Sensors **2014**, *14*, 18960-18981; doi:10.3390/s141018960

OPEN ACCESS
sensors
ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

Multivariate Spatial Condition Mapping Using Subtractive Fuzzy Cluster Means

Hakilo Sabit * and Adnan Al-Anbuky

Electrical and Electronic Engineering, Auckland University of Technology, 24 St Paul Street, Auckland 1010, New Zealand; E-Mail: aalanbuk@aut.ac.nz

* Author to whom correspondence should be addressed; E-Mail: hsabit@aut.ac.nz; Tel.: +64-21-025-95310; Fax: +64-9-921-9973.

External Editor: Leonhard Reindl

Received: 24 February 2014; in revised form: 23 September 2014 / Accepted: 24 September 2014 / Published: 13 October 2014

Abstract: Wireless sensor networks are usually deployed for monitoring given physical phenomena taking place in a specific space and over a specific duration of time. The spatio-temporal distribution of these phenomena often correlates to certain physical events. To appropriately characterise these events-phenomena relationships over a given space for a given time frame, we require continuous monitoring of the conditions. WSNs are perfectly suited for these tasks, due to their inherent robustness. This paper presents a subtractive fuzzy cluster means algorithm and its application in data stream mining for wireless sensor systems over a cloud-computing-like architecture, which we call sensor cloud data stream mining. Benchmarking on standard mining algorithms, the k-means and the FCM algorithms, we have demonstrated that the subtractive fuzzy cluster means model can perform high quality distributed data stream mining tasks comparable to centralised data stream mining.

Keywords: data stream mining; sensor cloud; fuzzy clustering; wireless sensor network

---

1. Introduction

Wireless sensor networks (WSNs) are usually deployed to measure given physical phenomena over a certain space, within a specific time frame. The spatio-temporal distribution of these phenomena often correlates with certain physical events. To appropriately characterise these events-phenomena relationships over a given space for a specific time frame, continuous monitoring of the conditions is

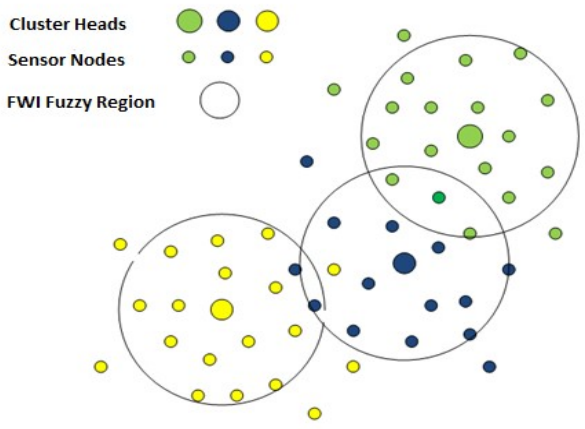
required. WSNs are perfectly suited for these tasks due to their inherent robustness. For instance, spatio-temporal distribution of weather conditions over a forestry area closely correlate with the likelihood of forest fire events in the area. WSNs can be deployed to continuously measure the weather conditions at very precise time and space-scales. WSNs can feed the current levels of these conditions into forest fire event monitoring and prediction algorithms. As a result, the computation of the likelihood of these events can be automated and graphically presented. This problem falls under the general category of multivariate spatial condition mapping, with the added complexity of continuously streaming data.

The physical sensor cloud's ability to run the computation of a given task distributed over a number of wirelessly connected nodes in parallel over a coordinated network platform resembles the cloud computing architecture. In this paper, we propose a subtractive fuzzy cluster means algorithm and its application in data stream mining for wireless sensor systems over a physical cloud-computing-like architecture, which we call sensor cloud data stream mining.

The motivational example for our paper arises from wildfire event mapping experiences. Wildfires play an important role in the structure and functioning of many of of the world's ecosystems [1]. The wildfire activity patterns that generally characterise a given area change with alterations in temperature and precipitation conditions under climate change [2,3]. Therefore, there is a need to characterise the complex relationship between wildfires, climate, vegetation and human activities at fine scales.

The Canadian Fire Weather Index (FWI) system is an example of such a system. The FWI system is an estimation of the risk of wildfire based on the empirical model developed by Van Wagner [4]. FWI is used to estimate fuel moisture content and generate a series of relative fire behaviour indices based on weather observations. The fuel moisture and fire behaviour indices are used by operational personnel to aid in the estimation of expected daily fire occurrence, potential fire behaviour and the difficulty of suppression across a fire management district, region or province [5]. The complex relationship between forest weather variables and the forest floor moisture profiles is modelled by the FWI system. The system represents the forest floor moisture profile by a set of codes and indexes known as fuel moisture codes and fire behaviour indexes. We use the FWI indexes for producing spatial maps of wildfire events, as shown in Figure 1.

Figure 1. Spatial map of weather variables and the fire weather index (FWI).



1.1. Spatial Condition Clustering

Given a set of data, clustering is the method of categorizing data into groups based on similarity. It has been used in exploratory data analysis to find unexpected patterns in datasets. It is also referred to as unsupervised learning or data mining. Clustering is inherently an ill-defined problem. Input data to the clustering algorithm is usually a vector (also known as "tuple" or "record"), which may be numerical, categorical or Boolean. Clustering algorithms must include a method for computing the similarity of or distance between vectors. Distance is the most natural method for numerical data similarity measurement. Lower values indicate more similarity. Common distance metrics include Euclidean distance and Manhattan distance. However, the distance metric does not generalise well to non-numerical data. The detailed description of basic clustering concepts, including major phases involved in clustering a dataset and examples from various areas, including biology, healthcare, market research, *etc*. can be found in [6]. k-means and fuzzy c-means (FCM) clustering algorithms have been two of the most popular algorithms in this field.

k-means [7] is one of the simplest unsupervised learning algorithms that solves the well-known clustering problem. Given a set of numeric points in d-dimensional space and an integer, the k-means algorithm generates k or fewer clusters by assigning all points to a cluster at random and repetitively computes the centroid for each cluster, reassigning each point to the nearest centroid until all clusters are stable. The major limitation of k-means is that the parameter k must be chosen in advance.

Fuzzy c-means (FCM) is a method of clustering developed by Dunn [8] and later improved by Bezdek [9]. FCM allows a piece of data to belong to two or more clusters with varying degrees of membership. In real applications, there is often no sharp boundary present between neighbouring clusters. This leads to fuzzy clustering being better suited to the data. Membership degrees between zero and one are used in fuzzy clustering instead of crisp assignments of the data to clusters. The most prominent fuzzy clustering algorithm is the fuzzy c-means, a fuzzification of k-means. A detailed description of the fundamentals of fuzzy clustering, basic algorithms and their various realisations, as well as cluster validity assessment and result visualisation are provided in [10]. Similar to k-means, a major drawback of FCM in exploratory data analysis is that it requires the number of clusters within the data space to be known beforehand. When the purpose of clustering is to automatically partition multivariate data coming from a dynamic source, the number of partitions in the data space is typically unknown. Hence, in this research, subtractive clustering and FCM algorithms are combined to implement an algorithm that does not require prior information of the number of clusters in the data space. The proposed algorithm is called the subtractive fuzzy cluster means algorithm (SUBFCM) [11].

1.2. Data Stream Mining Algorithm

Data stream mining is the process of extracting knowledge structures from continuous data streams. A data stream is an ordered sequence of instances that in many applications of data stream mining can only be read a few times using limited computing and storage capabilities. Examples of data streams include among others, computer network traffic, phone conversations, ATM transactions, web searches and sensor data. In data mining, we are interested in techniques to find and describe structural patterns in the data, as a tool to help explain the data and make predictions from it [12]. One of the

popular data mining techniques in a centralised environment is data clustering. The general goal of the clustering technique is to decompose or partition datasets into groups, such that both intra-group similarity and inter-group dissimilarity are maximised [13]. WSNs can benefit a great deal from stream mining algorithms in terms of energy conservation and efficient services. However, for WSNs to achieve significant energy conservation, the data stream mining has to be performed distributed within the network, due to their resource constraints [11,14]. In a distributed (non-centralised) computing environment, the most prominent works include the state-of-the-art in learning from data streams [15], dynamic learning models in non-stationary environments [16] and incremental or evolving clustering methods [17–19]. Data mining applications place special requirements on clustering algorithms available to the WSN nodes, including the ability to find clusters embedded in subspaces of high dimensional data and scalability. The algorithm is also required to produce frequent summaries of the corresponding inputs from the network sensor nodes. In stream mining [20,21], WSN data mining applications further place strict requirements on the underlying algorithm. Collecting data generated in a WSN to a central location and performing data mining is undesirable, due to the energy and bandwidth limitations. Therefore, the data mining algorithm has to perform in-network and autonomously on limited resource applications. The algorithm has to converge as fast as possible over the limited datasets to ensure that the processor can take on the next set of streams. The novelties of this work lie in concurrently addressing these WSN requirements, namely in-network computing, autonomously deciding the number of cluster centres, limited utilisation and high rate of convergence.

1.3. Subtractive Clustering Method

The subtractive clustering method was developed by Chiu [22]. It is a modification of mountain clustering [23] with improved computational complexity. This clustering method assumes that each data point is a potential cluster centre (prototype). A data point with more neighbouring data will have a higher potential to become a cluster centre than points with fewer neighbouring data. In subtractive clustering method, the computation is proportional to the number of data points and is independent of the dimension of the problem. Subtractive clustering considers a data point with the highest potential as a cluster centre and penalises data points close to the new cluster centre to facilitate the emergence of new cluster centres. Based on the density of surrounding data points, the potential value for each data point is calculated as follows:

$$pot_i = \sum_{j=1}^{m} e^{-\alpha \|u_i - u_j\|^2} \tag{1}$$

where u_i, u_j are data points and $\alpha = \frac{4}{r_a^2}$ and r_a is a positive constant defining a neighbourhood. Data points outside this range have little influence on the potential. Following the potential calculation of every data point, the point with the highest potential is chosen as the first cluster centre. Let u_k be the location of the first cluster centre and pot_k be its potential value. The potential of the remaining data points u_i is then revised by:

$$pot_i = pot_i - pot_k e^{-\beta \|u_i - u_k\|^2} \tag{2}$$

where $\beta = \frac{4}{r_b^2}$ and r_b is a positive constant $(r_b > r_a)$.

Thus, the data points near the first cluster centre will have greatly reduced potential and, therefore, are unlikely to be selected as the next cluster centre. The constant r_b is the neighbourhood defining radius and will have a significant reduction in potential. r_b is set to be greater than r_a to avoid closely-spaced centres. The ratio between r_a and r_b is called the squash factor (SF), which is a positive constant greater than one. The subtractive clustering algorithm is shown in Algorithm 1.

Algorithm 1. Subtractive clustering algorithm.

Inputs: $\{u_1, u_2, u_3, ...u_i\}$ Outputs: $\{c_1, c_2, c_3, ...c_C\}$
Initialise: r_a, r_b, AR, RR

1. Calculate potential of each data point, Equation (1)
2. Set the maximum potential as pot_k
3. choose the the data point corresponding to pot_k as a cluster centre candidate
4. If $pot_i > AR * pot_k$, then accept u_i as a cluster centre
5. Update the potential of each point using Equation (2) and continue
6. Else if $pot_i < RR * pot_k$, then reject u_i as a cluster centre
7. Else
8. Let d_r be relative distance
9. If $\frac{d_r}{r_a} + \frac{pot_k}{pot_i} \geq 1$ accept u_i as a cluster centre
10. Update the potential of each point using Equation (2) and continue
11. else
12. reject u_i and set the potential $pot_i = 0$
13. Select the data point with the next highest potential as the new candidate and re-test
14. end if
15. end if

The potential update process (2) will continue until no further cluster centre is found. The parameters known as the accept ratio (AR) and reject ratio (RR) together with the influence range and squash factor set the criteria for the selection of cluster centres. The accept ratio and reject ratio are the upper acceptance threshold and lower rejection threshold, respectively, and they take a value between zero and one. The accept ratio should be greater than the reject ratio.

First criteria: If the potential value ratio of the current data point to the original first cluster centre is larger than the accept ratio, then the current data point is chosen as a cluster centre.

Second criteria: If the potential value falls in between that of the accept and reject ratios, then the compensation between the magnitude of that potential value and the distance from this point to all of the previous chosen cluster centres (relative distance) is taken into consideration. If the sum of the potential value and the ratio of the shortest distance between the current data point and all other previously found cluster centres to the influence range is greater than or equal to one, then the current data point is accepted as a cluster centre.

Third criteria: If the sum of the potential value and the ratio of the shortest distance between the current data point and all other previously found cluster centres to the influence range is less than one, then the current data point is rejected as a cluster centre.

Forth criteria: If the potential value ratio of the current data point to the original first cluster centre is less than the reject ratio, then the potential value of the current data point is revised to zero and the data point with the next highest potential is tested.

1.4. Fuzzy C-Means Clustering

Fuzzy clustering algorithms are based on minimisation of the fuzzy c-means objective function formulated as:

$$J_o = \sum_{c=1}^{C} \sum_{i=1}^{m} (v_{ci})^{\theta} \left\| u_i - v_i \right\|^2 A \tag{3}$$

where v_{ci} is a fuzzy partition matrix of u,

$$v = [v_1, v_2, v_3, ...v_c] \tag{4}$$

is a vector of cluster centres, which have to be determined,

$$d_{ciA}^2 = \left\| u_i - v_i \right\|_A^2 = (u_i - v_i)^T A (u_i - v_i) \tag{5}$$

is a squared inner-product distance norm and

$$\theta \in [1, \infty] \tag{6}$$

is a parameter that determines the fuzziness of the resulting clusters.

The conditions for a fuzzy partition matrix are given as:

$$v_{ci} \in [0, 1], \qquad 1 \leq c \leq i, \qquad 1 \leq i \leq n \tag{7}$$

$$\sum_{i=1}^{c} v_{ci} = 1, \qquad 1 \leq i \leq n \tag{8}$$

$$0 < \sum_{i=1}^{n} v_{ci} < N, \qquad 1 \leq i \leq c \tag{9}$$

The value of the objective function (3) can be seen as a measure of the total variance of u_i from v_i. The minimisation of the objective function (3) is a non-linear optimisation problem that can be solved by iterative minimisation, simulated annealing or genetic algorithm methods. The simple iteration method through the first-order conditions for stationary points of (3) is known as the fuzzy c-means (FCM) algorithm.

The stationary points of the objective function (3) can be found by adjoining the constraint (8) to J_o by means of Lagrange multipliers:

$$J = \sum_{c=1}^{C} \sum_{i=1}^{m} (v_{ci})^{\theta} d_{ciA}^2 + \sum_{i=1}^{n} \lambda_i \left[\sum_{i=1}^{c} v_{ci} - 1 \right] \tag{10}$$

and by setting the gradient of J with respect to the fuzzy partition matrix u, the vector of cluster matrix v and λ to zero.

Now, if $d_{ciA}^2 > 0$, λ_i, c and $\theta > 1$, then (u, v) may minimise the objective function (3) only if:

$$v_{ci} = \frac{1}{\sum_{j=1}^{c} (d_{ciA}/d_{cjA})^2/(\theta - 1)}, \qquad 1 \leq j \leq c, \qquad 1 \leq i \leq n \tag{11}$$

and:

$$v_i = \frac{\sum_{i=1}^{n} (v_{ci})^\theta u_i}{\sum_{i=1}^{n} (v_{ci})^\theta}; \qquad 1 \leq i \leq c \tag{12}$$

This solution also satisfies the Equations (7) and (9). Equations (11) and (12) are the first-order necessary conditions for stationary points of the objective function (3). The FCM algorithm iterates through Equations (11) and (12). The sufficiency of the necessary Equations (11) and (12), as well as the convergence of the FCM algorithm is proven in [24].

Before using the FCM algorithm, the parameters are: the number of clusters, C, the fuzziness exponent, θ, the termination tolerance, ϵ, and the norm-inducing matrix, A. The fuzzy partition matrix, u, must also be initialised. Note that the FCM algorithm converges to a local minimum of the objective function (3); hence, different initialisations may lead to different results. The Fuzzy c-means algorithm is shown in Algorithm 2.

Algorithm 2. Fuzzy c-means clustering algorithm.

	Inputs:	$\{c_1, c_2, c_3, ...c_C\}$ $\{u_1, u_2, u_3, ...u_i\}$
	Outputs:	$v_i, i = 1, 2, 3, ...C$ $U_{i,j}, i = 1, 2, ..C, j = 1, 2, ...n$
	Initialise:	ϵ, θ, A
1.	For, $l = 1, 2, 3, ...$ Repeat	
2.	Compute cluster centres (prototypes):	
3.	$v_i = \frac{\sum_{i=1}^{n} (v_{ci})^\theta u_i}{\sum_{i=1}^{n} (v_{ci})^\theta}; \quad 1 \leq i \leq c$	
4.	Compute distances:	
5.	$d_{ciA}^2 = (u_i - v_i)^T A (u_i - v_i), 1 \leq c \leq C, 1 \leq i \leq n$	
6.	Update the partition matrix:	
7.	For $1 \leq i \leq n$	
8.	If $d_{ciA} > 0$ for all $c = 1, 2, ..., C$	
9.	$v_{ic} = \frac{1}{\sum_{j=1}^{C} (d_{ciA}/d_{cjA})^2/(\theta - 1)}$	
10.	Else	
11.	$v_{ci} = 0$ if $d_{ci} > 0$ and $v_{ci} \ni [0, 1]$ with $\sum_{i=1}^{C} v_{ci} = 1$	
12.	until $\|v_{ci}^{(l)} - v_{ci}^{(l-1)}\| < \epsilon$	

1.5. The SUBFCM Algorithm

Embedding an autonomous cluster mining algorithm in WSN nodes requires that the algorithm take datasets as input and generates output without data preprocessing. In applications where the number of clusters in a dataset must be discovered, the FCM algorithm cannot be used directly. For clustering WSN data autonomously, the number of cluster prototypes (categories) has to be determined from the datasets. Hence, in this research, subtractive clustering and FCM algorithms are combined to implement an algorithm that determines the number of clusters in the data space from the input datasets: subtractive fuzzy cluster means (SUBFCM).

The SUBFCM algorithm uses a subtractive clustering approach to determine the number of cluster prototypes C and the prototype centres c. The algorithm then partitions the stream into C fuzzy clusters using the prototype centres from the above step as initial fuzzy cluster centres.

Initially, the SUBFCM algorithm assumes each D-dimensional data point $u_i, i = 1, 2, 3, ..., m$ as a potential cluster centre with a measure of the potential (pot) of data points in the stream as:

$$pot_i = \sum_{j=1}^{m} e^{-\alpha \|u_i - u_j\|^2} \tag{13}$$

where $\alpha = \frac{4}{r_a^2}$ and r_a is a positive constant defining the cluster radius. A large value of r_a results in fewer large clusters, while smaller values result in more smaller diameter clusters. $\|\|$ Denotes the Euclidean distance, which defines the distance between two points $u_i(x_1, y_1, z_1)$ and $u_2(x_2, y_2, z_2)$ as being equal to the length of the vector.

$$\|X_1 - X_2\| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{14}$$

where

$$X_1 \equiv \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

and

$$X_2 \equiv \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

The measure of potential for a given data point is a function of its distances to all other points. A data point with many neighbouring points will have a high potential value. After computing the potential for every point, the point u_k with the highest potential pot_k will be selected as the first cluster centre c_1. The potential for every other point is then updated by Equation (15):

$$pot_i = pot_i - pot_k e^{-\beta \|u_i - u_k\|^2} \tag{15}$$

where $\beta = \frac{4}{r_b^2}$ and r_b is a positive constant that can be set to a value that is greater than r_a. After the first cluster centre is determined, the value of r_b determines the potential of data points becoming subsequent cluster centres. Setting $r_b > r_a$ reduces the potential of data points close to the first cluster centre and, hence, avoids closely-spaced cluster centres [25]. The parameter α is a stopping criterion and should be

selected within $(0, 1)$ [26]. α with a value close to zero will result in a large number of hidden centres, whereas α close to one leads to a small network structure.

Following the update process, the data point with the highest remaining potential is selected as the next cluster centre c_2, and the process repeats until a given threshold ϵ for the potential is reached and C such centres are computed. SUBFCM then uses the clustering criterion of squared distance d_{ci}^2 between the $i-th$ stream sample and the $c-th$ prototype and defines the objective function (J_0) as:

$$J_0 = \sum_{c=1}^{C} \sum_{i=1}^{m} v_{ci}^{\theta} d_{ci}^2 \tag{16}$$

where the squared distance function is given as:

$$d_{ci}^2 = \|u_i - Cc\|^2 \tag{17}$$

where v_{ci} represents the membership degree of the $i-th$ stream sample to the $c-th$ cluster.

Membership is determined under the conditions:

$$v_{ci} \in [0, 1], \quad c = 1, 2, 3, ..., C \quad i = 1, 2, 3, ...m \tag{18}$$

and:

$$\sum_{c=1}^{C} v_{ci} = 1, \quad i = 1, 2, 3, ..., m \tag{19}$$

where θ is a weighing exponent or fuzziness measure. If $\theta = 1$, the clustering model is reduced to the hard k-means model. The larger the θ, the fuzzier the memberships is. θ is usually set to two [24].

The stream partitioning takes place by optimizing the criterion function (16) through iteration, updating the cluster prototype centres c_j and the membership function v_{ci} as Equations (20) and (21) respectively:

$$c_j = \frac{\sum_{i=1}^{m} (v_{ij})^{\theta} u_i}{\sum_{i=1}^{m} (v_{ij})^{\theta}} \tag{20}$$

$$v_{ci} = \left(\sum_{l=1}^{C} \left(\frac{\|u_i - c_j\|}{\|u_i - c_l\|} \right)^{2/(\theta-1)} \right)^{-1} \tag{21}$$

The iteration should stop when:

$$max = \left\{ \left| v_{ci}^{(l+1)} - v_{ci}^{(i)} \right| \right\} < \epsilon \tag{22}$$

where ϵ is the termination criterion, $0 < \epsilon < 1$ and l is the iteration step.

SUBFCM takes the fuzzy radius r_a and fuzziness measure θ as inputs and autonomously reveals the structures in the data stream space. The parameter r_a determines the granularity of the structures. The smaller it is, the higher the resolution of the structures and the more computational overhead. The SUBFCM algorithm is shown in Algorithm 3.

Algorithm 3. The subtractive fuzzy cluster means (SUBFCM) algorithm.

	Inputs: $\{u_1, u_2, u_3, ...u_i\}$
	Outputs: $v_i, i = 1, 2, 3, ...C$ $U_{i,j}, i = 1, 2, ..C, j = 1, 2, ...n$
	Initialise: $\epsilon, \theta, r_a, \alpha$
1.	For, $i = 1 \leftarrow to\ m$ Repeat
2.	Compute potential (pot_i) using Equation (13):
3.	Set $c_j = pot_k$
4.	Do
5.	For $i = 1 \leftarrow to\ m - 1$
6.	Compute pot_i using Equation (15)
7.	Set $c_{j+1} = pot_k$
8.	while $pot_i > \alpha$
9.	Do
10.	For, $i = 1 \leftarrow to\ m$
11.	Compute d_{ci} using Equation (17)
12.	Set $v_{ci} = d_{ci}$
13.	Compute c_j using Equation (20)
14.	Compute v_{ci} using Equation (21)
15.	while $v_{ci}^{(l+1)} - v_{ci}^{(l)} > \epsilon$

2. Simulation and Analysis

Both the SUBFCM algorithm and the target application performances are evaluated based on simulations using TrueTime, a MATLAB/Simulink-based simulator for real-time control network systems. TrueTime facilitates co-simulation of task execution in real-time kernels, network transmissions and continuous system dynamics. The TrueTime kernel block and wireless network blocks parameters are tuned for a small-scale physical WSN deployment.

2.1. SUBFCM Performance Characterisation

A database consisting of 200 instances, each containing weather parameters, are clustered using the SUBFCM algorithm. Each instance of the database has its classification of the fire weather index (FWI) rating. There are four FWI ratings in each instance of the database: low, moderate, high and extreme. Each of the FWI ratings present in the database has 50 examples that, in total, summarise the 200 instances of the database. The graphs below (Figures 2 and 3) show the classification of each instance by comparing each pair of attributes present in the database. The SUBFCM algorithm-generated classes are shown in different colors; blue for a low FWI class, red for a moderate FWI class, green for a high FWI class and magenta for an extreme FWI class. The database contains four attributes: temperature, relative humidity, wind speed and rain fall. The different clusters do not seem to be well separated in 2D depictions. However, they are clearly separate clusters in a 3D graph.

Figure 2. 2D classification of the weather database.

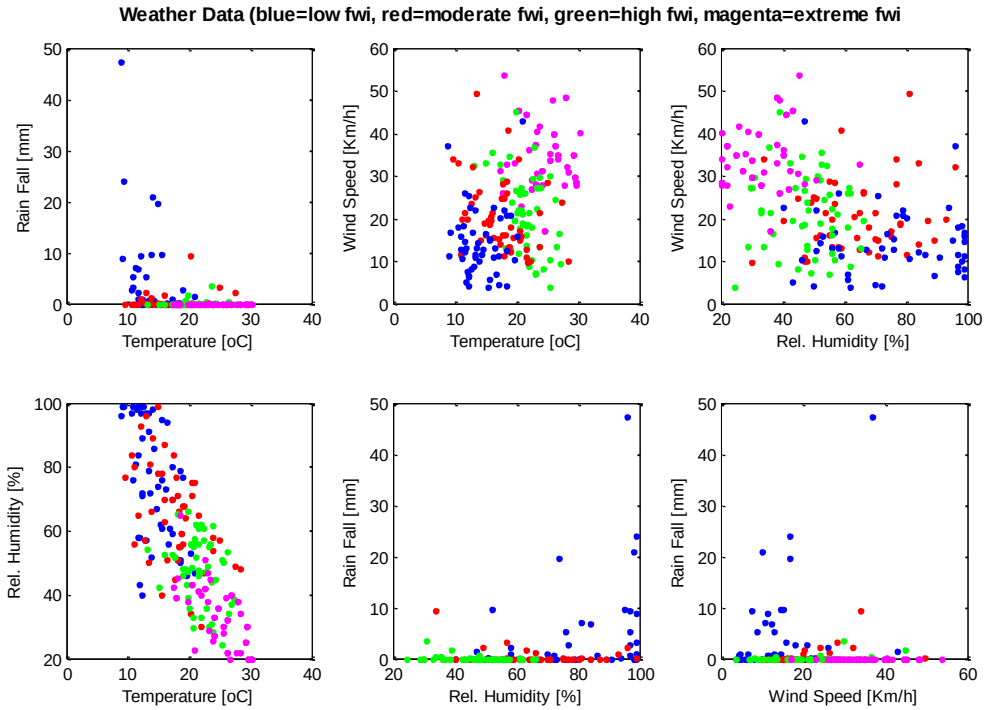
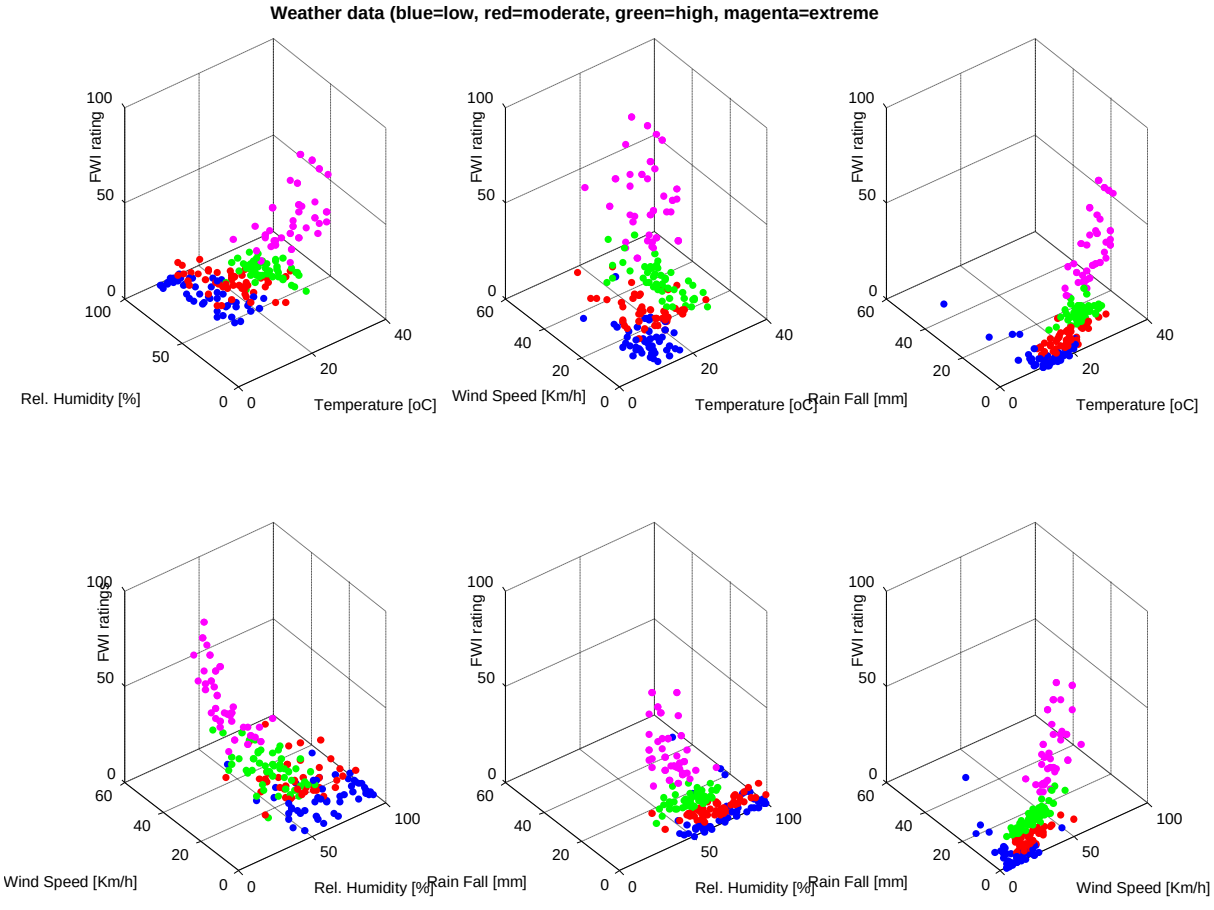


Figure 3. 3D classification of the weather database.



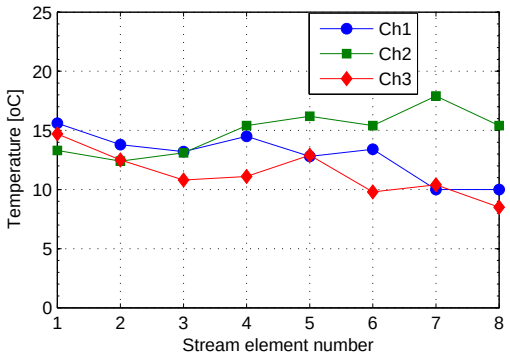
3. Results and Discussion

The results presented in this section regarding performance evaluation of the application and the network services are based on averages of 10 to 15 simulation runs with realistic parameters obtained from experimental tests. For the purpose of evaluating the system, we consider two parts: the evaluation of sensor cloud data stream mining quality and evaluation of network services quality. The dataset that we used contains 10-minute weather observations of 200 days in Sydney, Australia, recorded from June 2011, to January 2012 [27]. Each day is regarded as a data stream, and each stream has 144 points $(24 \times 60/10)$. Each data stream instance consists of temperature, relative humidity, wind speed and rainfall. The data streams are known to represent three levels of forest fire danger ratings (low, moderate and high) on the McArthur Fire Danger Index (FDI) scale [28]. Initially, we evaluate the cluster quality of the sensor cloud model using benchmark standard clustering algorithms; the k-means and the FCM. Using the same data stream sets, we vary the stream dimension and stream periods to investigate the nature and the complexity of streams that can be handled by the model. The first set of simulation considers the system performance on different stream dimensionality and stream rates. Stream sets with single to four dimensions are used to investigate the mining performance with respect to the benchmark models. Sources generating streams as slow as every minute to as fast as every second are used to investigate the effect on cluster quality and validity.

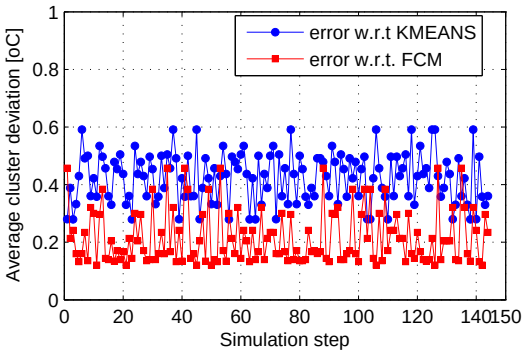
3.1. One-Dimensional (1D) Stream Results

Figure 4a shows a snapshot of the first eight elements to have arrived at the cluster heads, Ch1 to Ch3. This makes up the first sliding window of the first three cluster head streams on which the SUBFCM algorithm runs and extracts local cluster models. Analysis of clusters obtained from the SUBFCM system taking into account 144 simulation steps in Figure 4b shows that the temperature cluster centres obtained deviate by 0.4241 °C and 0.2113 °C, on average, in reference to the central k-means and FCM systems, respectively. The maximum cluster centre displacements observed are 0.59 °C and 0.46 °C compared to k-means and FCM systems, respectively. The maximum deviation is only 2.8% of the maximum temperature in the stream.

Figure 4. One-dimensional stream results. (**a**) Snapshot of the first values of member nodes; (**b**) average cluster deviations.



(a)

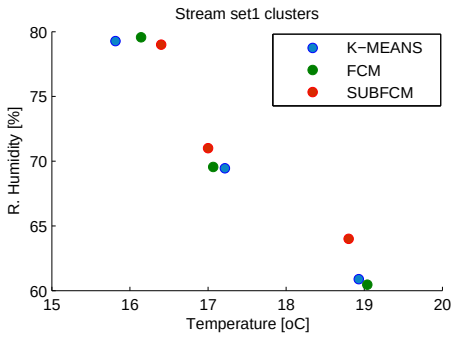


(b)

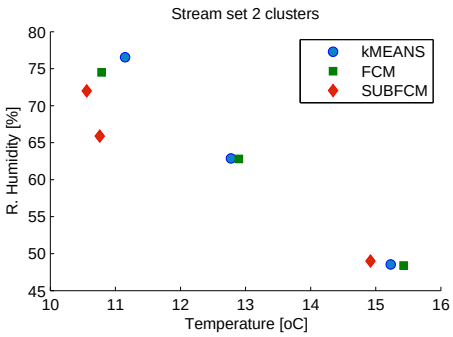
3.2. Two-Dimensional (2D) Stream Results

Figure 5a–c shows a snapshot of the first sliding windows of cluster head one to three. The first stream set, taken at the first simulation step, along with the cluster centres obtained using the distributed SUBFCM system and reference systems; k-means and FCM. Analysis of the results obtained shows that the distributed system cluster centres deviate by 3.86% and 1.46% of the cluster radius, on average, with respect to the k-means and FCM cluster centres, respectively. The maximum cluster deviation observed in this simulation is 13.22% with respect to k-means and 5.16% with respect to FCM, as shown in Figure 6.

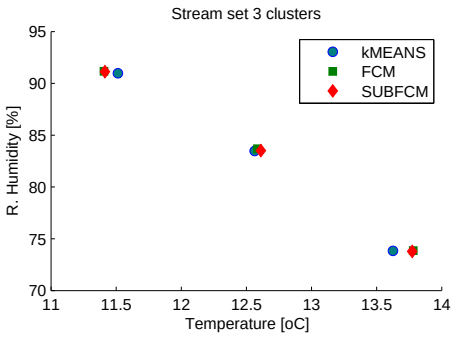
Figure 5. A snapshot of the first sliding windows of cluster heads, one to three. (**a**) snapshot of the first sliding windows of cluster head one; (**b**) snapshot of the first sliding windows of cluster head two; (**c**) snapshot of the first sliding windows of cluster head three.



(a)

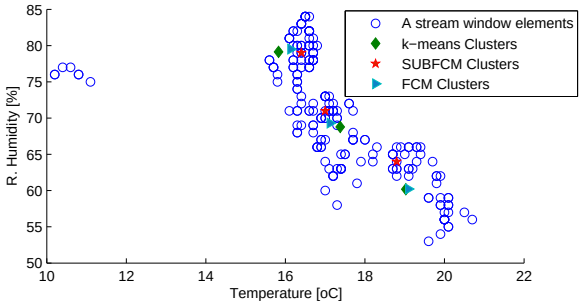


(b)



(c)

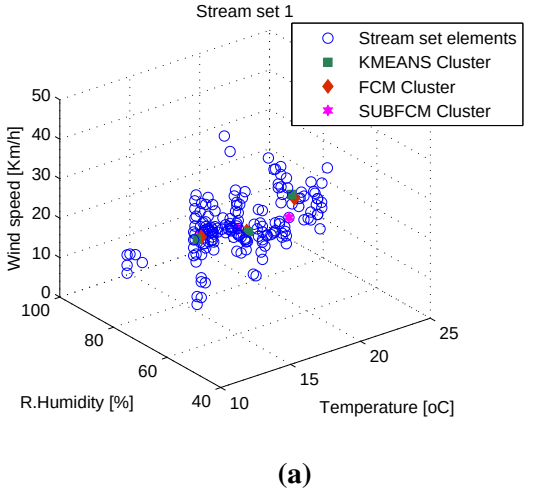
Figure 6. Stream sets and clusters from the first simulation step.



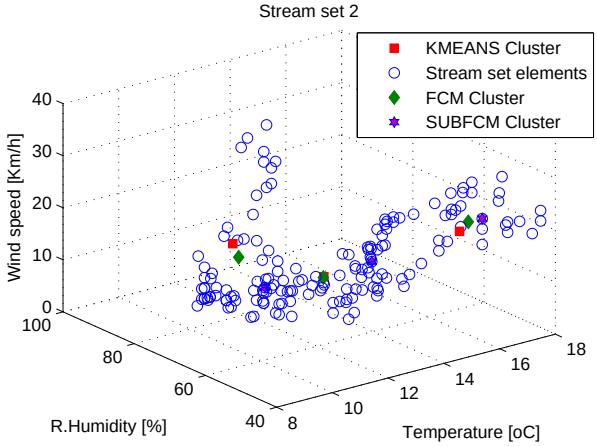
3.3. Three-Dimensional (3D) Stream Results

Three-dimensional stream analysis simulation considers three variables (temperature, relative humidity and wind speed) from the same data sources used in previous simulations. Figure 7a–c shows stream sets and clusters for the first three stream sets. Simulation result analysis shows that the average cluster deviations of the distributed SUBFCM system is 11.63% and 6.05% compared to the k-means and FCM systems, respectively. The maximum observed cluster deviation is 15.52% compared to the k-means system. Figure 8 shows average cluster deviations for the 144 simulation runs with respect to k-means and FCM, respectively.

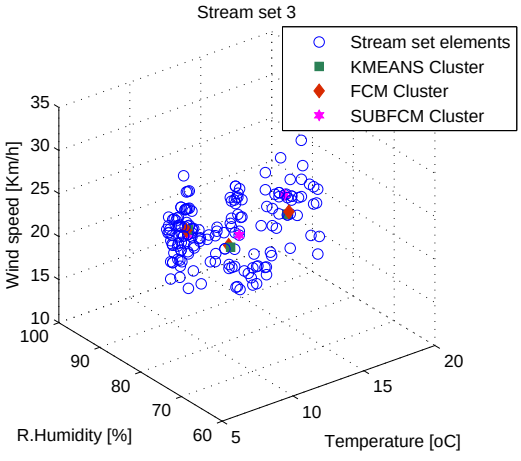
Figure 7. Stream sets and clusters for the first three stream sets. (**a**) Stream sets and clusters for the first stream set; (**b**) stream sets and clusters for the second stream set; (**c**) stream sets and clusters for the third stream set.



(a)

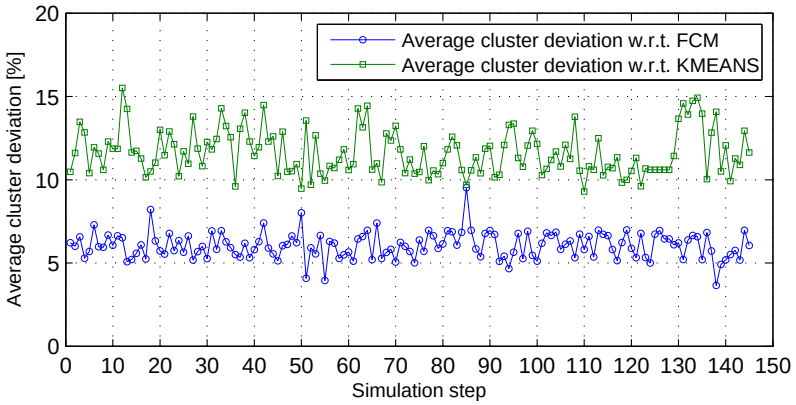


(b)



(c)

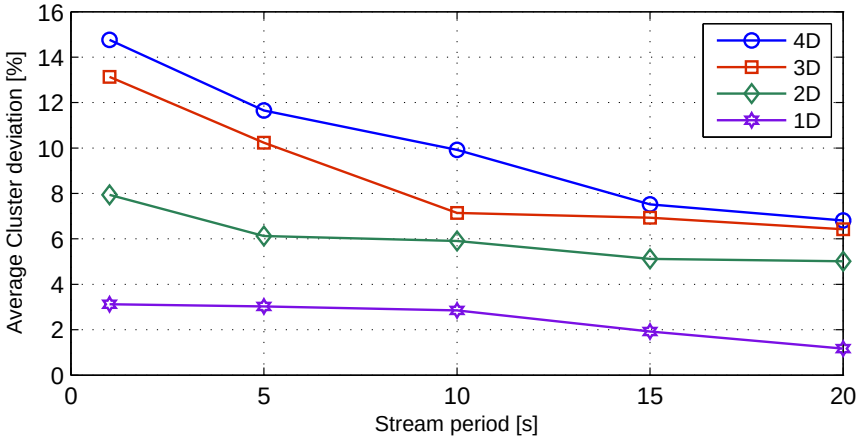
Figure 8. Average cluster deviations with respect to k-means and FCM for 3D streams.



3.4. Stream Rate Results

This simulation involves varying the stream periods. The simulation is then repeated for the different stream dimensions discussed in previous sections. We simulated stream periods of 20 s, 15 s, 10 s, 5 s and 1 s. The effect of stream periods on cluster deviation is shown in Figure 9. The streams with the higher dimensions show higher deviations, as a function of stream period. 4D stream sets show the highest standard deviation of 3.23 around the mean deviation of 10.13%, while 1D stream sets exhibit the least standard deviation of 0.85 with a mean cluster deviation of 2.42%. The results indicate that when the data streams consist of higher than 2D elements, the average cluster deviations increase with the increase in the stream periods.

Figure 9. Average cluster deviation variation with stream period.

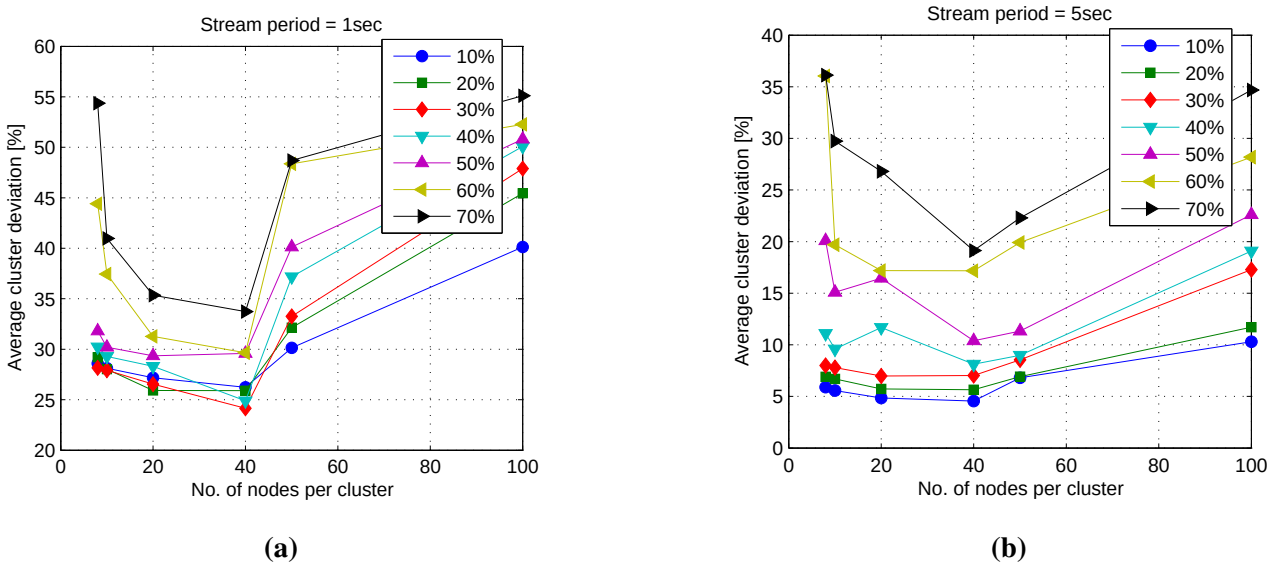


3.5. Cluster Density

The second set of simulations investigates the effect of network architectural variances on the mining performance. The variables considered in this simulation are cluster density, local model drift threshold (*i.e.*, the maximum amount of local model drift before the system starts updating the global model) and non-uniform clusters. Simulations reveal that the optimum cluster density for the best clustering

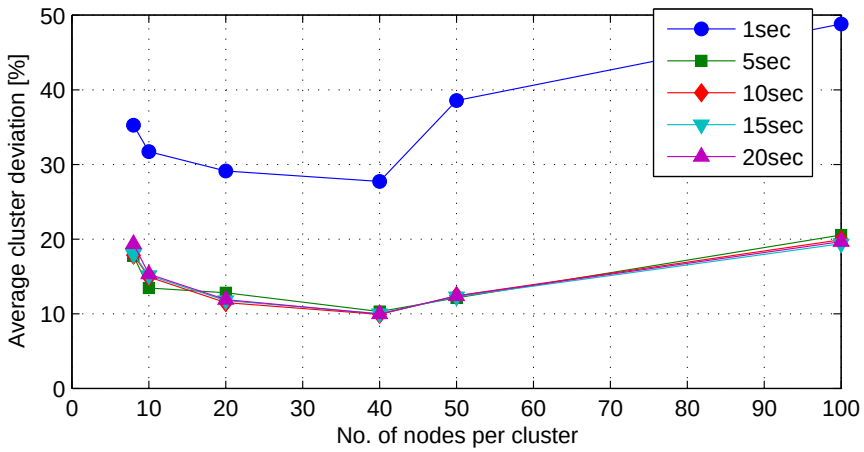
results under the given network architecture is 40 nodes per cluster. The ability of the model to handle data streams arriving in periods of longer than one second is also observed. The stream period of one second or lower is, however, too fast for the model, as manifested in the relatively higher average cluster deviations, as shown in Figures 10 and 11. High cluster deviations at a very low number of nodes per cluster in all simulations point to the fact that by dividing a given large quantity of datasets into smaller sets, mining these smaller sets at distributed locations simultaneously and incrementally extracting the hidden global structures can yield results comparable to that of mining the whole dataset at a central location. However, as the number of divisions increase, the number of distributed mining locations increases with very small sub-sets of data, and the mining results start to degrade in comparison to the central mining results.

Figure 10. Average cluster deviation with varying cluster densities. (**a**) Average cluster deviation with varying cluster densities in a 1-s stream period; (**b**) average cluster deviation with varying cluster densities in a 5-s stream period.



(a) **(b)**

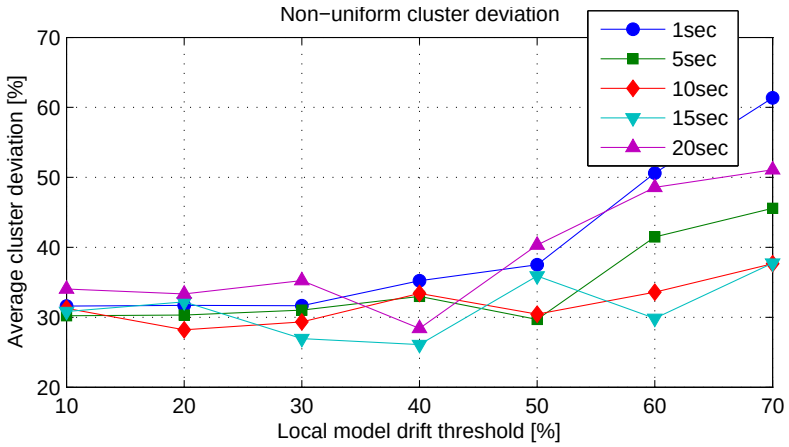
Figure 11. Average of cluster deviations for different stream periods.



3.6. Non-Uniform Cluster Density

For the case of non-uniform cluster density, the average cluster deviation as a function of the local model drift threshold for the different stream periods is plotted in Figure 12. The cluster deviations are generally higher compared to the uniform node densities per cluster. The stream periods exhibit different minimum and maximum cluster deviations for different local model drift thresholds. The irregular cluster deviation pattern is due to the assignment of the same stream period for clusters of different densities.

Figure 12. Average cluster deviation for non-uniform cluster density.



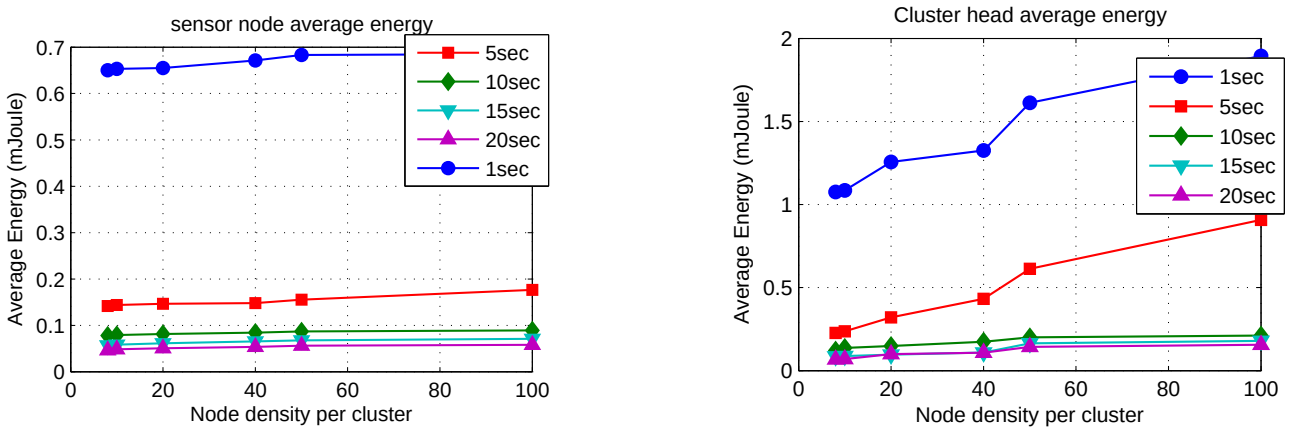
3.7. Network Quality of Service Results

The third and final set of simulation considers the impact of the model by evaluating the network services behavior. This evaluation considers the average energy consumption, the average data delivery delay and the packet delivery ratio.

3.7.1. Average Energy Consumption

The average energy consumption of a typical node taking the average of all nodes in a cluster for a single step of simulation run is shown in Figure 13a. Similarly, the average energy consumption of a typical cluster head taking the average of all cluster heads in the network for a single simulation step is shown in Figure 13b. The average energy consumption for the entire simulation run can be found by multiplying this with the number of simulation steps. The average energy consumption of the sensor nodes and the cluster heads largely vary with the stream periods and node densities per cluster. The results in Figure 13a show that streams with shorter periods result in higher average energy consumption than those with longer periods. This is because for longer stream periods, the nodes spend more time in sleep mode. The sensor nodes and cluster heads consume above 90% less energy when mining data streams with a period of 20 seconds compared to mining the same data streams with a period of one second. This could also be due to a higher packet collision and lower data delivery ratio at such fast speeds.

Figure 13. The average energy consumption of the sensor nodes and the cluster heads. (**a**) Sensor node average energy consumption; (**b**) cluster head average energy consumption.

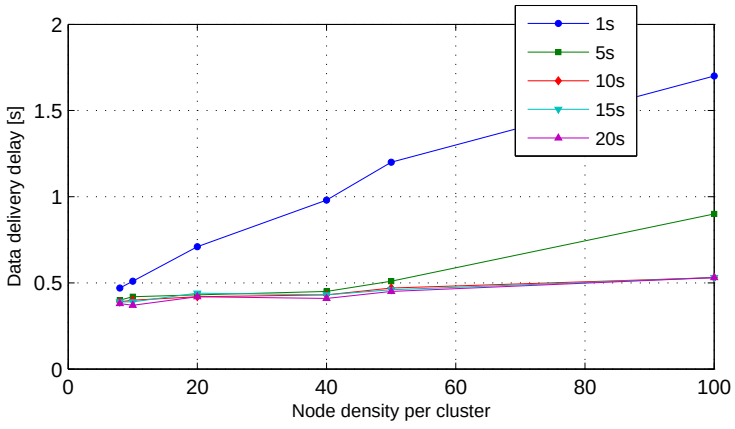


(a) **(b)**

3.7.2. Average Data Delivery Delay

The packet delivery delay is defined as the time elapsing between the instant at which a packet is generated at a source and the instant at which a copy of the packet is first received to a destination [29]. In our model, we define the data delivery delay as the time elapsed between the instant a data packet is generated at a source and the instant at which the local cluster models generated at the cluster heads, corresponding to the data packet, arrive at the sink. The average data delivery delay of a typical sensor node is shown in Figure 14. We can observe that the average data delivery delay increases as the number of nodes per cluster increases. For a stream period of one second, the average data delivery delay exceeds the stream period when more than 40 nodes exist per cluster. This situation indicates saturation of the system due to streams arriving at a rate higher than the rate at which the system can process them. However, for stream periods of five seconds or more, the average data delivery delays are well below 500 ms, with the exception of a five-second stream period at more than 50 nodes per cluster density.

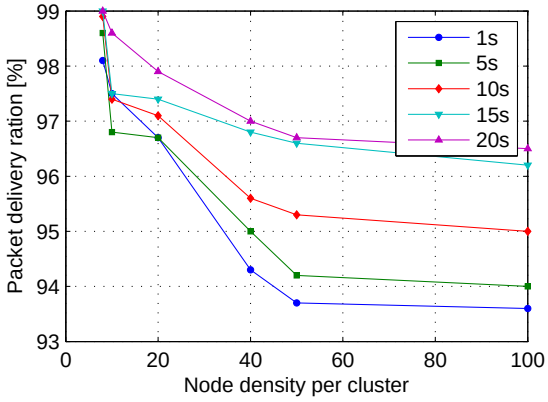
Figure 14. Packet delivery delay variation with cluster density.



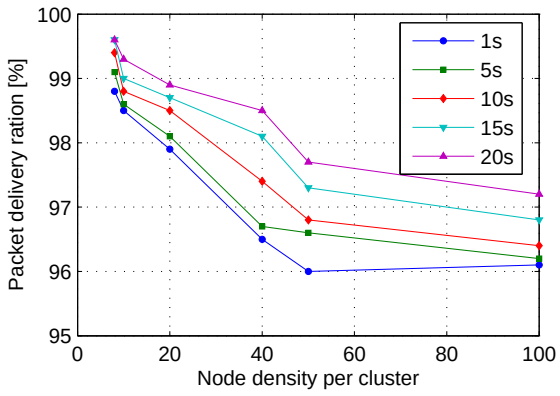
3.7.3. Packet Delivery Ratio

The packet delivery ratio is the ratio of packets received at the sink to the packets generated by all other nodes [30]. In our model, using cluster architecture and in-network processing, we define two packet delivery ratios; the sensor node-to-cluster head packet delivery ratio and the cluster head-to-sink packet delivery ratio. The former is the ratio of packets received by cluster heads to the packets sent by cluster members; whereas the later is the ratio of packets received by the sink to the packets sent by the cluster head. The packets delivery ratios are observed to vary with stream period and cluster density variations. Packet delivery ratios of between 96.7%–99% are observed when every cluster contains less than 20 nodes, no matter what the stream periods are, as exhibited in Figure 15a. However packet delivery ratios drop rapidly to as low as 93.7% to 96.8% when the node density is increased to 50 nodes per cluster, especially at faster stream periods. The stream periods have a significant impact on the rate of the packet delivery ratio drop as node density increases. This indicates that for optimal performance, applications utilizing the distributed model should limit the cluster density to below 20 if a packet delivery ratio of above 96% is desired. The packet delivery ratios of cluster head-to-sink are shown in Figure 15b. Packet delivery ratios with a drop of 99.5–96.5 are observed for node densities of eight to 50 nodes per cluster for all stream periods, except for a one-second stream period, which further drops to 96%. The lower packet deliver ratio observed is about 96% for a node density of 100 nodes per cluster. For the case of cluster head-to-sink packet delivery ratio, the stream periods do not show a significant impact on the rate of the packet delivery ratio drops, as in the sensor node-to-cluster head packet delivery ratio drops. Applications that can tolerate stream packet delivery ratios as low as 94% for the sensor node-to-cluster head ratio and 96% for the cluster head-to-sink ratio can deploy as much as 100 nodes per cluster given that the increased energy consumption, and data delivery delays, as a consequence, are acceptable.

Figure 15. The packet delivery ratios of node-to-cluster head and cluster head-to-sink. (**a**) Packet delivery ratio of sensor node-to-cluster head; (**b**) packet delivery ratio of cluster head-to-sink node.



(a)



(b)

4. Conclusions

The SUBFCM algorithm is a light unsupervised method for the analysis of data and the formulation of empirical models from the gathered data. It has been shown that a systematic combination of subtractive clustering and fuzzy c-means clustering algorithms, SUBFCM, is a light data clustering algorithm computationally suited for low resource systems, like WSN. The sensor cloud data stream mining WSN model is evaluated through simulations. The robustness of the model to different data stream dimensions and data stream rates is demonstrated through the first set of simulations. Benchmarking on standard mining algorithms, the k-means and the FCM algorithms, we have demonstrated that the model can perform high quality distributed data stream mining tasks, comparable to centralised data stream mining. The second set of simulations have shed light on the network architectural design guidelines required to satisfy desirable application demands without compromising the distributed data stream mining task integrity. The third and final set of simulations has also discussed the energy cost and network quality of service impacts for optimal system performance.

Acknowledgments

This article is extracted from my PhD thesis with Adnan Al-Anbuky as my primary supervisor. A PhD research thesis conducted under the Sensor Network and Smart Environment (SeNSe) research group at Auckland University of Technology (AUT). The authors thank the School of Engineering, AUT, for their financial support.

Author Contributions

Both authors (Hakilo Sabit and Adnan Al-Anbuky) jointly designed the study concept, developed the algorithm, performed the measurements and analysed the resulting data. Hakilo Sabit drafted the manuscript with suggestions from Adnan Al-Anbuky. Both authors read and approved the final manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Bond, W.J.; Keeley, J. Fire as a global "herbivore": The ecology and evolution of flammable ecosystems. *Trends Ecol. Evol.* **2005**, *20*, 387–394.
2. Dwyer, E.; GrAgoire, J.P.J. *Climate and Vegetation as Driving Factors in Global Fire Activity*; Springer Netherlands: Dordrecht, The Netherlands, 2000; pp. 171–191.
3. Eva, H.; Lambin, E.F. Fires and land-cover change in the tropics: A remote sensing analysis at the landscape scale. *J. Biogeogr.* **2000**, *27*, 765–776.
4. Van Wagner, C.E. *Development and Structure of the Canadian Forest Fire Weather Index System*; Technical Report 35; Canadian Forestry Service: Ottawa, Canada, 1987.

5. Amiro, B.D.; Logan, K.B.M.J.D. Fire weather index system components for large fires in the Canadian boreal forest. *Int. J. Wildland Fire* **2004**, *13*, 391–400.
6. Gan, G.; Ma, C.; Wu, J. *Data Clustering: Theory, Algorithms, and Applications*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2007.
7. MacQueen, J. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, CA, USA, 21 June–18 July 1965; Volume 1, pp. 281–297.
8. Dunn, J.C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *J. Cybern.* **1973**, *3*, 32–57.
9. Bezdek, J.C. *Pattern Recognition with Fuzzy Objective Function Algorithms*; Kluwer Academic Publishers: Norwell, MA, USA, 1981.
10. Kruse, R.; Dring, C.; Lesot, M.J. Fundamentals of Fuzzy Clustering. In *Advances in Fuzzy Clustering and its Applications*; John Wiley & Sons, Ltd: Hoboken, NJ, US, 2007; pp. 1–30.
11. Sabit, H.; Al-Anbuky, A.; Gholamhosseini, H. Data Stream Mining for Wireless Sensor Networks Environment: Energy Efficient Fuzzy Clustering Algorithm. *Int. J. Auton. Adapt. Commun. Syst.* **2011**, *4*, 383–397.
12. Witten, I.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed.; Elsevier Science: San Francisco, CA, USA, 2005.
13. Klusch, M.; Lodi, S.; Moro, G. Distributed Clustering Based on Sampling Local Density Estimates. In Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico, 9–15 August 2003; pp. 485–490.
14. Sabit, H.; Al-Anbuky, A.; Gholam-Hosseini, H. Distributed WSN Data Stream Mining Based on Fuzzy Clustering. In Proceedings of the Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, Brisbane, Australia, 7–9 July 2009; pp. 395–400.
15. Gama, J. *Knowledge Discovery from Data Streams*, 1st ed.; Chapman & Hall/CRC: London, UK, 2010.
16. Sayed-Mouchaweh, M.; Lughofer, E. *Learning in Non-Stationary Environments: Methods and Applications*; Springer: New York, NY, USA, 2012.
17. Beringer, J.; Hüllermeier, E. Online Clustering of Parallel Data Streams. *Data Knowl. Eng.* **2006**, *58*, 180–204.
18. Bouchachia, A. Evolving clustering: An asset for evolving systems. *IEEE SMC Newsl.* **2011**, *36*.
19. Lughofer, E. A dynamic split-and-merge approach for evolving cluster models. *Evol. Syst.* **2012**, *3*, 135–151.
20. Agrawal, R.; Gehrke, J.; Gunopulos, D.; Raghavan, P. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. *SIGMOD Rec.* **1998**, *27*, 94–105.
21. Agrawal, R.; Gehrke, J.; Gunopulos, D.; Raghavan, P. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, Seattle, WA, USA, 2–4 June 1998; pp. 94–105.
22. Chiu, S.L. Fuzzy Model Identification Based on Cluster Estimation. *J. Intell. Fuzzy Syst.* **1994**, *2*, pp. 267–278.

23. Yager, R.R.; Filev, D.P. Generation of Fuzzy Rules by Mountain Clustering. *J. Intell. Fuzzy Syst.* **1994**, *2*, 209–219.
24. Bezdek, J. A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* **1980**, *2*, 1–8.
25. Chen, J.; Qin, Z.J.J. A Weighted Mean Subtractive Clustering Algorithm. *Inf. Technol. J.* **2008**, *7*, 356–360.
26. Yang, P.; Zhu, Q.; Zhong, X. Subtractive Clustering Based RBF Neural Network Model for Outlier Detection. *J. Comput.* **2009**, *4*, 755–762.
27. Climate Data Online Service. Commonwealth of Australia 2011, Bureau of Meteorology. Available online: http://www.bom.gov.au/climate/data-services (accessed on 18 February 2011).
28. McArthur, A. *Fire Behaviour in Eucalypt Forests*; Technical Report 107; Commonwealth of Australia Forestry and Timber Bureau: Yarralumla, Australia, 1967.
29. Resta, G.; Santi, P. A Framework for Routing Performance Analysis in Delay Tolerant Networks with Application to Noncooperative Networks. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *23*, 2–10.
30. Kumar, P.; GuÌĹnesÌğ, M.; Mushtaq, Q.; Schiller, J. Optimizing Duty-Cycle for Delay and Energy Bound WSN Applications. In Proceedings of the 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA), Perth, WA, USA, 20–23 April 2010; pp. 692–697.