# Visualisation of Cluster Dynamics and Change Detection in Ubiquitous Data Stream Mining

Brett Gillick, Mohamed Medhat Gaber, Shonali Krishnaswamy and Arkady Zaslavsky

Faculty of Information Technology, Monash University
{Brett.Gillick, Mohamed.Medhat.Gaber, Shonali.Krishnaswamy, Arkady.Zaslavsky}@infotech.monash.edu.au

**Abstract.** The process of Ubiquitous data mining (UDM) allows data stream mining operations to be conducted on handheld devices with limited resources. Algorithms which take advantage of visualisation can assist users in understanding and interpreting data mining results more quickly. However, there are currently no on-line real-time visualisation tools to complement the UDM algorithms. In this paper we investigate the use of visualisation techniques applied to the cluster change detection domain in a UDM environment. We demonstrate a proof of concept implementation for visualising cluster dynamics and cluster change detection.

## 1 Introduction

Handheld devices are continually increasing in both storage and processing capabilities. This increasing power has allowed lightweight versions of traditional data mining algorithms to be developed to operate on these devices. This new application area is known as Ubiquitous Data Mining (UDM), the process of performing mining of data streams on resource limited devices [7].

UDM allows "anytime, anywhere" [13, 19] analysis of streaming data for mobile users. Analysis of data in real-time allows on the spot decision making and action to be taken as changes in the data occur.

Algorithms have been developed to enable resource limited devices to perform data mining operations [8, 12]. These algorithms are designed to work within the resource constraints of the device upon which they are operating, while allowing streamed data to be mined.

Making use of the human visual system to assist the data mining process through the creation of visualisations of the data allows users to identify features within the data which would not otherwise become apparent [14]. Through the use of appropriate visualisations, like those used in traditional data mining [15], the time taken for user interpretation of the results of the data mining process can be decreased. Applying visualisation enhancement of the data mining process to the UDM domain will assist decision making by mobile users.

The ever increasing power of mobile devices means that more complicated processing and visualisation operations will be able to be performed in real-time on these devices.

The use of appropriate visualisation techniques in order to present results to users to assist with, and speed up, the decision making process of mobile users provides our motivation for this research. The use of visualisation tools in conjunction with UDM algorithms in time critical mobile environments will allow faster interpretation of results to occur.

Take, for example, the situation of a business user. This user is monitoring streaming stock market data and needs to know the instant that an important occurrence, such as a drop in share price, is detected.

Another scenario we can consider is that of a physics or chemistry laboratory scientist. In this scenario the scientist has set up many experiments in their lab which require constant monitoring. If a monitored attribute of the experiment changes significantly this could affect the outcome. Therefore the scientist must be alerted to these changes in order to direct further experimentation.

Since the research area of UDM is itself rather new, currently no on-line real-time visualisation techniques exist in order for users to view data mining results or to interact with, and guide, the data mining process on resource limited devices.

We propose a model for a novel real-time visualisation module for use in conjunction with clustering algorithms. Our approach uses three-dimensional graphics to present interactive visualisations of UDM clustering and cluster change detection algorithms to users.

In this paper we discuss the issues related to the visualisation of cluster change information on a mobile device. The paper is structured as follows: in Section 2 we review data stream clustering techniques; in Section 3 we present our proposed visualisation model; Section 4 demonstrates a practical implementation of the model. In Section 5 we summarise the contributions of this paper.

## 2 Related Work

Data streams are high speed, high volume collections of continuously arriving data [2, 3, 8, 10, 18]. The high volume of data means that storage becomes impractical, so algorithms analysing the data must process data as it arrives and store synopses, or histories, of previous data to be used in future analysis. UDM algorithms must take into account, and adapt to, changing resource and connectivity conditions that are inherent when operating in a ubiquitous environment.

Various data stream clustering algorithms have been proposed in [1, 4, 6, 11, 16]. In [1], the authors present the CluStream framework which processes the data stream in two modules. The first module operates online and creates summary statistics, or micro-clustering, of the data which are passed to an offline module which is able to provide application specific analysis of the summary statistics. In [4], the authors present a method of enhancing the technique presented in [11] with their extended exponential histogram method to provide variance and k-median statistics for data

streams. The algorithm proposed by [11] uses the K-median technique to do single pass clustering of the data stream. The authors in [6] use a sampling method to create a model of an infinite data set in finite time using a function to minimise loss in the model relative to the number of samples taken from the data set. They show that the algorithm is able to speed up the standard k-means algorithm. The authors in [16] create an algorithm to cluster chunks of incoming data, then performing re-clustering of the clusters generated for each chunk. They demonstrate that it outperforms k-means.

Visualisation assists the user with interpreting and understanding complex data. It allows representations of raw data and data processing results to be created which, if presented in a meaningful way, can allow the user insights which would not otherwise be possible.

Visualising data streams is a difficult task because of the volume and rapidity of data arrival, and the inherent limitations of graphical devices to show infinitely detailed images. For example, in [17] only a small subset of the IP address information is able to be shown because the millions of possible IP addresses could not possibly be displayed on a standard screen.

One example of UDM visualisation has been provided in [12] where stock market data mining results are presented to a mobile user to assist with stock selection. In this system the processing of data for Fourier spectrums is performed by a central server, with results sent to the user's PDA for visualisation.

As we have discussed in this section, there is limited work which has been done to provide users with visualisations to aide the ubiquitous data mining process. However, these efforts do not focus on performing all of the processing required by the visualisation, on the mobile device. Therefore, we propose and develop a framework for the integration of visualisation into the UDM process by utilising the processing and graphical capabilities of mobile devices.

The following section presents our extension to a cluster change detection algorithm with a visualisation layer to present cluster histories and cluster change alerts in order to increase understanding of the algorithm output.

## 3 Visualisation of Cluster Changes

We propose and develop a visualiser for cluster change detection technique. The technique we focus on has two main components:

- Light weight clustering (LWC) [8]
- Cluster change detection [9]

A brief description of each of these components is presented below, as they form the basis upon which our model is built.

The model for lightweight clustering (LWC) proposed the use of algorithm output granularity (AOG) to solve problems associated with performing data mining operations on resource limited devices. The LWC algorithm performs a single pass over

incoming data in order to produce clusters. The LWC algorithm operates as follows (from [8]):

1. Data items arrive in sequence with a data rate.
2. The algorithm starts by considering the first point as a centre.
3. Compare any new data item with the centres to find the distance.
4. If the distance for all the centres is greater than a threshold, the new item is considered as a new centre; else increase the weight for the centre that has the shortest distance between the data item and the centre by 1 and let the new centre equals the weighted average.
5. Repeat 3 and 4.
6. If the number of centres = k (according to the available memory) then create a new centres vector. Where k is the algorithm output granularity represented by the number of clusters kept in memory.
7. Repeat 3, 4, 5, and 6.
8. If memory is full then re-cluster (incrementally integrate the clusters)

The cluster change detection algorithm in [9] analyses the output of the LWC algorithm to produce statistics about clusters. The statistical information includes such things as:

- Average of cluster means
- Standard deviation of cluster means
- Average of cluster size.
- The cluster centre's domain.

The change detection stage of the model collects the statistical information at fixed time steps and analyses it to detect interesting changes. At each time step the current state of the clusters is compared to stored information to detect if any significant changes have occurred. The changes of interest are cluster domain changes and cluster distribution changes.

The cluster change technique uses STREAM-DETECT algorithm developed by Gaber and Yu [9]. The algorithm starts with calling online clustering to form a clustering model over a time frame. The output is summarized using the above features. The online clustering is called again to form another clustering model and also summarized. The deviation in the clustering output represents change in the domain of the data stream or the distribution.

The time frame duration and the deviation threshold are determined using a preprocessing tuning algorithm. Once these parameters are set, the algorithm performance is outstanding [9].

The STREAM-DETECT algorithm is followed by a voting-based classification technique (CHANGE-CLASS) [9] to predict previously observed changes over the training time using STREAM-DETECT algorithm. The classifier works over the change features to identify the phenomenon or event.

### 3.1 A Model for Cluster Change Detection Visualisation

We propose a model for alerting mobile users to significant cluster domain and distribution changes through the use of visualisations suitable for this environment. Our model provides a visualisation layer on top of lightweight data mining algorithms.

A diagram displaying the connections between modules in the model is shown in Figure 1. The clustering module's output is used by both the change detection and visualisation modules. The visualisation module also makes use of the output of the change detection module.

Figure 1 shows the steps in the clustering and change detection process. Firstly, the LWC algorithm processes the incoming data stream to create a number of clusters to fit within available memory.

Periodically, the change detection algorithm is run using the results of the LWC clustering to produce a statistical analysis of the current set of clusters. The change detection module analyses the current set of clusters and compares them to an older set of clusters in order to determine if a change has occurred.

The visualisation module maintains a list of the most recent sets of clusters produced by the clustering module. The current set of clusters and zero or more history sets are displayed to the user with the display being updated as changes occur. The visualisation module also alerts the user to any change detection information which may be useful.

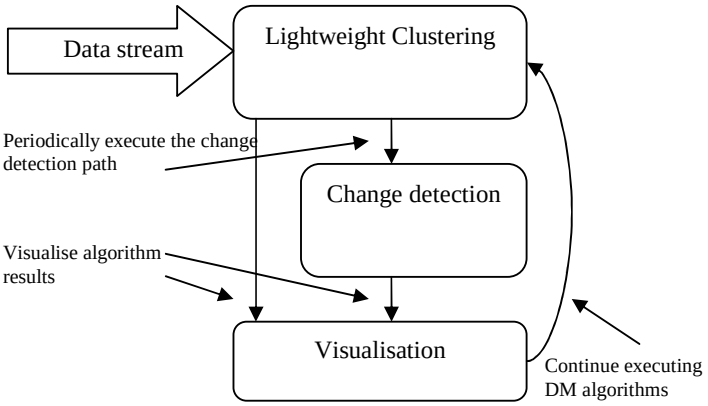


**Fig. 1.** The combined model for clustering and change detection

Without visualisation, a user is faced with the task of interpreting raw information presented in textual format, as can be seen in Figure 2. Visualisation in the context of

change detection, allows the user to more easily see where changes have occurred by observing the relative positions and sizes of a current set of clusters to previous sets. Using visualisation for change detection allows users to more quickly interpret how cluster domains, sizes, and distributions have changed over time. Moreover, the mobile user can easily be alerted to the change that has occurred.

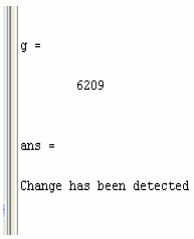


**Fig. 2.** The current output of the change detection algorithm

As an example, we can look at the situation of a laboratory scientist moving around their environment monitoring a series of experiments. Our scientist is being constantly bombarded with high data rate streams of information about various aspects of each experiment. This high volume stream of data may contain many important changes within the state of the experiment, but without suitable data mining and visualisation methods, these changes could be easily overlooked.

We propose the following algorithms for representing cluster dynamics and change detection algorithm results through the use of three-dimensional visualisation:

Cluster dynamics visualisation algorithm:

1. Let $m$ be the number of history sets of clusters stored in memory
2. Let there be $n = \{ CS_1, CS_2, \ldots, CS_m \}$ sets of clusters resulting from the clustering algorithm where $CS_1$ is the current set of clusters and $CS_m$ is the oldest stored set of clusters
3. Let there be $CC = \{ cc_1, cc_2, \ldots, cc_n \}$ cluster centres in each $CS$
4. Let $C = \{ c_1, c_2, \ldots, c_m \}$ be a set of colour codes indicating the cluster set's time stamp
5. A colour $c_i$, is assigned to represent a particular cluster set $CS_j$ where $i,j=1..m$
6. Let $G$ be the graphical object used in the visualization to represent a cluster centre and $GW$ be the graphical object associated with $G$ representing the cluster's weight
7. Each $CC$ will be coloured according to its cluster set with colour $c_i$
8. The size of each cluster's enclosing object will be equal to the cluster's weight

Change detection visualisation algorithm:

1. Let $CS_1$ be the set of clusters before a detected change
2. Let $CS_2$ be the set of clusters after a detected change
3. Let $c_1$ be the colour used to indicate a pre-change set of clusters

4. Let $c_2$ be the colour used to indicate a post-change set of clusters
5. Let $G$ be the graphical object used in the visualization to represent a cluster centre and $GW$ be the graphical object associated with $G$ representing the cluster's weight
6. Each cluster in $CS_1$ will be assigned the colour $c_1$
7. Each cluster in $CS_2$ will be assigned the colour $c_2$

The algorithm presented above has been utilised in a proof of concept implementation using three dimensional graphics for output. Solid cubes are used to represent cluster centres. A larger cube centred on each cluster indicates the weight of the cluster. Any shape or object can be used to represent a cluster centre or cluster weight, although for cluster weights the shape must be able to convey relative size information for weight comparisons between clusters.

## 4 Implementation

In the implementation we have created a data stream provider which changes attribute domains at set intervals. A visualisation is produced from the results of the LWC and change detection algorithms in order to allow users to monitor the current and previous state of the clusters. The visualisation displays clusters as well as providing alerts for statistically significant changes in the data.

The display shows cluster centres and cluster weights. The cluster centres are positioned using three attributes to produce a position in 3D space. Cluster centre positions are updated as clusters merge according to the LWC algorithm and cluster weights/sizes change as new data elements are assigned to clusters.

Each cluster centre is represented by a small blue cube, in the case of the current set of clusters, or a small cube coloured according to the cluster's age. The weight of each cluster is represented by a cube surrounding the cluster centre. This cube is coloured the same as the cluster centre it is associated with.

Figure 3 shows a sample view from the visualisation software showing cluster centres, cluster weights, and change detection information.

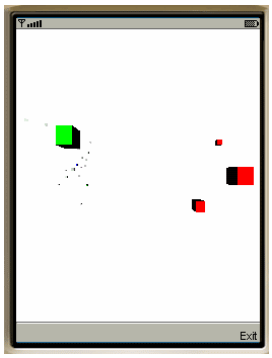


**Fig. 3.** A sample view from the visualisation software

We have implemented all of the algorithms and visualisation code using Sun Microsystems' Java 2 Mobile Edition (J2ME) language, as it has an abundance of emulators and real world devices available for prototyping and testing.

All code for the implementation has been written using Netbeans 4.1 with the optional Mobility Pack. We have made use of the emulators contained in the Mobility Pack using the Connected Limited Device Configuration (CLDC) 1.1 and Mobile Information Device Profile (MIDP) 2.0.

In order to present the visualisation to the user, our implementation makes use of the graphics features exposed by the Mobile 3D Graphics (M3G) library which is an optional package for J2ME and runs alongside MIDP. The functionality of the library can be implemented in software or hardware and is designed for devices with limited processing power and memory, which makes it ideal for our purposes. Currently, few devices have dedicated 3D hardware, but as this feature becomes more widespread it will allow more complex visualisations to be created.

The clustering and change detection algorithms behave as they are described in [8, 9]. We have also created a data generator which is necessary to provide appropriate data stream for the clustering algorithm.

For the implementation of the visualisation module, the camera is placed within the 3D scene. The user is able to manoeuvre the camera to provide different views of the clusters. The current set of clusters, as well as up to nine history sets, is displayed within the scene. We have chosen to display nine history sets of clusters as an initial step towards resource adaptability in our framework in the future. The number of history sets will change according to the resources available on the device. Cluster weights are displayed for every cluster in the form of an object centred on the cluster with a size relative to the weight of the cluster.

Currently, we do not handle the case of clusters becoming too large to fit within the camera's view. In our experimental study, we have not used a dataset that can produce very large clusters to be viewed. However, a scaling factor used when displaying the clusters would alleviate any problems which may arise. Colouring schemes are also planned to be used.

As shown in Figure 3, domain changes which have been detected by the algorithm are represented by displaying the set of clusters before the change in green, and the set of clusters which the user is being alerted about in red. From colour theory [5], blue and green were chosen as neutral, or passive, colours for the non-alert clusters. Red was chosen as the colour for the clusters about which the user is being alerted as it is a more 'active' colour.

The data generator produces a continuous stream of data elements which are consumed by the LWC algorithm to produce the clusters used by the change detection and visualisation modules. The screenshot in Figure 4 shows a view of cluster dynamics and cluster weights. The current cluster set (circled) is coloured blue while older sets are shown using grey boxes with transparency increasing with the age of the cluster.

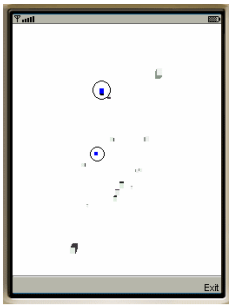


**Fig. 4.** A sample view from the visualisation software

Although this implementation uses J2ME and the M3G library any language capable of executing on a handheld device which has a 3D graphics library available could be used for the implementation. Also, taking application specific visualisation needs into consideration, a 2D graphics library could also be used.

## 5 Conclusion

In this paper we have presented our model for the visualisation of cluster dynamics and cluster change detection using our visualisation framework. Applications of such a model can vary from decision making in business applications to critical scientific and astronomical ones. Implementation and evaluation provide an evidence of efficiency in targeting the goals of these applications. Applying the model in real applications in wireless sensor networks is planned for future work.

## References

1. Aggarwal, C. C., Han, J., Wang, J., Yu, P. S.: A Framework for Clustering Evolving Data Streams, Proc. 2003 Int. Conf. on Very Large Data Bases (VLDB'03), Berlin, Germany (2003)
2. Aggarwal, C. C.: A Framework for Diagnosing Changes in Evolving Data Streams. Proceedings of the ACM SIGMOD Conference (2003)
3. Aggarwal, C. C.: On Change Diagnosis in Evolving Data Streams. IEEE Transactions on Knowledge and Data Engineering 17(5), (2005) 587-600
4. Babcock, B., Datar, M., Motwani, R., O'Callaghan, L.: Maintaining Variance and k-Medians over Data Stream Windows, Proceedings of the 2003 ACM Symposium on Principles of Database Systems (PODS 2003) (2003)
5. Birren, F.: Creative Color. Reinhold Publishing, New York (1961)
6. Domingos, P., Hulten, G.: A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering, Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA (2001) 106-113
7. Gaber, M, M., Krishnaswamy, S., and Zaslavsky, A., Ubiquitous Data Stream Mining, Current Research and Future Directions Workshop Proceedings held in conjunction with

The Eighth Pacific-Asia Conference on Knowledge Discovery and Data Mining, Sydney, Australia May 26 2004.
8. Gaber, M. M., Krishnaswamy, S., Zaslavsky, A.: Cost-Efficient Mining Techniques for Data Streams, Australasian Workshop on Data Mining and Web Intelligence (DMWI2004), Dunedin, New Zealand (2004)
9. GabYu. Gaber, M. M., Yu P. S.: Classification of Changes in Evolving Data Streams using Online Clustering Result Deviation, submitted to the 3rd International Workshop on Knowledge Discovery from Data Streams to be held in conjunction with ICML'06, June 2006.
10. Gollapudi, S., Sivakumar, D.: Framework and algorithms for trend analysis in massive temporal data sets, presented at Thirteenth ACM conference on Information and knowledge management, Washington, D.C., USA (2004)
11. Guha, S., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams, in Proc. FOCS, (2000) 359-366
12. Kargupta, H., Park, B., Pittie, S., Liu, L., Kushraj, D., Sarkar, K.: MobiMine: Monitoring the Stock Market from a PDA. ACM SIGKDD Explorations, Volume 3, Issue 2. ACM Press (2002) 37-46
13. Kargupta, H., Bhargava, R., Liu, K., Powers, M., Blair, P., Bushra, S., Dull, J., Sarkar, K., Klein, M., Vasa, M., Handy, D.: VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring. Accepted for publication in the Proceedings of the SIAM International Data Mining Conference, Orlando. (2004)
14. Keim, D. A.: Information visualization and visual data mining. IEEE Transactions On Visualization And Computer Graphics, 8(1) (2002) 1-8
15. Keim, D. A., Schneidewind, J., Sips, M.: CircleView: a new approach for visualizing time-related multidimensional data sets. AVI 2004 (2004) 179-182
16. O'Callaghan, L., Mishra, N., Meyerson, A., Guha, S., Motwani, R.: Streaming-data algorithms for high-quality clustering. Proceedings of IEEE International Conference on Data Engineering (2002)
17. Wegman, E., Marchette, D.: On some techniques for streaming data: A case study of Internet packet headers, Journal of Computational and Graphical Statistics, 12(4) (2003) 893-914
18. Wong, P. C., Foote, H., Adams, D., Cowley, W., Thomas, J.: Dynamic Visualization of Transient Data Streams, IEEE Symposium on Information Visualization (2003)
19. Zaki, M. J.: Online, Interactive and Anytime Data Mining, guest editorial for special issue of SIGKDD Explorations, Volume 3, Issue 2 (2002) i-ii