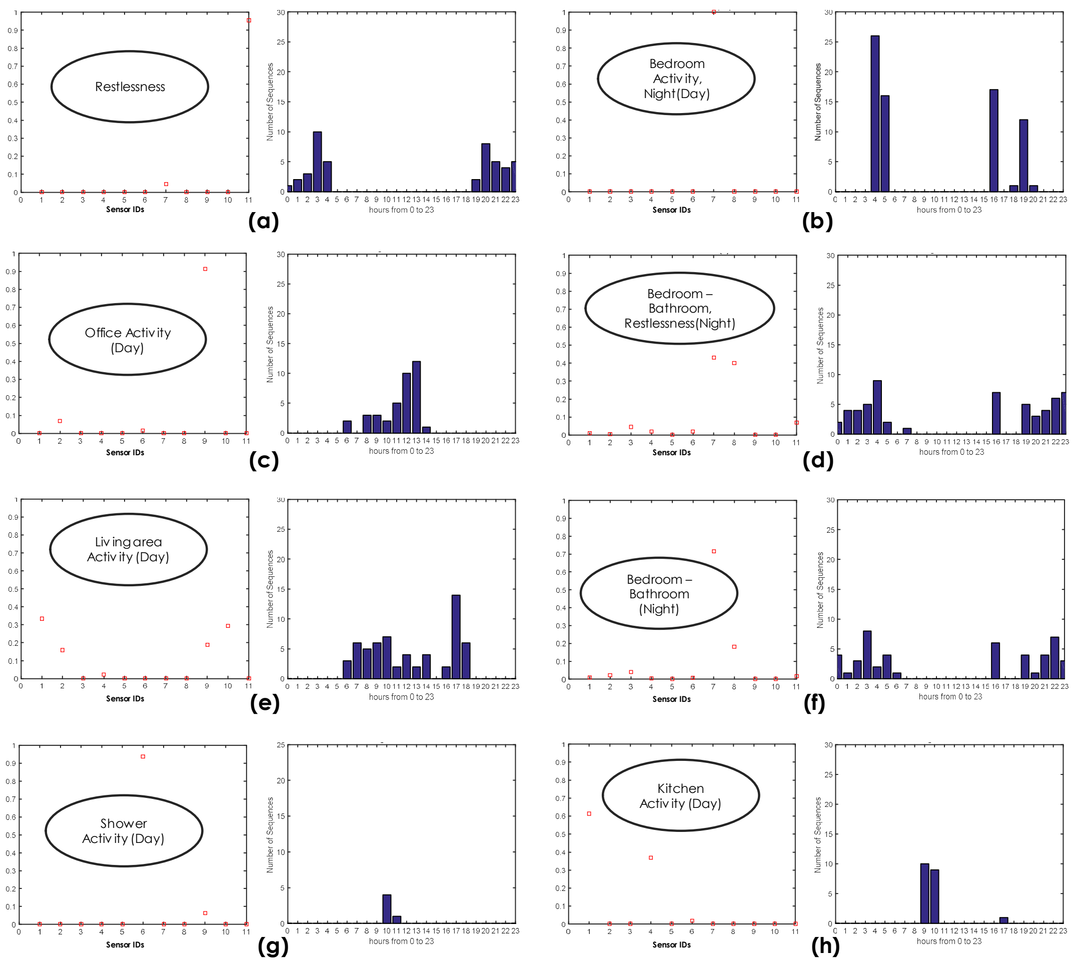


Figure 9-8. Cluster distribution and histograms when using Euclidean distance measure to compute the distance between normalized sequences for 14 normal days of resident 1 in table 9-4.

9.7 Conclusion

Our computational algorithm successfully detected abnormal days of three TigerPlace residents utilizing a completely unsupervised approach based on the resident's daily activities. Ward hierarchical clustering was used together with the Calinski-Harabasz index to build a base model for an elderly resident for normal activity. The mean (m) and standard deviation (std) of number of sequences in each cluster of the model (from each day) were computed, which provides the frequency of each activity performed in a normal day. The sequences of a new day were mapped to the clusters of the model. If a new day has total number of sequences in any cluster that are more than or less than one standard deviation from the mean (m ± std) of the cluster, we assign that as a violated cluster for that day. Furthermore, when we have two or more violated clusters in a day, we flag it as abnormal. Our average detection rate was around 75%, which is very good, particularly given behavior variations after recovering from illness. Our approach is completely unsupervised where we do not search for known activities but, rather a general approach that relies on a resident's daily activities. Furthermore, this method allows us to detect abnormal days with different health issues that affect the resident's daily routine. The proposed algorithm will be integrated into the existing alert system in TigerPlace to detect abnormal days by identifying early functional changes reflected in deviations from daily routines.

9.8 Acknowledgment

This work was funded by the National Library of Medicine grant #R01LM012221.

9.9 References

[1] KV Grayson, AV Victoria, "The next four decades: the older population in the United States: 2010 to 2050", U.S. Department Commerce, Economics and Statistics Administration, U.S. Census Bureau, 2010, pp.I-14.

[2] TL Hayes, M Pavel, JA Kaye, "An unobtrusive in-home monitoring system for detection of key motor changes preceding cogniti ve decline", Proc. of the 26th Annual IntI. Conf. of the

[3] MJ Rantz, K Donnan-Marek, M Aud, HW Tyrer, M Skubic, G Demiris, et aI., "A technology and nursing collaboration to help older adults age in place", Nursing Outlook, 2005, pp.40-45.

[4] Z Hajihashemi , M Popescu, " Predicting Health Patterns Using Sensor Sequence Similarity and NLP", 2012 IEEE International Conference on BIBM, 2012, pp.948-950.

[5] Rantz, M.J., et al., “Improving nurse care coordination with technology”. CIN:Computers, Informatics, Nursing, 2010. 28, p.325-332

[6] J Rowan, ED Mynatt, "Digital Family Portrait field trial: support for Aging in Place", Proc. of the SIGCHI Conf. on Human Factors in Computing Systems, New York,2005, pp. 521-530, ACM Press.

[7] S Intille, K Larson, E Munguia-Tapia, J Beaudin, P Kaushik, J Nawyn, et ai, "Using a live-in laboratory for ubiquitous computing research", Proc. of Pervasive Health, 2006.

[8] CD Kidd, RJ Orr, GO Abowd, IA Atkeson, B Essa, B Macintyre, et aI., "The Aware Home: A living laboratory for ubiquitous computing research", Proc of the 2nd International Workshop on Cooperative Buildings-CoBuild'9, 1999.

[9] KZ Haigh, LM Kift: G Ho, "Independent Lifestyle Assistant: Lessons earned", Assistive Technology, 2006, pp.87-1 06.

[10] MJ Rantz, et al., "TigerPlace: A new future for older adults," Journal of Nursing Care Quality, vol 20, no. 1, 2005, pp. 1-4.

[11] D Heise, M Skubic, "Monitoring pulse and respiration with a noninvasive hydraulic bed sensor", Proc 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society. Buenos Aires, Argentina. 2010.

[12] R Beckwith, "Designing for ubiquity: The perception of privacy", Pervasive Computing, 2003, pp.40-6.

[13] D Mack, M Alwan, B Turner, R Suratt, R Felder, "A passive and portable system for monitoring heart rate and detecting sleep apnea and arousals: Preliminary validation", Proceedings Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2). 2006.

[14] MJ Rantz, K Donnan-Marek, M Aud, HW Tyrer, M Skubic, G Demiris, et aI., "A technology and nursing collaboration to help older adults age in place", Nursing Outlook, 2005, pp.40-45.

[15] M Skubic, G Alexander, M Popescu, M Rantz, J Keller, "A Smart Home Application to Eldercare: Current Status and Lessons Learned", Journal of Technology and Health Care, 2009, pp.183-201.

[16] O. A. Ibrahim, J. Shao, J. M. Keller and M. Popescu, “A temporal analysis system for early detection of health changes," In IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 186-193, 2016.

[17] O. A. Ibrahim, Y. Du, and J. M. Keller, “Extended robust on-line streaming clustering (EROLSC),” In IPMU, 2018.

[18] M Popescu, A Craver, L Phillips, R Koopman, G Alexander, L Despins, M Rantz, " Linking Resident Behavior to Health Conditions in an Eldercare Monitoring System," Accepted on American Medical Informatics Association (AMIA) Annual Symposium Proceedings, 2017.

[19] Li, Yun, KC Ho, and M Popescu "Efficient source separation algorithms for acoustic fall detection using a Microsoft Kinect." IEEE Transactions on Biomedical Engineering , vol. 61, no. 3, 2014, pp.745-755.

[20] Hajihashemi, Zahra, Maria Yefimova, and Mihail Popescu. "Detecting daily routines of older adults using sensor time series clustering." In 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 5912-5915. IEEE, 2014.

[21] O. A. Ibrahim, J. Keller and M. Popescu, "Context preserving representation of daily activities in elder care," 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Kansas City, MO, USA, pp. 547-551, 2017.

[22] O. A. Ibrahim, M. Popescu, and J. Keller, " An Automatic Framework for Semantic Annotation of Eldercare Sensor Data," submitted to BMC Medical Informatics and Decision Making, 2019.

[23] O A. Ibrahim, M Popescu, J Keller, " Unsupervised Analysis of Activity Patterns in Eldercare Monitoring," on American Medical Informatics Association (AMIA) Annual Symposium Proceedings, 2017.

[24] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences,” J. Molecular Biol., vol. 147, pp. 195–197, 1981.

[25] T. Calinski and J. Harabasz. A dendrite method for cluster analysis. Communications in Statistics, 3, no. 1:1–27, 1974.

[26] L Maaten and G Hinton, "Visualizing data using t-SNE," Journal of machine learning research, vol. 9, pp. 2579-2605, 2008.

[27] M Popescu, A Mahnot, "Early Illness Recognition in Older Adults Using In-Home Monitoring Sensors and Multiple Instance Learning," Methods of information in medicine, vol 51, no. 4, 2012.

Chapter 10: An Automatic Framework for Semantic Annotation of Eldercare Sensor Data

Omar A Ibrahim

Electrical Engineering and Computer Science Department

University of Missouri-Columbia

Columbia, Missouri, USA

oai9bc@mail.missouri.edu

Mihail Popescu

Health Management and Informatics Department

University of Missouri-Columbia

Columbia, Missouri, USA

popescum@health.missouri.edu

James Keller

Electrical Engineering and Computer Science Department

University of Missouri-Columbia

Columbia, Missouri, USA

KellerJ@missouri.edu

Abstract— Automatic health monitoring systems are a candidate solution to preventing health issues in independently living older adults by detecting and reporting signs of early illness. Abnormal sensor patterns produced by certain

resident behaviors could be linked to early signs of illness. In this paper, we propose a framework for detecting health patterns based on non-wearable sensor sequence similarity and natural language processing (NLP). In TigerPlace, an eldercare facility that promotes aging-in-place in Columbia, MO, we deployed 47 sensor networks along with a nursing electronic health record (EHR) system. The proposed framework utilizes sensor sequence similarity and medical concepts extracted from the EHR to automatically inform the nursing staff when health problems are detected. A context preserving representation of daily activities is used to measure the similarity between the sensor sequences of different days. The medical concepts are extracted from the nursing notes using MetamapLite, an NLP tool included in the Unified Medical Language System (UMLS). The proposed idea is validated on two pilot datasets from twelve Tiger Place residents, with a total of 5810 sensor days out of which 1966 with nursing notes.

Keywords— Eldercare monitoring; Wireless sensor networks; Early illness recognition; NLP; Sensor annotation.

10.1 Introduction

In 1996, the American Academy of Nursing asked researchers to find new ways to modify and improve the standards of eldercare services [1]. The rapid aging of the society gained extra attention from the healthcare industry in developed countries. Published statistics show that elderly population of age 65 and older will increase from 13% in 2010 to 19% in 2030 whereas the ratio between the population of age 15 to 65 (working age) and the elderly population is decreasing from 4.3 to 2.3 [2]. On one hand, elderly favor living independently despite complex conditions such as dementia, the risk of falling and frailty.

Independent living could potentially lead to poor long-term health outcomes due to delayed health assessments which might be due to lack of monitoring [3]. The late health assessment is an aggravating risk factor that typically happens due to the fear of being institutionalized and the lack of physician's assessments [4]. Automatic health monitoring systems are a candidate solution to managing health issues in independent living of older adults by detecting and reporting signs of early illness.

In the last decades, ubiquitous sensor networks have been deployed as a possible solution to improving quality and reducing the cost of eldercare by monitoring the decline in their functional abilities as well as improving the management of housing, healthcare, and social services [4, 5]. MIT's PlaceLab, Georgia Tech's Aware House, Honeywell's Independent Lifestyle Assistant, and University of Missouri's TigerPlace are examples of monitoring environments [5]–[8]. Fig. 10-1 shows the trajectory of typical functional decline in older adults (solid line) [9]. The typical decline has quasi-plateaus followed by sharp step-downs due to acute events such as falls that lead to loss of some functional ability such as dressing, walking, etc. Some of the step-downs are temporary (that is why we used the term "quasi-plateau" above) such as the ability to walk after having a leg injury, before they become permanent. We believe that using sensor-based health assessment and early recognition and/or prediction of health problems we can reduce the functional decline (dotted line curve) and improve the quality of life as in Fig. 10-1 (figuratively denoted as "squaring the life curve").

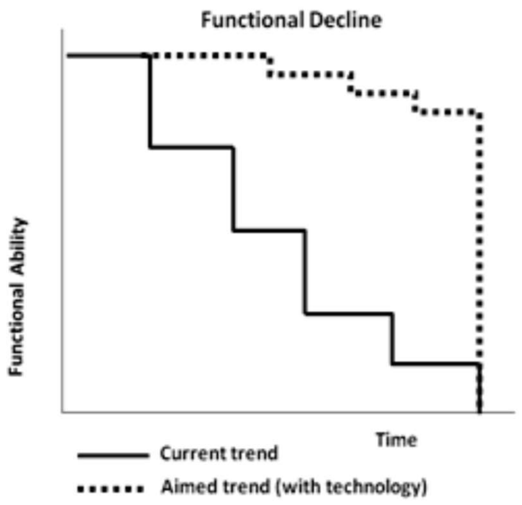


Figure 10-1 Trajectory of typical functional decline the goal with early illness recognition.

A variety of activity recognition (AR) approaches and medication compliance methods have been introduced in the literature [9]–[12]. Some AR methodologies are either detecting a specific set of known activities such as walking or falling [13] or looking for outliers such as too many bathroom visits [14]. Major differences in the proposed AR approaches are due to the machine learning models, the sensing technology, and the experimental setup [15, 16]. Despite these differences, most of AR methods are performed on known or scripted sequences of activities. One of the strengths of this paper is that it uses data collected in a real living environment for a relatively long period of time (5810 days) where the activities are not known and not scripted, hence requiring an unsupervised learning approach.

An important property of our health-monitoring system is the ability to unobtrusively and continuously collect activity data. The system processes the collected sensor data, detects activities such as taking a shower, “bathroom visit” or “out of the apartment,” and looks for changes in the behavior of the monitored older adults. Behavior changes could result from early signs of an imminent illness or an exacerbation of an existing chronic condition. The relationship between health patterns and behavior variations is based on the assumption that, for a specific resident, similar medical conditions lead to similar abnormal behaviors which result in similar sensor patterns. Hence, if a sensor pattern is not similar to the previous one in a similar context (location, time of the day), it is assumed that the patterns are generated due to some unknown health problem. These assumptions, in general, have a certain degree of validity for elderly whereas they might not generally be true for a more mobile, younger people [14, 17, 18]. Various sensors such as motion, depth camera, sound, radar, etc. have been deployed to capture people's behaviors [8]. In eldercare applications, the sensors are mounted in the living environment to capture the resident's behavior. These sensors produce multi-attribute time series datasets (MATS). Computing the similarity between these MATS is the basis for assessing the similarity of behavior patterns based on sensor data. Calculating MATS similarity is a challenging task which depends on the application and the type of the attribute (continuous or discrete). The first approach is the Euclidean-based distance measure for continuous MATS with equal length [20]. The drawback of this measure is that it is sensitive to outliers, cannot capture time and is not applicable for sequences that represent the same behavior but have different length (i.e., same action but performed slower). The second common approach is a non-Euclidean metric such as dynamic time warping (DTW) and longest common subsequence (LCSS) which are applied on sequences of different lengths [21]-[25]. Also, Symbolic Aggregate Approximation (SAX) can be applied for MATS dimensionality reduction [26]. If an entire MATS is converted to a series of symbols (symbolic data) similar to protein sequences in bioinformatics, then Smith-Waterman (SW) is another possible similarity measure that can be applied [27]. However, measuring the similarity of mixed sequences (some attributes are discrete such as motion sensor hits and others are continuous such as heart rate values) in eldercare mentoring systems is a challenging task that is still an open research question. Discrete data measures such as SW and LCSS [27] can be applied to continuous sequences by modifying them [20] or changing all the discrete sequences to continuous and applying the modified measures [14, 24, 28, 29]. In our previous work [30], we introduced an unsupervised approach to segment different daily activities. Also, we developed a context-preserving technique to represent daily activities of older adults such that it can be used to compute the similarity between different days of a specific resident [31].

To predict early illness, the sensor data can be combined with medical concepts extracted from nursing notes by employing a Natural Language Processing (NLP) method [32]. There are several challenges in this research that we would like to address. Identifying the similarity measure is the first challenge, with the second being the use of the Unified Medical Language System (UMLS) concepts to extract meaningful semantic types and medical and behavioral terms. In [32], all the terms were extracted without type constraints which resulted in, among other, trivial terms such as "the", "a", "an", which do not convey meaningful information for behavior inference. In this paper, we use the measure in [31] to determine the similarity between sensor sequences. To extract only medical and behavior related terms from the nursing notes, we parse each note and the ULMS concepts using semantic type constraints. Finally, we employ an aggregation technique to infer the most probable health concepts associated to an unknown sensor sequence.

There are two main aspects that distinguish our work from other related ideas: the dataset and the unsupervised machine learning methodology. The dataset used in this paper consists of real-world living environments with no prescripted scenarios. Elderly adults live in their personal apartments where they conduct their daily routines naturally without any constraints from researchers. Therefore, our dataset reflects the complexity of unrestricted real-world scenario and it allows for testing of the robustness of our activity recognition method. The second aspect is the machine learning approach to measure the similarity for discrete time series in eldercare applications. Our unsupervised approach discovers the most frequent activities (as sensor sequences) of daily living without knowing their identity. We combine all these sensor sequences (activities) that a resident performed in a specific day and we generate a single vector representing that day. We test our approach on two datasets from Tiger Place: one with three residents and another one with 9 residents, each resident having around two years worth of data.

This paper is organized as follows. In section 8.2, we describe the monitoring system architecture and available sensor data. The proposed method of illness prediction based on UMLS medical concepts and sensor data similarity is presented in section 8.3. The dataset description is presented in section 8.4. The evaluation metrics used in the experiment is discussed in section 8.5. In section 8.6, we show extensive experiments to validate our approach. Finally, in the last section, we provide conclusions and future work.

10.2 System Architecture

TigerPlace is an aging-in-place facility in Columbia, MO, USA, where, with the University of Missouri IRB approval, we implemented our integrated monitoring system. Only no wearable sensors are used for the monitoring process based on the recommendations from multiple focus groups in 2004 with elderly residents of TigerPlace [8, 33, 34]. The monitoring started in the fall of 2005. Currently, we have 50 apartments online where sensors are installed and data is collected. There are two datasets used in this paper. The first dataset is for three residents from 2005 to around 2007 with the old data acquisition system as in Fig. 10-2. The second dataset was obtained from 9 residents during 2015 using our new (updated) data acquisition system (see Fig. 10-3) where an alert manager, depth sensors (Kinect) and updated (hydraulic) bed sensors were added. The main components of the monitoring system are a sensor network, a data logger, a reasoning system for decline detection and recognition, an electronic health record (EHR) system, an alert manager to inform clinicians of possible problems, and a secure Web-based interface to display the data for clinicians and researchers.

Various motion passive infrared (PIR) sensors are mounted in different locations of a TigerPlace apartment: bathroom, living room, den, bedroom, and kitchen. The PIRs and the pneumatic bed sensor send X10 signals that are logged together with a time stamp in our sensor database as shown in Fig. 10-2. All the residents who are part of the study have logger computers in their apartments that capture the data from each wireless sensor network. Then, the data from these computers are sent to the main secure storage server via a secure wired network connection. Fig. 10-4 shows a floor map of a typical TigerPlace apartment with the locations of some of the sensors.

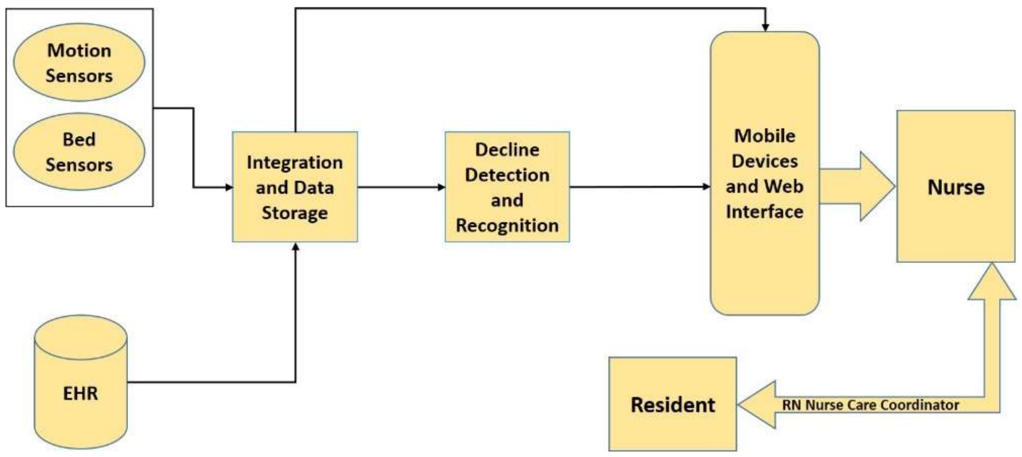

Motion Sensors
Bed Sensors
Integration and Data Storage
Decline Detection and Recognition
Mobile Devices and Web Interface
Nurse
EHR
Resident
RN Nurse Care Coordinator

Figure 10-2 Old TigerPlace sensor network architecture.

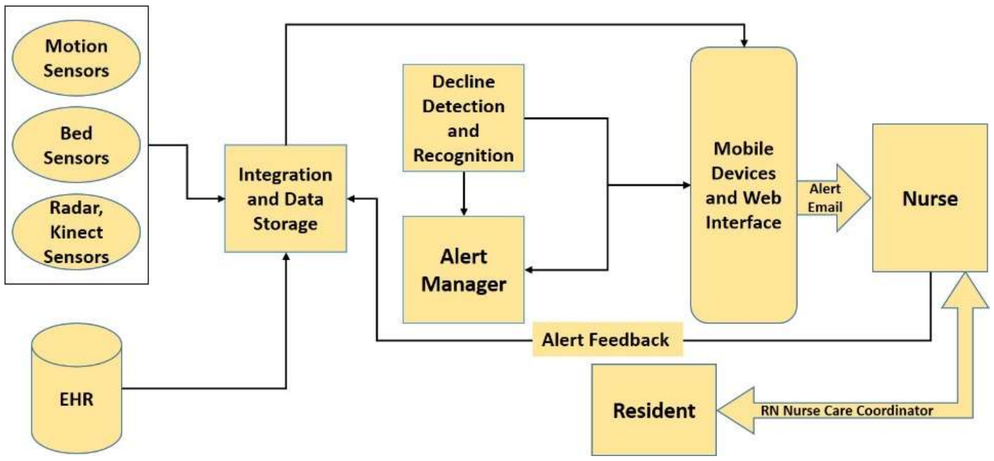

Motion Sensors
Bed Sensors
Radar, Kinect Sensors
EHR
Integration and Data Storage
Decline Detection and Recognition
Alert Manager
Mobile Devices and Web Interface
Alert Email
Nurse
Alert Feedback
Resident
RN Nurse Care Coordinator

Figure 10-3 Updated TigerPlace sensor network architecture.

The sensors used in this paper along with their identifier are shown in Table 10-1. The X10 motion sensors capture the residents' activity in their environment using PIR sensing (ID 1-10). The bed sensor is used to measure bed movements (ID 11–14), breathing (ID 15–17), and pulse (ID 18–20). The bed sensor uses pneumatic pressure to capture movement (restlessness), respiration, and the resident's ballistocardiogram as the mechanical effect of the heartbeat [23]. On the old system in Fig. 10-2, the discretization of the bed sensor is performed in hardware by the bed sensor system as described in [11]. However, on the new bed sensors showed in Fig. 10-3, the hydraulic bed sensor provides continuous signals for all three previously mentioned quantities without artificial quantization [9].

Table 10-1 Sensor identifiers used in this paper

Sensor ID	Sensor Type	Sensor ID	Sensor Type
1	Front-door Motion	11	Bed Movement 1
2	Living-room Motion	12	Bed Movement 2
3	Closet Motion	13	Bed Movement 3
4	Kitchen Motion	14	Bed Movement 4
5	Patio Motion	15	Breathing 1 (low)
6	Shower Motion	16	Breathing 2 (ok)
7	Bedroom Motion	17	Breathing 3 (high)
8	Bathroom Motion	18	Pulse 1 (low)
9	Office Motion	19	Pulse 2 (fine)
10	Den Motion	20	Pulse 3 (high)

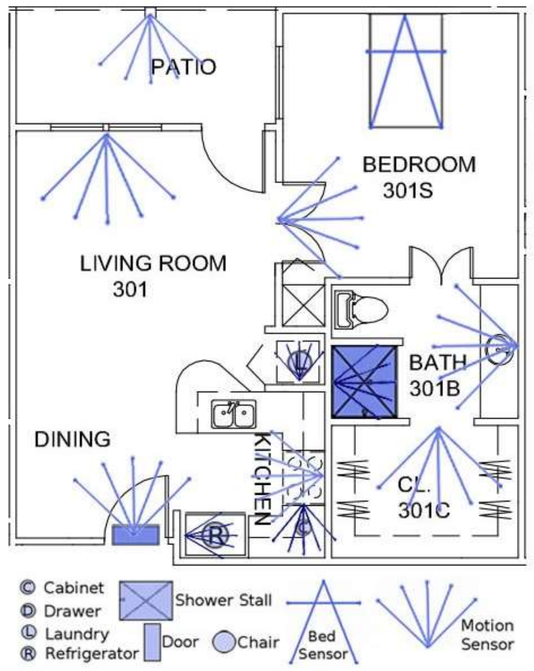


Figure 10-4 Typical apartment layout in TigerPlace.

Table 10-2 shows a snippet of the sensor firing data recorded in the log file for a resident of TigerPlace on November 19, 2006, around 08:0 pm where the person had most of the sensors firings in the living room and the office. Sensor ids 2, 4 and 9 are motion sensors in the living room, kitchen, and office respectively. Note that the data in Table 10-2 is a sensor sequence of length 10.

Table 10-2 Snippet of the log file of a resident in TigerPlace

Year	Month	Day	Hour	Minute	Second	Sensor ID	Year
2006	11	19	19	7	56	2	2006
2006	11	19	19	8	3	2	2006
2006	11	19	19	8	10	2	2006
2006	11	19	20	18	50	2	2006
2006	11	19	20	18	53	9	2006
2006	11	19	20	18	56	4	2006
2006	11	19	20	19	4	9	2006
2006	11	19	20	31	35	9	2006
2006	11	19	20	32	26	9	2006
2006	11	19	20	32	41	9	2006

10.3 Method

Our goal is to evaluate the similarity between resident's days based on the sensor data, and to use that information for annotation. After finding the set of most similar days $\{D_i\}$, to a target day T, we transfer the set of UMLS concepts $\{C_i\}$ associated with $\{D_i\}$ to T. These concepts, $\{C_i\}$, can then be used to describe the behavior of the resident on day T and, possibly, predict his or her health status. For each day, the sensor firings in Table 10-1 are converted into subsequences of discrete symbols where each symbol represents one sensor type. The idea is to represent the activity (behavior) of a resident by a sequence of sensor firings as it would be represented by his/her genomic sequences. The daily sensor sequence is split into subsequences using a separation threshold of 30 seconds, which provides enough granularity to capture daily activities [31]. Too small of a separation threshold would break the activities into smaller sub-activities, which is not of our interest. Too large of a separation threshold results in merging different activities into a single activity, which is also not desired for our approach. After investigating different threshold values, we concluded that a 30 seconds threshold worked best for our problem. In [31], we developed a measure to find the similarity between a resident's days using his/her sensor data and showed that it produces better results than a string based measure (Smith-Waterman).

We will explain the process of generating the sequence features through an example. In Table 10-3, we display a snippet of the motion sensor firing data recorded in the log file for a resident of TigerPlace on September 27, 2005, around 01:45 pm where the person had most of the sensors firings in the living room. Sensor ids 2, 7 and 8 are motion sensors in the living room, bedroom, and bathroom respectively. The data in Table 10-3 is a sensor sequence of length 16 and when using our 30 second separation threshold to divide the sequence into subsequences of activities, we get three subsequences of different lengths. The resulting subsequences are ('LLLL', 'LDT', and 'DLLLLLL') where L, D and T are symbols annotation of sensors ids 2, 7 and 8 respectively. The sequence 'LLLL' represents a living room activity, 'LDT' is a bathroom visit, which is a walk from living room to bathroom going through the bedroom, and 'DLLLLLL' represents a living room activity after finishing the bathroom visit. In our experiment, sequences of length 2 or less and those of length 50 or more are pruned from each day because they are either not long enough to represent an activity, or very long due to nurses, staff or other visitors. We mention that having a depth camera in each apartment allowed us to inspect some unclear aspects of our dataset.

Employing a bag-of-words approach, each sequence is mapped into an M-Dimensional Euclidean space representing the percentage of each symbol in a sequence, where M represents the number of sensors in a particular apartment. To map a sequence of a specific resident to the M-Dimensional space, each dimension is assigned to one symbol (sensor) and the sum of the similar symbols are divided by the length of the sequence in each of the M features. For example, assuming that the sensor positions in the feature vector are [L, F, D, T, P], the sequences ('LLLL', 'LDT', and ‘DLLLLLL’) are mapped to [1 0 0 0 0], [1/3 0 1/3 1/3 0], and [6/7 0 0 1/7 0]) assuming that there are five sensors in our system. Using this representation, all sensor sequences will have the same length, M.

Table 10-3 A sensor sequence snippet for a TigerPlace resident

Year	Month	Day	Hour	Minute	Second	Sensor ID	Sequences
2005	9	27	13	44	9	2	Sequence 1
2005	9	27	13	44	15	2	
2005	9	27	13	44	22	2	
2005	9	27	13	44	32	2	
2005	9	27	13	52	35	2	Sequence 2
2005	9	27	13	52	40	7	
2005	9	27	13	52	45	8	
2005	9	27	13	54	51	8	Not a valid sequence
2005	9	27	13	54	54	7	
2005	9	27	13	55	30	7	Sequence 3
2005	9	27	13	55	36	2	
2005	9	27	13	55	46	2	
2005	9	27	13	55	53	2	
2005	9	27	13	56	1	2	
2005	9	27	13	56	9	2	
2005	9	27	13	56	16	2	

To explain the process of generating a single data point representation for a day using the sequences in the M-Dimensional space, we extend the example from Table 10-3 by considering three sequences in the same day (see Table 10-4). First, we generate the histogram for each feature using 5 bins, that is (0 - 0.2, 0.2 - 0.4, 0.4 - 0.6, 0.6 - 0.8, and 0.8 - 1). Then, we concatenate the histograms of all features to generate a single feature vector representing the entire day. Note that using this representation, we preserve the activity context since the histograms are based on the sequences (activities) performed that day.

Table 10-4 Example explaining the process of converting sequences of a day into a single data point

Selected Day Numeric Data Representation				
Feature 1 (L)	Feature 2 (F)	Feature 3 (T)	Feature 4 (D)	Feature 5 (P)
1	0	0	0	0
1/3	0	1/3	1/3	0
6/7	0	0	1/7	0
Histogram 1	Histogram 2	Histogram 3	Histogram 4	Histogram 5
0 1 0 0 2	0 0 0 0 0	0 1 0 0 0	1 1 0 0 0	0 0 0 0 0
Concatenated Histograms to Get a Single Data Point Representation				
0 1 0 0 20 0 0 0 00 1 0 0 01 1 0 0 00 0 0 0 0				

The complete sequence annotation system (SAS) is shown Fig. 10-5. Our goal is to annotate a day (target day) by using all related UMLS terms from the most similar days in the database. The SAS system processes the sensor sequence of each day to find a single vector representation and uses it to detect the most similar sensor vectors to this pattern, and selects the top N. Then, SAS extracts the nursing notes from the days associated with the sensor representations selected by our measure. For these notes, SAS applies MetaMapLite (https://metamap.nlm.nih.gov/MetaMapLite.shtml) from UMLS to convert raw text (nursing notes) to a set of semantic types and concepts unique identifiers of the medical (CUI) terms. MetaMapLite is an NLP software made available by National Library of Medicine as part of the UMLS tools. As opposed to generic NLP tools, MetaMapLite uses a medical ontology (UMLS) and it is aware of problems specific to medical domain such as medical abbreviations, medical noun phrases and medical negation. Finally, SAS suggests the possible problem that generate the behavior of the target day based on the annotated CUI terms and their semantic types. For example, if an alarm is generated in the target day, the email message to the caregiver could read: “Abnormal patient sleep pattern, possibly due to pain”. SAS uses all the available sensor history for each resident such that the search is conducted among the days with and without nursing notes. If no nursing note is found, we add a term ‘normal’ for days where there is no comment.

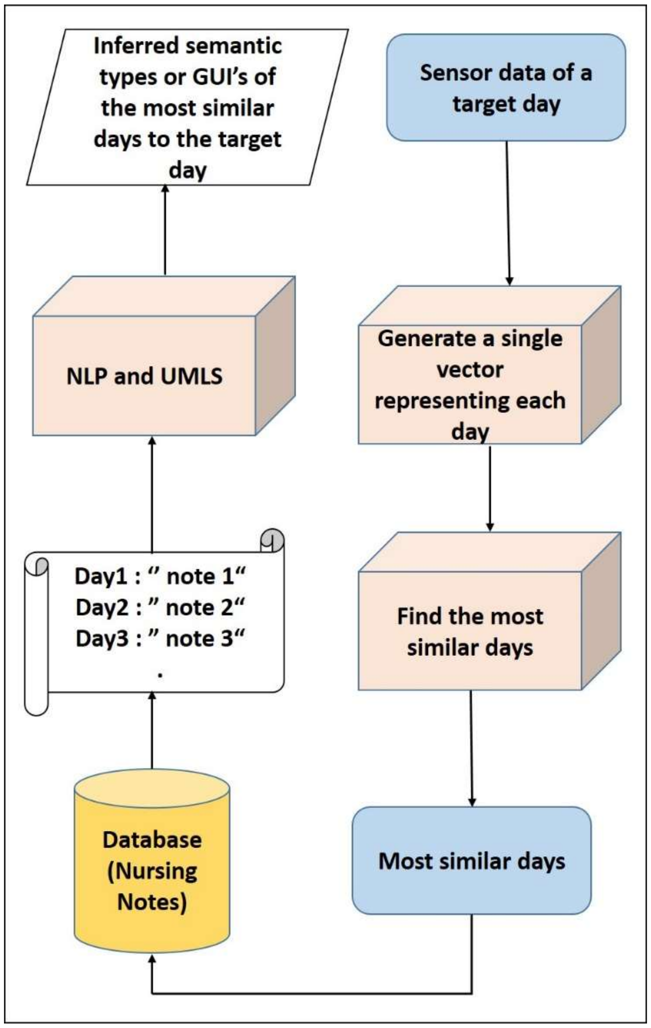

Inferred semantic types or GUI’s of the most similar days to the target day
Sensor data of a target day
NLP and UMLS
Generate a single vector representing each day
Day1 : ” note 1“
Day2 : ” note 2“
Day3 : ” note 3“
.
Find the most similar days
Database (Nursing Notes)
Most similar days

Figure 10-5 Sensor data annotation system (SAS) diagram.

As MetaMapLite extracts the UMLS terms from the nursing notes, it may display duplicate terms due to their membership in different semantic classes. Hence SAS has to first, filter out the duplicate terms, and second, if a CUI term belongs to multiple semantic types then we use a priority of semantic types to make sure it belongs to only one that is the most related to our desired category. The reason behind the second decision is that we would like to have more symptoms or behaviors than clinical attributes. Out of the 128 UMLS semantic types, we only selected 32 that we believed could be linked to monitoring elderly. The selected UMLS semantic types together with their UMLS abbreviations We focus on 'sosy: Sign or Symptom', 'dsyn: Disease or Syndrome', 'acty: Activity', 'menp: Mental Process','bdsu: Body Substance', and 'bhvr: Behavior' as the highest priority. For example, 'sosy' and 'dysn' denote a disease, while 'acty', 'bhvr', 'menp' and 'blor' can easily show the physical and mental activity of the resident.

Table 10-5 Selected ULMS semantic types

Abbreviation	Full Name	Abbreviation	Full Name
sosy	Sign or Symptom	dsyn	Disease or Syndrome
acty	Activity	menp	Mental Process
bdsu	Body Substance	bhvr	Behavior
blor	Body Location or Region	anst	Anatomical Structure
antb	Antibiotic	bacs	Biologically Active Substance
bdsy	Body System	biof	Biologic Function
bpoc	Body Part, Organ, or Organ Component	bsoj	Body Space or Junction
clna	Clinical Attribute	clnd	Clinical Drug
diap	Diagnostic Procedure	dora	Daily or Recreational Activity
edac	Educational Activity	evnt	Event
fndg	Finding	hlca	Health Care Activity
inbe	Individual Behavior	lbtr	Laboratory or Test Result
medd	Medical Device	mobd	Mental or Behavioral Dysfunction
npop	Natural Phenomenon or Process	orch	Organic Chemical
orgf	Organism Function	patf	Pathologic Function
phsf	Physiologic Function	phsu	Pharmacologic Substance
socb	Social Behavior	top	Therapeutic or Preventive Procedure
Normal	Normal behavior		

10.4 Dataset

The pilot sensor datasets used in the paper are shown in Table 10-6 (the data from the old system) and Table 10-7 (the data from the new system). There is a total of 5810 sensor days with more than 5 million sensor hits, collected from twelve residents of TigerPlace. The content of the nursing notes is about various physical, social and mental health complaints included by the onsite nursing personnel in the resident's EHR. From Tables 10-6 and 10-7, we see that there are fewer days with notes than the total number of days because, for some days, there were no nursing comments. On the other hand, some residents may have multiple comments per day depending on their health conditions. In this case, we merged all extracted terms for that day.

Table 10-6 Dataset of the residents from our old system in Fig. 10-2

Resident ID	**1**	**2**	**3**
Total # Days	441	744	498
# Days with Notes	38	21	135

Table 10-7 Dataset of the residents from our new system in Fig. 10-3

Resident ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Total # Days	599	164	478	349	592	406	423	533	583
# Days with Notes	30	20	120	137	392	268	133	418	254

10.5 Evaluation Metrics and Experiment Setup

Our goal is to detect days (events) in which the residents need to be assessed (referred to as an abnormal day). For each resident, we build a machine learning model because disease behavior is a subjective matter that differs from resident to resident. For instance, for some residents, several nighttime bathroom visits are considered normal whereas multiple bathroom visits during the night for some residents who have more sound sleep patterns during the night are considered as an abnormal event. We used the extracted terms from the nurses' notes, and we impute the term normal for all days with no notes found. We performed two types of experiments: one in which we try to predict the semantic type of the annotation (e.g., is this a symptom or a medication change?) and another in which we try to predict the exact UMLS terms (e.g., pain, Lisinopril). Then, we compute the precision and recall using (1) and (2). The following evaluation metrics are applied in our experiments:

$$Precision = \frac{t_p}{t_p + f_p} \quad (1)$$

$$\mathrm{Re}call = \frac{t_p}{t_p + f_n} \quad (2)$$

where, t_p is the number of true positives, t_n is the number of true negatives, f_p is the number of false positives, and f_n is the number of false negatives.

To explain the process of computing the precision and recall from the semantic type or the terms, we present an example. Assume day 1 has the following terms: Day1= {T7, T9, T17, T17, **T22**, **T33**, **T33**, and day X (or a group of days where their terms concatenated together) has the following terms: Day X= {T18, **T22**, T26, T26, T28, **T33**, **T33**, T34, T35, T35, T35, T35, T35}. Using Table 10-8 below and equation 1 and equation 2, we get the following:

Precision: 3/ (3+10) =3/13

Recall: 3/ (3+4) =3/7

Table 10-8. f_p, t_n, and f_n calculation

	10(FP)
4(FN)	3(TP)

10.6 Experimental Results

To investigate the performance of the proposed sensor data annotation methodology (SAS), we use the two TigerPlace datasets shown in Table 10-6 and Table 10-7. We mention that the fact that the dataset are quite different shows that our methodology is robust. We compare the results to the approach in [35] that uses a 48-dimensional vector to represent the sensor sequence of a day. The first 24 dimensions are the hourly accumulated motion sensor hits and the other 24 dimensions are the hourly sum of the restlessness (in seconds) from the bed sensor. We conducted a separate experiment on each resident to test the performance of our algorithms for annotating a day with UMLS concepts from the nursing notes. The reason behind resident-based experiments is that, typically, mathematical models (such as classifiers, algorithm parameters, etc.) for behavior recognition based on non-wearable sensors are not transferrable from one person to another, due to different disease-behavior associations that could be quite different between individuals.

To shed more light into our methodology, we examine some case studies. The first case is from resident 8 in Table 10-7. On 12/18/2015 the resident has the following nursing note: "*The resident wasn't feeling well last night and complaining of SOB. He/she did not want to go to the ER last night. He/she is feeling better this morning. He/she received some guidelines from cardiologist's nurse regarding when to go to the ER if necessary: 1) HR 120+, 2) SOB, and 3) chest pain. His/her cardiology appointment is Wed. Feels he/she went into A. Fib. and recovered. Resident has nitro tabs; will take if experiencing chest pain. I assisted him/her with putting on compression hose, pants and socks/shoes*". After parsing the note, we get the following semantic types and CUI's: "menp:Feeling; sosy:Chest Pain; clna:Chest pain; sosy:SOB;". The most similar day has the following comment: "*Resident just came back from VA ER following assessment for fluid in legs, especially thighs, weakness, joint pain. ER told the resident he/she is having SE of Tamiflu he took for 2 days. ER told him he/she would recover better at home due to so many people with flu, pneumonia, etc in hospital and sent home with Tylenol and Guifenasin. Assisted to BR and back to bed.*

Started to undress. Nurse came in to finish undressing. Had only gotten his/her coat off and shirt off. Very slow and weak." The semantic types and the CUI's are: "bpoc:Legs; sosy:weak; bdsu:FLUID; sosy:joint pain; sosy:weakness; acty:assessment; blor:back; blor:thighs; dsyn:flu; bpoc:coat". We can see that both notes convey a medical situation where the resident needs to be assessed. Therefore, inferring sosy (Sign or Symptom), dsyn (Disease or Syndrome) in general or pain in particular, would be a good annotation for the 12/18/2015 day.

The second case is from resident 1 in Table 10-7 on 01/11/2014 where he/she had the following nurse comment: "*came to nurse station wanting something for cough. said didn't sleep well, slept in chair because every time he/she laid down, he/she would have a coughing fit. I called Nurse and she said he/she can go to urgent care if he/she likes to make sure he/she doesn't have pneumonia*". The parsed semantic types and the extracted terms are: {acty:Fit; sosy:fit; dsyn:Pneumonia; orgf:Sleep; sosy:Coughing; sosy:Cough}. The most similar days have the following notes along with their semantic types and ULMS terms: 1) "*Pt states he/she is feeling better. Lungs CTA. Productive cough, sputum clear. Afebrile*" and {sosy:Productive cough; fndg:Clear;} 2) "*Pt states he/she has had a nasal cold for approx 1 week. Lungs CTA. Afebrile. Denies SOB. Pt may take Guafenesin for drainage. Instructed pt to return to clinic if worsens.*" and {bdsu:Drainage; bpoc:Nasal; dsyn:Cold; bdsu:drainage; fndg:worsens}; 3) "*having a temp during night and cough worse. I requested nurse make SNV to assess him/her. He/she is up and dressed. States she/he had to sleep on three pillows last night to keep from coughing and had temperature. States had several coughing episodes that just couldn't stop coughing. L lung clear but has rhonchi and slight wheeze throughout R lung fields today which had been clear. SN called Dr. X office to try* *to get an appointment with Someone there today*" and {fndg:worse; sosy:wheeze; topp:TEMP; acty:stop; bpoc:lung; blor:back; sosy:coughing; sosy:cough; fndg:clear}. It is clear that all the most similar days have similar note content about cough and that can be inferred from both semantic types (sosy and dsyn) or from the CUI terms ("cough" and "cold").

The third case is from resident 6 in Table 10-7 on 09/25/2014 with the following note, semantic types and the CUI terms: "*Pt states the resident fell outside church on sidewalk, face first 2 days ago. States he/she is very sore in R rib area and it hurts when he coughs. States it is not difficult to breathe. Vision clear and unchanged. Denies headache. Very small bruise on R mid rib area. Small scabs present on chin, lip and R forehead. Alert and oriented. Walking as usual. Took Ibuprofen this am with min results. Appt set up today with Dr. X at Y Clinic at 2:20pm for assessment. Pt will ride TP van. DON notified*." and extracted terms {sosy:headache; fndg:fall; fndg:church; acty:assessment; sosy:sore; blor:chin; bpoc:rib; fndg:bruise; fndg:clear; bpoc:lip; blor:forehead; fndg:alert; fndg:difficult; clna:alert; sosy:sore; fndg:present; blor:face; fndg:green; fndg:unchanged; fndg:very small}. The most similar day has the following note along with the semantic types and the CUI terms: "*resident came by office following fall yesterday while out to Mass for wife at X Center. Has scab over several small lacerations, one on R side of eye temple area, one on R side of lower lip and one on R side of nose*." and extracted terms {bpoc:eye; sosy:mass; fndg:center; blor:temple; bpoc:nose; blor:temple; fndg:fall; bpoc:lower lip}. In this case we can see that both days are related to the resident condition after a fall which can be inferred to the target day. Fall detection is one of our top priority alerts in our monitoring system, as many falls are not reported or detected, so when one of the inferred CUI terms is "fall", an alert could be generated to monitor the resident's activity in more detail.

After presenting three case studies that explain our approach, we run the proposed system on each resident in Table 10-6 and Table 10-7. For each day in each column in Table 10-6 and Table 10-7, we find the K most similar days. We compare the semantic types and the CUI terms of the target day with the sematic types and the CUI's of the K most similar days by measuring the precision and recall (PR) values using formulas (1) and (2). After computing the PR values for all the days, we find the average values which represents one point in the PR curve. Then, we vary K and find the average PR value for all possible K values to generate the graphs in Fig. 10-6. High precision means you found most of the semantic types (terms) in the target day from the K nearest days with less false positives (requires fewer neighbors). High recall means you matched most of the semantic types of the target day (requires more neighbors).

The PR curves for the residents in Table 10-6 are shown in Fig. 10-6 where semantic types are used in computing the PR values. The PR curves of the resident in Table 10-7 are displayed in Fig. 10-7. We can see that as number of neighbors increases, the precision decreases and the recall increases because the true positive increases (matching more semantic types of the target day) and the false positive increases (inferring terms that do not match the target day's terms). Resident 3 in Fig. 10-6 has the lowest PR values compared to residents 1 and 2 because most of his data has consecutive days with nursing notes, which makes it harder to annotate them with high PR values. In addition, resident 3 uses a walker which reduces the variety between his routines in normal days and days where there are health-related issues.

Fig. 10-7 displays the results of the residents in Table 10-7. The youngest, and more active, resident has more variation among his/her daily routines, which makes it easier for our system to find a meaningful annotation. It can be noticed that our measure outperforms the measure in [35] in all the residents in Fig. 10-6 and Fig. 10-7. These results show a great improvement over the approach in [35] where each day was represented using the hourly sum of the firings of sensors hits during the entire day, and hence, where the identity of the daily activities was not preserved.

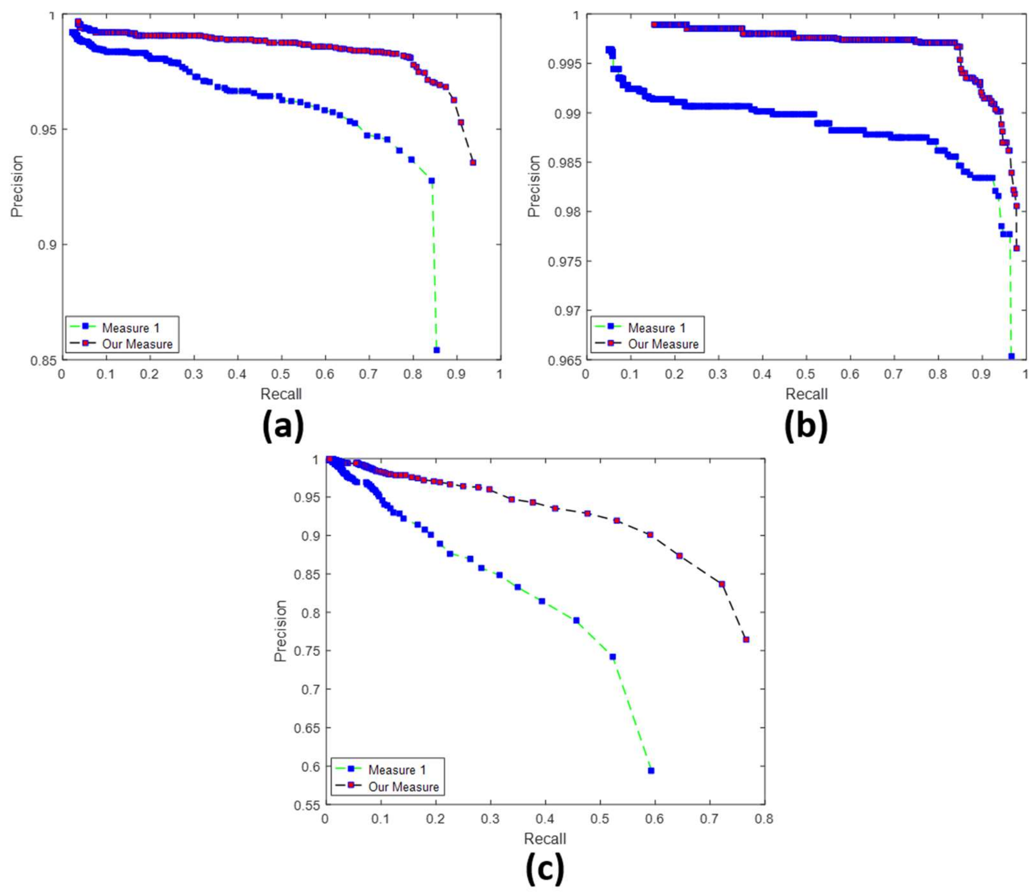


Figure 10-6 Precision-Recall curves using the semantic types of the residents in Table 10-6: a) resident 1, b) resident 2, and c) resident 3, where measure 1 is the measure in [35] and our measure.

In Fig. 10-6 and Fig. 10-7, the semantic types are used to measure the PR values. Using the CUI terms to annotate a day leads to more precise description. Therefore, we run the same experiment on some of the residents in Table 10-7. Fig. 10-8 shows the PR curves for residents 1, 2, 3, 4, 7, and 8. Comparing them with the curves in Fig. 10-7, we see a decrease in the PR values due to the variety of the CUIs as compared to the semantic types, which represents the main categories. If the resident notes include enough clinical information, then annotating with CUI terms makes more sense. However, for a resident with multiple health issues and a variety of less specific nursing notes, annotating with semantic types makes more sense.

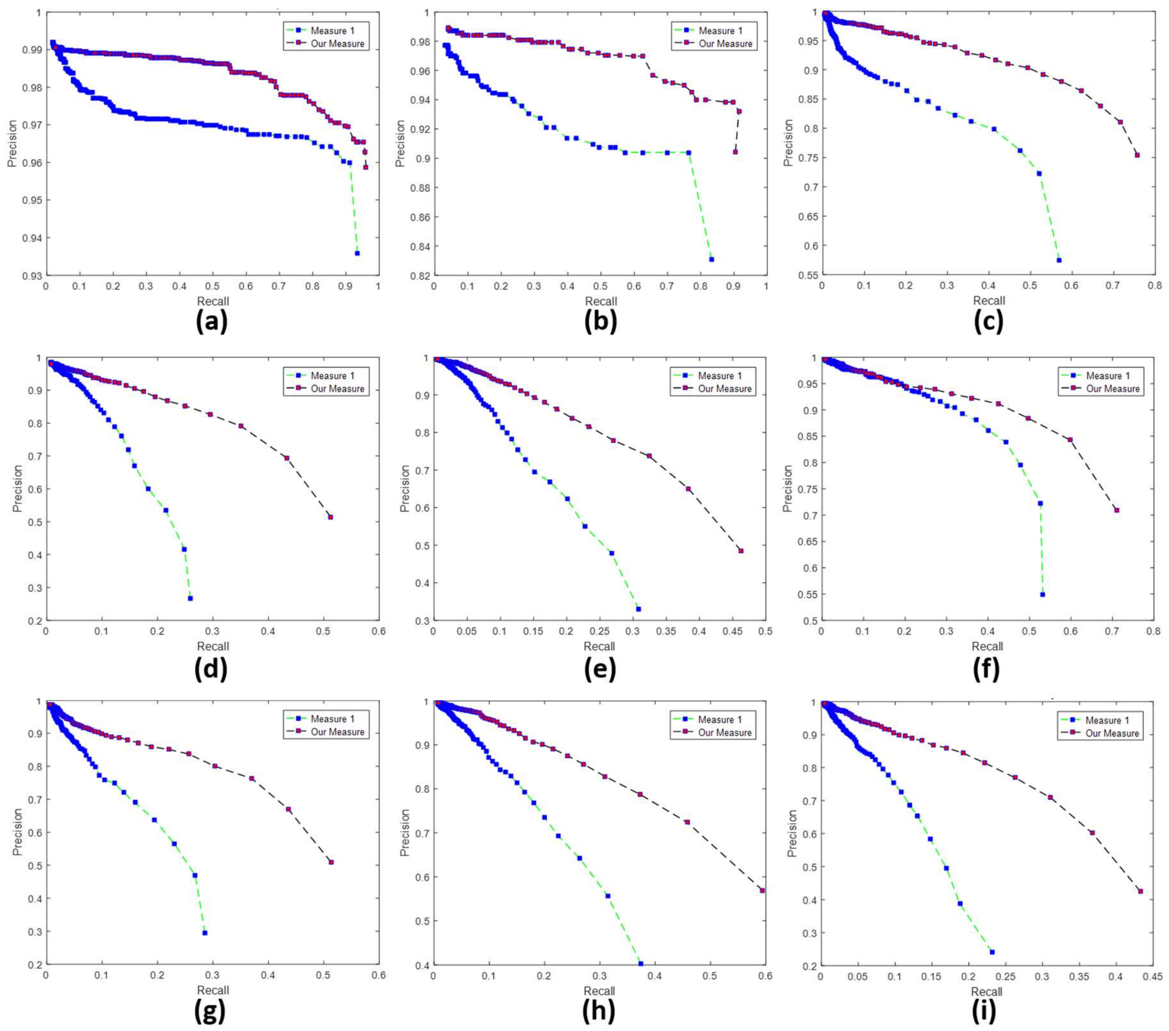


Figure 10-7 Precision-Recall curves using the semantic types of the residents in Table 10-7: a) resident 1, b) resident 2, c) resident 3, d) resident 4, e) resident 5, f) resident 6, g) resident 7, h) resident 8, and i) resident 9, where measure 1 is the measure in [35] and our measure is the measure.

10.7 Conclusion

In this paper, we presented a framework for health pattern (illness or activity) prediction using sensor networks in eldercare employing a context-preserving measure. The health patterns were described by UMLS concepts that were extracted from nursing notes using MetaMapLite, an NLP tool provided by National Library of Medicine. Our unique nursing EHR that captures both sensor and nursing clinical information for TigerPlace residents made possible the association of nursing notes to sensor sequences. We presented three case studies where our system annotated the days with ULMS concepts (semantic types and CUI terms) that convey the same information as the available nursing notes on those days. Comparing the system annotation with the available nursing notes shows the effectiveness of our system in informing clinical personnel of resident medical problems. Also, we tested our framework on two datasets that consisted, overall, of 5810 days of sensor data with 1966 days with nursing notes. We compared our results with an approach that uses the hourly accumulated motion sensor hits and the hourly sum of the restlessness (in seconds) from the bed sensor. PR curves were used as the evaluation metric where our measure outperforms an earlier measure because it preserves the context of daily routines. The proposed framework will be integrated into the existing alert system in TigerPlace to detect abnormal days by identifying early functional changes reflected in deviations from daily routines.

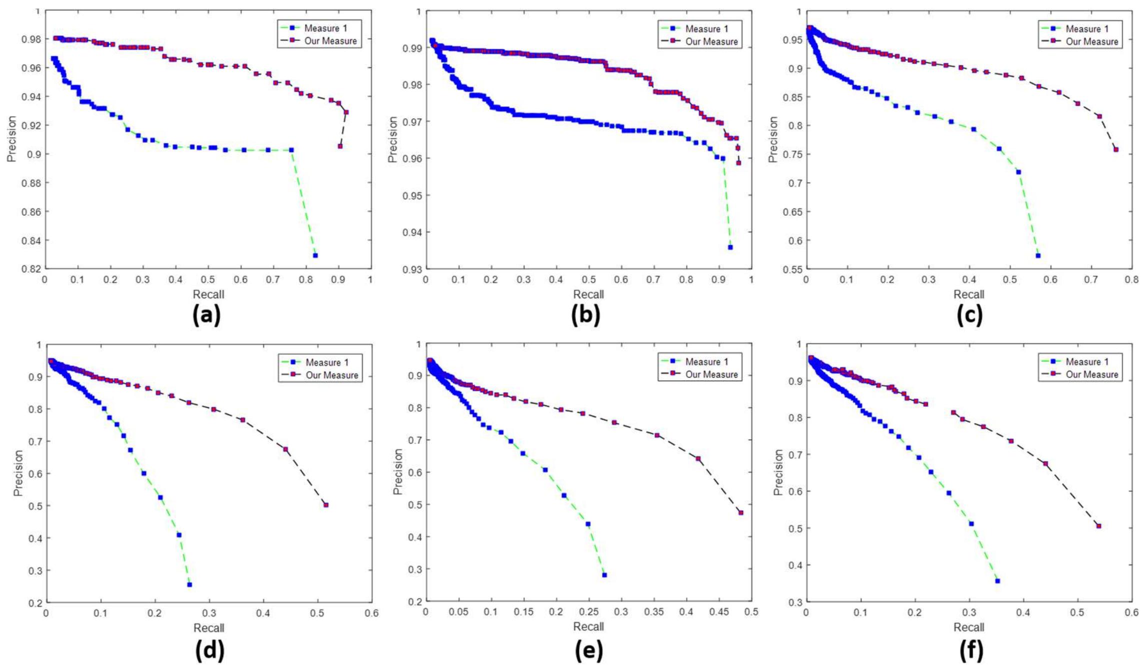


Figure 10-8. Precision-Recall curves using the CUI terms of the residents in Table 10-7: a) resident 2, b) resident 1, c) resident 3, d) resident 7, e) resident 4, and f) resident 8, where measure 1 is the measure in [35] and our measure is the measure

10.8 References

[1] M. Fergenson, “TigerPlace: An innovative ‘aging in place’ community,” Am. J. Nursing, vol. 113, no. 1, pp. 68–69, 2013.

[2] K. V. Grayson and A. V. Victoria, “The next four decades: the older population in the United States: 2010 to 2050,” U.S. Dept. Commerce, Economics and Statistics Admin., U.S. Census Bureau, Washington DC, USA, pp. 1–14, 2010.

[3] J. S. Goodwin et al., “Risk of continued institutionalization after hospitalization in older adults,” J. Gerontol. A. Biol. Sci. Med. Sci., vol. 66, pp. 1321–1327, 2011.

[4] T. L. Hayes et al., “An unobtrusive in-home monitoring system for detection of key motor changes preceding cognitive decline,” in Proc. 26th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc., pp. 2480–2483. 2004.

[5] S Intille et al., “Using a live-in laboratory for ubiquitous computing research,” in Proc. 4th Int. Conf. Pervasive Comput., pp. 349–365, 2006.

[6] C. D. Kidd et al., “The aware home: A living laboratory for ubiquitous computing research,” in Proc. 2nd Int. Workshop Cooperative Buildings Integrating Inf., Org., Archit., pp. 191–198, 1999.

[7] K. Z. Haigh, L. M. Kiff, and G. Ho, “Independent lifestyle assistant: Lessons learned,” Assistive Technol., vol. 18, pp. 87–106, 2006.

[8] M Skubic et al., “A smart home application to eldercare: Current status and lessons learned,” J. Technol. Health Care, vol. 17, pp. 183–201, 2009.

[9] M. J. Rantz et al., “A technology and nursing collaboration to help older adults age in place,” Nursing Outlook, vol. 53, pp. 40–45, 2005.

[10] D. Heise and M. Skubic, “Monitoring pulse and respiration with a noninvasive hydraulic bed sensor,” presented at the 32nd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc., Buenos Aires, Argentina, 2010.

[11] R. Beckwith, “Designing for ubiquity: The perception of privacy,” IEEE Pervasive Comput., vol. 2, no. 2, pp. 40–46, Apr–Jun. 2003.

[12] D. Mack et al., “A passive and portable system for monitoring heart rate and detecting sleep apnea and arousals: Preliminary validation,” in Proc. 1st Transdisciplinary Conf. Distrib. Diagnosis Home Healthcare, pp. 51–54, 2006.

[13] Y Li, KC Ho, and M Popescu "Efficient source separation algorithms for acoustic fall detection using a Microsoft Kinect." IEEE Transactions on Biomedical Engineering, vol. 61, no. 3, pp.745-755, 2014.

[14] M. Popescu and A. Mahnot, “Early illness recognition in older adults using in-home monitoring sensors and multiple instance learning,” Methods Inf. Med., vol. 51, no. 4, pp. 359–367, 2012.

[15] C. K. Narayanan and D. J. Cook, “Activity recognition on streaming sensor data,” Pervasive Mobile Comput., vol. 10, pp. 138–154, 2014.

[16] D. J. Cook et al., “Activity discovery and activity recognition: A new partnership,” IEEE Trans. Cybern., vol. 43, no. 3, pp. 820–828, Jun. 2013.

[17] M. Popescu and E. Florea, “Linking clinical events in elderly to in-home monitoring sensor data: A brief review and a pilot study on predicting pulse pressure,” J. Comput. Sci. Eng., vol. 2, pp. 180–199, 2008.

[18] Y. Lee, D. Ho, and M. Popescu, “Microphone array system for automatic fall detection,” IEEE Trans. Biomed. Eng., vol. 59, no. 5, pp. 1291–1301, May 2012.

[19] T. Kahveci, A. Singh, and A. Gurel, “Similarity searching for multiattribute sequences,” in Proc. 14th Int.Conf. Sci. Statist. DatabaseManag., pp. 175–184, 2002.

[20] G. A. H. Ten et al., “Multi-dimensional dynamic time warping for gesture recognition,” in Proc. 13th Conf. Adv. School Comput. Imaging, 2007.

[21] M. Vlachos,M. Hadjieleftheriou, D. Gunopulos, and E. Keogh, “Indexing multi-dimensional time-series with support for multi distance measures,” in Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, pp. 216–225, 2003.

[22] P. Sanguansat, “Multiple multidimensional sequence alignment using generalized dynamic time warping,” WSEAS Trans. Math., vol. 11, no. 8, pp. 684–694, 2012. 962 IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS, VOL. 20, NO. 3, MAY 2016

[23] H. Ding et al., “Querying and mining of time series data: Experimental comparison of representations and distance measures,” Proc. VLDB Endowment, vol. 1, no. 2, pp. 1542–1552, 2008.

[24] T. Rakthanmanon et al., “Searching and mining trillions of time series subsequences under dynamic time warping,” in Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, pp. 262–270, 2012.

[25] J. Lin et al., “Experiencing SAX: A novel symbolic representation of time series,” Data Mining Knowl. Discovery, vol. 15, no. 2, pp. 107–144, 2007.

[26] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences,” J. Molecular Biol., vol. 147, pp. 195–197, 1981.

[27] L. Bergroth, H. Hakonen, and T. Raita, “A survey of longest common subsequence algorithms,” in Proc. 7th Int. Symp. String Process. Inf. Retrieval, pp. 39–48, 2000.

[28] Z. Hajihashemi and M. Popescu, “Predicting health patterns using sensor sequence similarity and NLP,” in Proc. IEEE Int. Conf. Bioinformat. Biomed. Workshops, pp. 948–950, 2012.

[29] F. Duchene, C. Garbay, and V. Riale, “Learning recurrent behaviors from heterogeneous multivariate time-series,” Art. Intell. Med., vol. 39, pp. 25–47, 2007.

[30] O. A. Ibrahim, M Popescu, J Keller, " Unsupervised Analysis of Activity Patterns in Eldercare Monitoring," American Medical Informatics Association (AMIA) Annual Symposium Proceedings, 2017.

[31] O. A. Ibrahim, J. Keller and M. Popescu, "Context preserving representation of daily activities in elder care," 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Kansas City, MO, USA, pp. 547-551, 2017.

[32] Z Hajihashemi, M. Popescu, “An Early Illness Recognition Framework Using a Temporal Smith Waterman Algorithm and NLP,” AMIA Annual Symposium Proceedings, 548-557, 2013.

[33] G. Demiris et al., “Older adults’ attitudes towards and perceptions of ‘smarthome’ technologies: A pilot study,” Med. Inf. Internet Med., vol. 29, no. 2, pp. 87–94, Jun. 2004.

[34] G. Demiris, M. J. Rantz, M. A. Aud, K. D. Marek, H. W. Tyrer, M. Skubic, and A. A. Hussam, “Seniors’ attitudes towards home-based assistive technologies,” presented at the 29th Annu. MNRS Res. Conf., Cincinnati, OH, USA, Apr. 1–4, 2005.

[35] Z. Huang, “Sensor Sequence Annotation Using Medical Record Data,” MS project, Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, Dec, 2017.

Chapter 11: Conclusions and Future Work

Streaming clustering, directed by change detection, can identify structures in online data whereas traditional batch approaches fail. Two versions of streaming clustering algorithms (MUSC1 and MUSC2) are developed and tested with synthetic and real-life datasets. Incremental cluster validity indices (iCVIs) provide an unsupervised method for monitoring the performance of online clustering algorithms. We discussed how iCVIs can be used to understand the performance of online clustering algorithms with respect to: (i) the appearance of new emerging clusters; (ii) how the clustering algorithm reacts to outlier data samples and evolving clusters, and (iii) the effect of clusters size on the iCVIs results. We also examined the ability of iCVIs to monitor the process of a streaming clustering algorithm to correctly identify new structures, and to properly handle outliers, in data with large dimensionality and large number of samples. We also showed that Random projections can be applied on the data in a streaming fashion to ameliorate the issues caused by distance calculations in very high dimensional spaces.

The data collected from in-home sensors in 110 elderly residents' apartments as part of an NIH-NLM project are another example of streaming data where the activity of the residents are captured temporally using motion and bed sensors. The goal here is to apply temporal data analysis techniques to capture early signs of illness. We developed a novel sequence-based representation to model daily activity of a resident. Then, we normalized the sequences and built a histogram to represent the daily activity of each resident. This representation enabled us to measure the similarity between different days. Using the similarity between previous days of a resident and his/her EHR data, we developed an annotation framework, which provides reasons why a specific day is abnormal from the correlation between current sensor and EHR data. We implemented our annotation system in 110 elderly residents' apartments as part of an NIH-NLM project to produce annotations for each day with a health alert. It has been running in our database for around one year and a half.

Our immediate next step is to investigate whether the annotation framework can help the nurses in monitoring the in-home sensor data. To this end, we are working to develop a deep learning-based annotation system and to compare the performance with our existing annotation system. We are also planning to make the multi-dimensional alert system in Chapter (9) live in our server and compare its performance with the existing one-dimensional alert system. For the streaming clustering, another Ph.D student at the lab is already using it for early warning alert system, which is a suitable application for it.

VITA

Omar Ibrahim graduated from the department of Electrical Engineering, Tikrit University, with first class honor. After graduation, he was appointed as a laboratory instructor at the same department. He was awarded a scholarship to finish his master degree in the US in 2010. He graduated from University of Missouri-Columbia with Mater of Science in Electrical and Computer Engineering in 2012. After that, he worked as an instructor at the department of Electrical Engineering, where he was conducting research and teaching undergraduate classes. Also, he was the director of the Alumni Center at the same university. In 2014, he joined the University of Missouri-Columbia to finish his Ph.D degree under the supervision of Professor James Keller. He joined the Center for Eldercare and Rehabilitation Technology as a Graduate Research Assistant, where he worked on two federally funded grants and published his work at various peer reviewed journals and conferences. During his doctoral degree, he worked on developing machine-learning algorithms to predict early signs of illness in Elderly. He also developed multiple algorithms for streaming data analysis and temporal decision making. His research interests include machine learning, computational intelligence, NLP, temporal decision making, deep learning and sensor based systems that improve general wellbeing. He enjoys spending time with his family, playing soccer, and travelling.