Dynamic Incremental Fuzzy C-Means Clustering

Bryant Aaron, Dan E. Tamir
Department of Computer Science,
Texas State University,
San Marcos, Texas, USA
{ba1127, dt19}@txstate.edu

Naphtali D. Rishe and Abraham Kandel
School of Computer Science
Florida International University
Miami, Florida, USA
rishen@fiu.edu, abekandel@yahoo.com

***Abstract*-Researchers have observed that multistage clustering can accelerate convergence and improve clustering quality. Two-stage and two-phase fuzzy C-means (FCM) algorithms have been reported. In this paper, we demonstrate that the FCM clustering algorithm can be improved by the use of static and dynamic single-pass incremental FCM procedures.**

Keywords-Clustering; Fuzzy C-Means Clustering; Incremental Clustering; Dynamic Clustering; Data-mining.

I. INTRODUCTION

The FCM algorithm provides a soft (fuzzy) assignment of patterns to clusters [1]. The assignment is represented by a partition matrix. The algorithm starts with either seeding the FCM with an initial partition matrix or through initial cluster centers and attempts to improve the partition matrix according to a given quality criterion.

Traditionally, in each of the FCM iterations, the algorithm is applied to the entire data set represented as a vector residing in the processor memory and representing a multi-dimensional set of measurements. Recently, however, the FCM algorithm, as well as numerous other clustering and data-mining algorithms, such as K-means, ISODATA, Kohonen neural networks (KNN), expectation maximization, and simulated annealing [1]-[6], have been exposed to a relatively new challenge referred to as "big-data." Often, the enormous amount of data available online cannot fit processors' physical memory. In fact, often the data does not even fit secondary memory. Given that input/output operations are generally the most taxing computer operations, working on the entire data set in every FCM iteration requires numerous consecutive reads of massive amounts of data. A scenario that might challenge the traditional approach to FCM clustering occurs when portions of the data are generated or become available dynamically and it is not practical to wait for the entire data set to be available.

This brings the need for incremental clustering into the forefront. Incremental clustering is also referred to as a single-pass clustering, whereas the traditional clustering is referred to as multi-pass clustering [7]-[11]. The idea is to cluster a manageable portion of the data (a data block) and maintain results for the next manageable block until exhausting the data. Under this approach, each block is processed by the algorithm a limited number of times, potentially only once. Ideally, a block, along with the preliminary number of initial centers selected should be as large as possible, occupying as much of the available internal processor memory. One might question the validity of "visiting" every data element for a limited number of iterations in a specific order as opposed to the traditional approach which considers every piece of data in each iteration.

A related approach is the multi-resolution or multistage clustering. Researchers observed that a multistage-based training procedure can accelerate the convergence and improve the quality of the training as well as the quality of the classification/decision phases of many of the clustering algorithms [5][12][13]. For example, our previous research reports show that the pyramid K-means clustering algorithm, the pyramid FCM, and multi-resolution KNN yield two-to-four times convergence speedup [13]. Both the multistage clustering and the incremental clustering apply an approach of sampling the data. In the multistage clustering, data is sampled with replacement, whereas in incremental clustering, due to the cost of replacement, the data is sampled without replacement. In both cases the validity of the sampling has to be addressed.

This paper describes a new approach for incremental FCM clustering. In difference from the pyramid FCM approach, the sampling is done without replacement. Furthermore, the sampling size is fixed. On the other hand, two measures are applied to the data in order to overcome the fact that each data block is processed only one time. First, the algorithm starts with a relatively large number of clusters and scales the number down in the last stage. This is referred to as a two-phase procedure. Hence, in the intermediate stages (first phase) each block might affect different cluster centers. A second and innovative version of the algorithm enables a dynamic number of clusters. Again, the algorithm starts with a relatively large number of clusters; however, each transaction on a block might change (increase or decrease) the number of clusters. In both cases the algorithms work on "chunks" of data referred to as blocks. This paper presents several experiments with multi-pass and static/dynamic single-pass versions of the FCM and empirically evaluates the validity of the static and dynamic incremental clustering approach.

The main contributions of this paper are: 1) a new approach, described in Section III.F, for incremental clustering, where the number of clusters is relatively high and is followed by clustering the resultant centers, is presented — this approach, increases the validity of incremental clustering, and 2) a second approach, described in Section III.G, where initially the number of clusters is relatively high and the number of clusters dynamically changes throughout execution, provides better incremental

 ISBN: 978-1-61208-343-8

clustering quality/validity, and can be used to resolve the issue of identifying the right number of cluster centers.

A literature review performed shows numerous papers on incremental clustering. To the best of our knowledge there are no reports on research that applies the operations listed in this paper to the FCM algorithm.

The rest of the paper is organized in the following way. Section II reviews related research. Section III provides details of several single-pass and multi-pass variants of FCM clustering and lists metrics used to assess the quality of clustering. Section IV describes a set of experiments conducted to assess the performance and validity of the incremental clustering algorithms described in Section III, and Section V concludes with findings and proposes further research.

II. REVIEW OF RELATED RESEARCH

Clustering is a widely-used data classification method applied in numerous research fields, including image segmentation, vector quantization (VQ), data mining, and data compression [14]-[20]. K-means is one of the most commonly used clustering algorithms; a variant of the K-means algorithm — the Linde, Buzo, and Gray (LBG) VQ algorithm with unknown probability distribution of the sources — is utilized in many applications [1][16]. The LBG algorithm has been intensively researched. Some of these research results relevant to K-means and FCM are reviewed next.

Lloyd proposes an iterative optimization method for quantizer design, it assumes that the distribution of the data is unknown and attempts to identify the optimal quantizer [21]. This approach is equivalent to 1-means (that is, K-means with $k = 1$). While Lloyd's method yields optimal minimum mean square error (MMSE) for the design of one dimensional quantizer, its extension to multi-dimensional data quantizer (i.e., VQ) with unknown distributions is not guaranteed to yield optimal results [21]. Consequently, K-means with $k > 1$ is not guaranteed to reach a global optimum.

The LBG method for VQ with unknown underlying distribution generalizes Lloyd's iterative method and sets a VQ design procedure that is based on K-means [16]. The LBG VQ procedure is currently the most commonly used/researched VQ approach. Garey has shown that the LBG VQ converges in a finite number of iterations, yet it is NP-complete [22]. Thus, finding the global minimum solution or proving that a given solution is optimal is an intractable problem. Another problem with K-means is that the number of clusters (k) is fixed and has to be set in advance of executing the algorithm. ISODATA is a generalization of K-means which allows splitting, merging, and eliminating clusters dynamically [23][24]. This might lead to better clustering (better local optimum) and eliminate the need to set k in advance. ISODATA, however, is computationally expensive and is not guaranteed to converge [2].

Several clustering algorithms and combinatorial optimization techniques, such as genetic algorithms and simulated annealing, have been devised in order to enforce the clustering algorithm out of local minima [4][25][27]. These schemes, however, require long convergence time, especially for large clustering problems. FCM and fuzzy ISODATA generalize the crisp K-means and ISODATA. The FCM clustering algorithm is of special interest since it is more likely to converge to a global optimum than many other clustering algorithms, including K-means. This is due to the fact that the cluster assignment is "soft" [28][29]. On the other hand, the FCM attempt to "skip" local optima may bear the price of numerous soft iterations and can cause an increase in computation time. FCM is used in many applications of pattern recognition, clustering, classification, compression, and quantization including signal and image processing applications such as speech coding, speech recognition, edge detection, image segmentation, and color-map generation [28]-[35]. Thus, improving the convergence time of the FCM is of special importance.

Multistage processing is a well-known procedure used for reducing the computational time of several applications, specifically, image processing procedures. This method uses a sequence of reduced resolution versions of the data to execute an image processing task. Results of execution at a low resolution stage are used to initialize the next, higher resolution stage. For example, Coleman proposes an algorithm for image segmentation using K-means clustering [14]. Hsiao has applied Coleman's technique for texture segmentation [36]. He used a $\frac{1}{16}$-sample of the image to identify k. Huang and Zhu have applied the Coleman algorithm to DCT based segmentation and color separation respectively [37][38]. Like Hsiao, they used $\frac{1}{16}$-of the image-pixels to set up the parameters of the final clustering algorithm, where the final clustering is performed on the entire image. They found that the final cluster-centers obtained in the training-stage are very close to the final cluster centers obtained from clustering the entire image. This lends itself to a two-stage K-means procedure that uses one low resolution sample to initiate the parameters of the actual clustering. Pyramid processing is a generalization of the two-stage approach where the resolution of samples is growing exponentially and each execution stage doubles the number of samples.

Additional applications of multistage architectures are reported in the literature [27][31][39]. Rosenfeld surveys the area and proposes methods for producing the multistage snapshots of an image [40]. Kasif shows that multistage linking is a special case of ISODATA [41] and Tilton uses multistage for clustering remote sensing data [42]. Tamir introduces a pyramid multistage method to non-supervised training in the context of K-means and neural networks. He has shown that the pyramid approach significantly accelerates the convergence of these procedures [12][13].

Several papers deal with accelerating the convergence of FCM [39][45][46]. Altman has implemented a two-stage FCM algorithm [47]. The first stage operates on a random sample of the data and the second stage uses the cluster centers obtained in the first stage to cluster the entire set.

Cheng improves the method proposed by Altman and has investigated a two-phase approach [31]. The first phase

implements a linear multistage algorithm which operates on small random slices of the data. Each slice contains Δ% of the data. The algorithm finds the cluster centers of the first slice (say S_1), then uses these centers as initial centers for clustering a sample that contains the first slice and an additional slice (S_2) obtained through random sampling. After running the multistage phase for n stages, the final centers for the combination of slices $\{S_1, S_2, \dots S_n\}$ which contain $n\Delta\%$ of the entire data are obtained. Next, in the second phase, these centers are used to cluster the entire data.

Other approaches for improving the convergence rate of clustering include data reduction techniques and data sampling using hypothesis testing [48][49].

A related research effort deals with clustering of very large data sets that are too big to fit into the available memory. One approach to this problem is to use incremental algorithms [10][11][27]. Several of these algorithms load a slice of the data, where the size of a slice is constrained by the available memory, and cluster this slice [7][9]. Results of clustering current slices (e.g., centers, partition matrices, dispersion, etc.) are used in the process of clustering upcoming slices. Hore has proposed a slice based single-pass FCM algorithm for large data sets [39]. The proposed method lumps data that has been clustered in previous slices into a set of weighted points and uses the weighted points along with fresh slices to commence with the clustering of the entire set in one path [39]. Another approach for clustering large data sets is to sample, rather than slice, the data [49].

Instead, in this report, the incremental approach we use has two phases. In the first phase, a very large set of clusters is used. Practically, we are trying to fit as many elements in a block and as many clusters per block as possible in the memory. In the second phase, after processing all the blocks, a process of clustering the centers obtained from the last block is applied.

It is interesting to note that K-means, FCM, Neural Networks (e.g., KNN), and many other iterative optimization algorithms have two main modes of operation, the batch mode and the parallel-update mode. For example, in the batch mode execution of FCM, each iteration considers every pattern individually and the centers are updated with respect to every pattern considered. The parallel-update mode, which is less computationally expensive and is the predominantly used mode in most current applications, assigns all the patterns to the relevant clusters and then updates the centers. In this context, the slice approach which is used for large data sets can be considered as a hybrid of batch and parallel update.

This brings the issue of parallel processing of clustering algorithms. Several ways to partition and distribute the clustering task have been considered [19][39][42][50]-[52]. One possible way is to assign a set of samples or a slice of data to each processor and eventually merge the cluster centers obtained from each processor into one set of centers. We plan to address this problem as a future research subject.

III. FUZZY C-MEANS CLUSTERING VARIANTS

In this section, we present several variants of FCM clustering.

A. The Classical Fuzzy C-Means Clustering Algorithm

The FCM algorithm is a generalization of the crisp K-means clustering. Actually, the generalization is quite intuitive. In the K-means algorithm, set membership is crisp. Hence, each pattern belongs to exactly one cluster. In the FCM, set membership is fuzzy and each pattern belongs to each cluster with some degree of membership. The following formalizes this notation.

Let $X = \{x_1, x_2, \dots, x_m\}$, where $x_i \in R^n$, be a set of m, n-dimensional vectors representing the data to be clustered into c clusters $S = \{S_1, S_2, \dots, S_c\}$ with cluster centers $\Omega = \{\omega_1, \omega_2, \dots, \omega_c\}$. Under the FCM, each element x_j belongs to every cluster S_i with some degree of membership u_{ij}. Hence, the matrix $U = [u_{ij}]$, referred to as the partition matrix, represents the fuzzy cluster assignment of each vector x_j to each cluster S_i. The goal of FCM is to identify a partition matrix U, such that U optimizes a given objective function. A commonly used FCM objective function is defined to be:

$$J_q = \sum_{i}^{c} \sum_{j}^{m} {u_{ij}}^{q-1} \cdot \left\| x_j - \omega_i \right\| \qquad (1)$$

where $q > 1$ is the weighting exponent. In this research, q is set to the most commonly used and proposed value of 2 [28].

The most common measures for FCM clustering quality are: 1) the value of the objective function, 2) the partition coefficients, 3) the classification entropy, 4) measures of deviation of the partition matrix from a matrix obtained with uniformly distributed data, and 5) measures of induced fuzziness [24][28][44]. It should be noted that some of the quality criteria are derived from distortion measures. Hence, in this case, the goal is to minimize distortion, and high quality means low distortion. In other words, the quality can be considered as the inverse of distortion. Measures 1 through 5 assume that the end result of the clustering is soft. Nevertheless, in many cases, it is desirable to obtain "hard clustering" assignment to be used for VQ, image segmentation, or other classification applications. In these cases, two additional quality criteria can be considered: 6) the rate distortion function, and 7) the dispersion matrix [2][44]. Of all these measures, 1, 6, and 7 are most commonly used. Specifically, for metric 1, the functional J_q can be interpreted as a generalized distortion measure, which is the weighted sum of the squared distances from all the points in the cluster domain to their assigned cluster center. The weights are the fuzzy membership values [28][29]. Hence, this metric is proportional to the inverse of the quality of FCM. Lower distortion denotes higher quality. Metrics 6 and 7 are further elaborated in the next section.

In general, the rate distortion function is used when the FCM is utilized for quantization. In this case, after convergence, the matrix $U = [u_{ij}]$ is defuzzified; e.g., by

using a nearest neighbor assignment. The compression rate of FCM is fixed by the selection of c. Hence, the rate distortion quality-measure boils down to the MMSE, given by:

$$D = \frac{1}{m}\sum_{i=1}^{c}\sum_{x_j \in \omega_i} \|x_j - \omega_i\| \quad (2)$$

Again, lower distortion denotes higher quality. When the clustering is used for classification, a quality criteria that measure the density of cluster as well as the relative distance between clusters can be used to estimate the recognition accuracy. In this case, a dispersion measure can be used. To elaborate, let $S = \{S_1, S_2, \dots, S_c\}$ be the set of clusters obtained through "hard clustering," and let $\Omega = \{\omega_1, \omega_2, \dots, \omega_c\}$ be the set of the corresponding cluster centers, then W_i, the within dispersion matrix of the cluster S_i, is defined to be the covariance matrix of the set of elements that belong to S_i. The within dispersion matrix of S, (W) is a given function of the entire set of the within dispersion matrices of the individual clusters. For example, the elements of W can be the averages of the compatible elements of W_i for $1 \le i \le c$. The between dispersion matrix of S, (B), is the covariance matrix of Ω. The quality of the clustering can be expressed as a function of the within dispersion matrix W and the between dispersion matrix B. A commonly used dispersion function is [44]:

$$D = tr(W)/tr(B) \quad (3)$$

where $tr(M)$ is the trace of the matrix M.

B. The Fuzzy C-Means Algorithm

The FCM consists of two main phases: setting/updating the membership of vectors in clusters and setting/updating cluster centers. Some variants of FCM start with a set of centers which induces a partition matrix [28][29]. In this case, seeding the algorithm relates to the initial selection of centers. Other variants initialize a partition matrix which induces initial centers [24]. Hence, seeding these FCM variants amounts to initializing the partition matrix. The two approaches are virtually equivalent that choosing one over the other is just a matter of convenience related to the format of data and the form of the application. We are using the second approach, where the seeding relates to selecting the initial partition matrix. Hence, in the seeding step, the membership matrix is initialized. In the next iterations, the cluster centers are calculated and the partition matrix is updated. Finally, the value of the objective function for the current classification is calculated. The algorithm terminates when a limit on the number of iterations is reached or a "short circuit condition" is met. A commonly used termination condition halts the algorithm when the derivative of the distortion function is small. Because the c-means algorithm is sensitive to the seeding method, a variety of procedures have been proposed for selecting seed points [26][52]. The following paragraphs include a formal definition of the algorithm and presents a pseudo-code.

Given a set of vectors $X = \{x_1, x_2, \dots, x_m\}$, where $x_i \in R^n$ and an initial partition matrix $U^{(0)}$, the FCM is an iterative algorithm for partitioning a set of vectors into c clusters $S = \{S_1, S_2, \dots, S_c\}$, with cluster centers $\Omega = \{\omega_1, \omega_2, \dots, \omega_c\}$. In iteration l the algorithm uses the cluster centers $\Omega^{(l)} = \{\omega_1^{(l)}, \omega_2^{(l)}, \dots, \omega_c^{(l)}\}$ induced by the partition matrix $U^{(l)}$ to re-partition the data set and obtain a new partition matrix $U^{(l+1)}$. Cluster centers at iteration l are computed according to:

$$\omega_i^{(l)} = \left(\sum_{j=1}^{m}\left(u_{ij}^{(l)}\right)^{q-1} \cdot x_j\right) \Big/ \left(\sum_{j=1}^{m}\left(u_{ij}^{(l)}\right)^{q-1}\right) \quad (4)$$

The matrix $U^{(l+1)} = \left[u_{ij}^{(l+1)}\right]$ is calculated according to:

$$U^{(l+1)} = \left[u_{ij}^{(l+1)}\right] = \sum_{l=1}^{c}\left(\frac{\|x_j - \omega_i^{(l)}\|}{\|x_j - \omega_l^{(l)}\|}\right)^{-\frac{2}{q-1}} \quad (5)$$

The process of center induction, data partition, and matrix update continues until a given termination condition, which relates to an optimization criteria or limit on the number of iterations, is met. The following is a commonly used criterion [16]:

$$\left|\frac{J_q^{(l-1)} - J_q^{(l)}}{J_a^{(l-1)}}\right| < \varepsilon \quad (6)$$

Fig. 1 is a pseudo code of the algorithm.

1. Parameters: a. $X = \{x_1, x_2, \dots, x_m\}$, $(x_j \in R^n)$ - a set of vectors b. m - the number of vectors c. c - the number of partitions d. q - a weighting exponent $(q > 1)$ e. $U^{(k)} = \left[u_{ij}^{(k)}\right]$ - the partition matrix at iteration k f. $\Omega^{(k)} = \{\omega_1^{(k)}, \omega_2^{(k)}, \dots, \omega_c^{(k)}\}$ - the set of clustering centers at iteration k g. N - the maximum number of iterations h. $J_q^{(k)}$ - the objective-function's value at iteration k	2. Set $k = 0$, choose an initial partition matrix $U^{(0)}$ 3. In iteration $k \ge 0$ let $\Omega^{(k)} = \{\omega_1^{(k)}, \omega_2^{(k)}, \dots, \omega_c^{(k)}\}$ be the induced clustering centers computed by equation 4. a. Set $U^{(k+1)} = \left[u_{ij}^{(k+1)}\right]$ according to equation 5. b. Compute $J^{(k+1)}$ according to equation 1. c. Set $k = k + 1$. 4. Stop if $k = N$; or if $k > 1$, and equation 6 holds for a small ε such as $\varepsilon = 10^{-6}$. Otherwise, go to (2).

Figure 1. Algorithm for baseline FCM

The idea behind the multistage methods reported in the next section is that an estimate of the partition matrix and the location of the cluster centers can be obtained by clustering a sample of the data. There is a trade-off that relates to the sample size. A small sample is expected to produce a fast yet less reliable estimation of the cluster

centers. This leads to a multistage approach, which involves several stages of sampling (with replacement) of the data and estimating the membership matrix for the next stage. The size of the first sample should be as small as possible. On the other hand, it should be statistically significant [44]. Each of the stages includes more objects from the data and sets the initial partition matrix of stage I according to the final partition matrix of stage $I-1$.

C. The LBG Termination Criterion

The main difference between the LBG variant of the FCM algorithm and the classical FCM algorithm is the termination condition. The LBG algorithm stops when an approximation for the derivative of the MSE given by $\frac{\left|D^{(m-1)}-D^{(m)}\right|}{D^{(m-1)}}$ is smaller than a threshold, $\left(\frac{\left|D^{(m-1)}-D^{(m)}\right|}{D^{(m-1)}}<\varepsilon\right)$.

D. The Sequential Fuzzy C-Means Algorithm

FCM can be executed in one of two basic modes; batch mode and online mode. The batch mode updates the partition matrix after considering the entire set of data. The online mode is also referred to as the sequential mode. In an epoch l of a sequential mode, for each pattern, the partition matrix $U^{(l)}$ and cluster centers $\Omega^{(l)}$ are found. The cluster centers $\Omega^{(l)}$ are used to find the distortion value and weighted distortion value. The partition matrix $U^{(l)}$ are used in the next epoch of the sequential mode.

E. The Block Sequential Fuzzy C-Means Algorithm

The block sequential mode is a compromise between the stringent computational requirements of the sequential FCM and the need to operate on data online. In this case, the clustering occurs on accumulated blocks of data. Each block is going through l epochs of FCM where the final centers of block i are used as the initial centers for block $i+1$. In many cases $l=1$. In this sense, the algorithm resembles other multistage clustering such as the pyramid FCM. The block sequential algorithm might be utilized in an iterative fashion, where each of the iterations performs l epochs of FCM on a single block of data elements at a time.

Note that all the clustering algorithms described so far assume that (at some point) the entire data set is available. Moreover, generally, due to the iterative fashion of execution, these algorithms access the same elements more than once (in different iterations). When the data is very large and cannot fit the memory of the processor, a different approach, referred to as single-pass, has to be adapted. Under the single-pass (incremental) approach, each block/data-element is accessed only one time and then removed from internal memory to provide space for new elements. This is described next.

F. The Incremental Fuzzy C-Means Algorithm

The incremental FCM algorithm presented in this paper is similar to the block sequential algorithm with the exception that each block is accessed only one time. Each block is going through a set of l epochs of FCM where the final centers of block i are used as the initial centers for block $i+1$.

The fact that each block is "touched" just one time might raise a question about the validity of the results. The results might be valid if the data elements of blocks share similar features. For example, the data elements are drawn from the same probability distribution function or the same fuzzy membership function. Alternatively, validity might be attained if the features of data elements vary "slowly" between blocks. We use two methods to improve the validity of results. First, we use a two-phase incremental algorithm. In the first phase, a very large set of clusters is used. Practically, we are trying to fit as many elements in a block and as many clusters per block as possible in the memory. In the next phase, after processing all the blocks, a process of clustering the centers obtained from the last block is applied. The second measure for increasing validity is using a relatively large number of clusters and at the same time allowing the number of clusters to change dynamically. This is described in the next section.

G. The Dynamic Incremental Fuzzy C-Means Algorithm

The dynamic incremental FCM algorithm presented in this paper is similar to the incremental FCM algorithm. The difference is that the number of clusters is allowed to change.

Several operations can change the number of clusters. First, following the ISODATA algorithm principles, clusters with too few elements might be eliminated, clusters that are too close to each other might be merged, and clusters with large dispersion might be split [13][25][27]. The criteria for merge and split might be related to the within and between dispersion of the clusters [1]-[3][43]. Other methods for changing the number of clusters might include incrementing/decrementing the number of clusters (without split/merge) based on a criterion such as a threshold on the distortion. We have implemented the threshold approach.

Each block is going through a set of l epochs of FCM where the final centers of block i are used as the initial centers for block $i+1$. Following the application of FCM on a block, a decision concerning the effective number of clusters is made and the number might be incremented or decremented based on a predetermined quality criteria threshold. We place an upper bound and a lower bound on the number of clusters, where the lower bound ensures that we still have enough clusters to maintain validity and enable the two-phase approach described above. Again, we are trying to fit as many elements in a block and a large number of clusters per block in the memory and apply a two-phase approach where the centers from the last iteration are clustered and provide the final set of clusters.

IV. EXPERIMENTS AND RESULTS

A. Experimental Setup

In order to compare and contrast the performance and validity of the FCM variants presented, we have implemented these algorithms numerous times on different

data sets, using different parameters. Three sets of data of data are used for the experiments performed. The first set (1) consists of the red, green, and blue (RGB) components of relatively small (512×512 pixels) color images. These images e.g., Lena and Baboon are used by many other researchers and the results of algorithms that use these images are published in numerous papers and books [2][6]. The images of the first set are subject to color quantization. The second set (2) is composed of the RGB components of relatively large aerial photography images ($9K \times 9K$ pixels or 81 million pixels) color this set is used for more aggressive color quantization. The third set (3) includes 20 million elements of six-dimensional synthetic data points with known centers and known distribution. Hence, data sets (2) and (3) are relatively large. It should be noted that the $9K \times 9K$ pixel images are the largest images that fit the memory of our current hardware/software configuration. This is important since we use the entire image for running non-incremental clustering in order to assess the results of the incremental clustering and this is the maximal size that can be used for non-incremental clustering. Three types of output data/results are collected: 1) records of convergences (distortion per iteration), 2) execution time, and 3) clustering quality (inverse of distortion at the final iteration).

The experiments are divided into two classes; multi-pass and single-pass. In the multi-pass experiments, we compared the performance of classical FCM, LBG based FCM, sequential FCM, and block sequential FCM. Given the constraints of these algorithms, they have been applied to a manageable data set (i.e., a data set with a medium number of elements). In the single-pass experiments, we tested the incremental and the dynamic incremental approaches with the same data used for the multi-pass algorithms and with a "huge" set of synthetic data that is not suitable for multi-pass processing. Nevertheless, to verify the results with the large data set, we ran the LBG variant of the FCM algorithm on that data using a "powerful" multicore computer. The computer worked on the data for several hours. For the dynamic incremental algorithm, we used an approach where the number of clusters is incremented/decremented by 3 based on a threshold on distortion and the number of clusters during the current epoch.

1) Color Quantization

The problem of color quantization can be stated in the following way: given an image with N different colors, choose $K \ll N$ colors such that the resulting K-color image is the least distorted version of the original image [1]. Color quantization can be implemented by applying the FCM clustering procedure to the image-pixels where each pixel represents a vector in some color representation system. For example, the clustering can be performed on the three-dimensional vectors formed by the RGB color components of each pixel in the image [1]. After clustering, each three-dimensional vector (pixel) is represented by the cluster-number to which the vector belongs and the cluster centers are stored in a color-map. The K-value image, along with the color-map, is a compressed representation of the N-colors original image. The compressed image can be used to reconstruct the original three-dimensional data set by replacing each cluster-number by the centroid associated with the cluster. In the case of 8-bit per color component and $K = 16$, the original 24-bit per pixel image is represented by a 4-bit per pixel image, along with a small color map. Hence, about six-fold compression is achieved. In this set of experiments, a block processed by the single-pass algorithm consists of an image row. The sequential algorithm is applied to every pixel of a scaled down version of the images, while the rest of the multi-pass algorithms operate on the entire set of the pixels of the original images. The experimental results are scaled to represent the distortion for the entire image. The static incremental algorithm starts with 192 clusters per block. Following the processing of the last block, the 192 centers are clustered into 16 centers and the distortion for the entire image with these centers is measured. The dynamic incremental algorithm starts with 192 clusters per block and allows fluctuations in this number. Following the processing of the last block, the centers are clustered into 16 centers and the distortion for the entire image with these centers is measured.

2) Synthetic Data

A set of 20 random cluster centers with a total of 20,000,000 six-dimensional vectors is generated. The vectors within a cluster are distributed according to a 2-D normal distribution with standard deviation of 0.05 around the center. For the single-pass experiments, the data is divided into 500 blocks of 40,000 elements per block. Other parameters are identical to the ones used for the color map quantization experiments.

B. Experimental Results

Fig. 2 shows the distortion per iteration of the multi-pass algorithms executed on the image Lena, which is a 512×512 RGB image. Fig. 2a shows the distortion results of the LBG variant the final distortion is 23.6 db. The block sequential algorithm distortion per iteration is presented in Fig. 2b. The distortion results converge to the value of 23.8 db. This is equivalent to the results of running the other single pass algorithms for several iterations. Visually, all the versions of clustering executed in this research produced about the same reconstructed image.

Fig. 3a and Fig. 3b show the results of running the single-pass incremental version on the image Lena with single row per block. Both the static and dynamic incremental algorithms converge to a distortion value of about 25 db. This is slightly higher than the multi pass algorithms. But, it is expected as the single pass algorithm "visits" every block only one time. The dynamic incremental algorithm has slightly lower distortion than the incremental algorithm. We have repeated these experiments numerous times, with different seed values using nine different images and obtained similar results.

Fig. 4a and Fig. 4b show the results of running the single-pass incremental version on the image "Neighborhood" with single row per block. Both the static This is a much larger image, and it is subject to more aggressive quantization which ends up in a monochromatic

 ISBN: 978-1-61208-343-8

image. In this case, the distortion obtained through the LBG variant on the entire image is 20.4 db. The incremental algorithm and the dynamic incremental algorithms produce a result of 23.2 db. and 24 db. Respectively. The degradation is justified by the fact that the $9K \times 9K$ image is the largest image that fits our software/hardware configuration. Hence, a larger image cannot undergo a multi pass procedure.

Fig. 5 shows the original and reconstructed versions of "Neighborhood." Figs. 6 and 7 show the same experiments with the image "Park." The results are similar with a distortion of 17.5 db., 19.5 db., and 20 db., for the LBG, incremental and dynamic incremental variants respectively.

Fig. 8a and Fig. 8b show the results of incremental clustering and dynamic incremental clustering with a large amount of synthetic data points (20,000,000 points) in a six-dimensional space). Fig. 8a shows the results of running the static incremental FCM on the synthetic data. Due to the fact that the data is drawn from a fixed distribution, the results of distortion per block are quite stable and the execution ends up at a distortion value of 0.38. The execution of the dynamic version of incremental FCM is depicted in Fig. 8b. The distortion per block is better than results obtained for the static case and stabilizes at 0.11.

V. RESULT EVALUATION

The results of the experiments reported and additional experiments performed show the utility of using a two-phase single-pass incremental FCM algorithm, where the first phase uses a large number of centers and the second phase clusters the centers obtained in the first phase into a desired size of clusters. Moreover, the dynamic clustering approach allows the number of centers in the first phase to vary and, in the case of very large data sets, outperforms the static incremental approach.

VI. CONCLUSION AND FUTURE WORK

This paper has reviewed static and dynamic single-pass and multi-pass variants of the FCM. A novel two-phase static single-pass algorithm as well as a dynamic two-phase single-pass algorithm have been presented and are showing high utility. Future research will concentrate on additional methods for dynamic change in the number of clusters in both steps of dynamic incremental FCM. In addition, we plan to initiate research on equivalent approaches in the KNN. We also plan to investigate parallel incremental algorithms. Finally, we have recently received access to an aerial photography data set where the resolution of each image is $10^6 \times 10^6$ pixels (1 tera pixels) and we plan to explore the algorithms with this data set. Additionally, we have access to a vector data set related to the aerial photography. This set contains 170 million records of varying size with an average size of 40 entries per record. Moreover, this set contains a mixture of categorical and numerical data. We plan to use this set as well for further exploration of our algorithms.

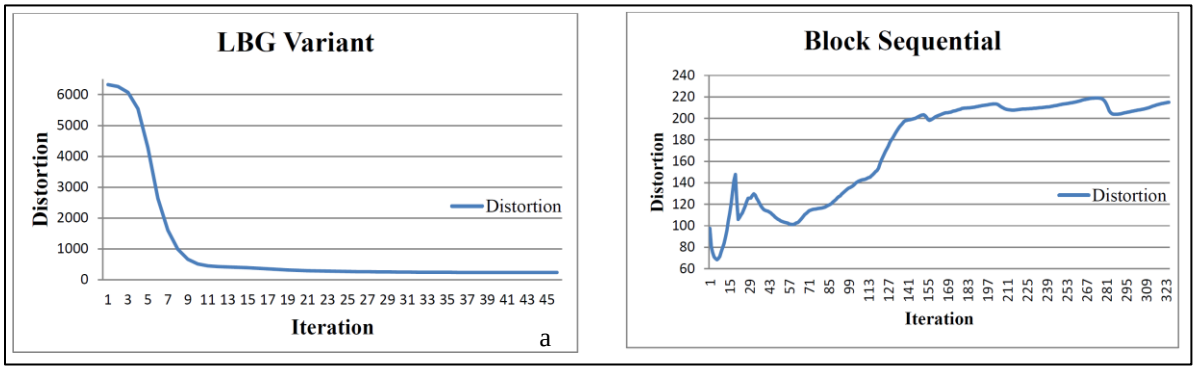

Figure 2. Distortion per iteration for the multi-pass algorithms with the Image Lena

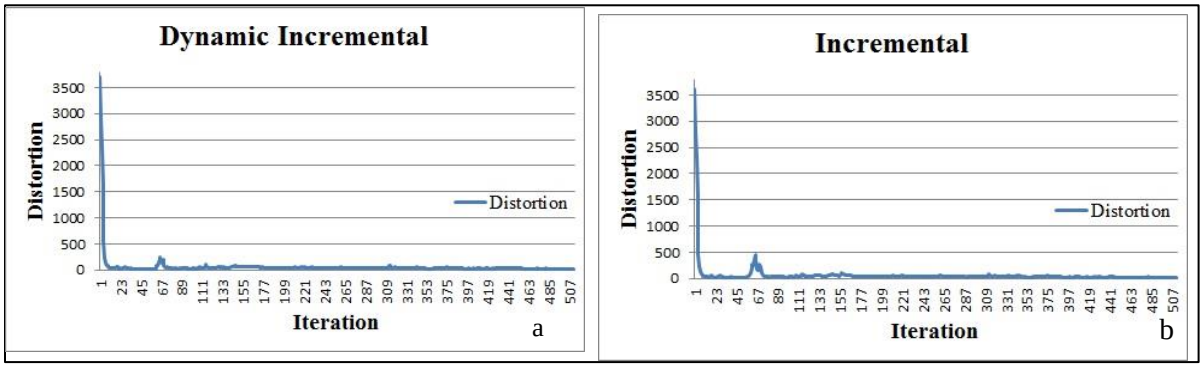

Figure 3. Incremental and Dynamic Incremental Clustering of the Image Lena

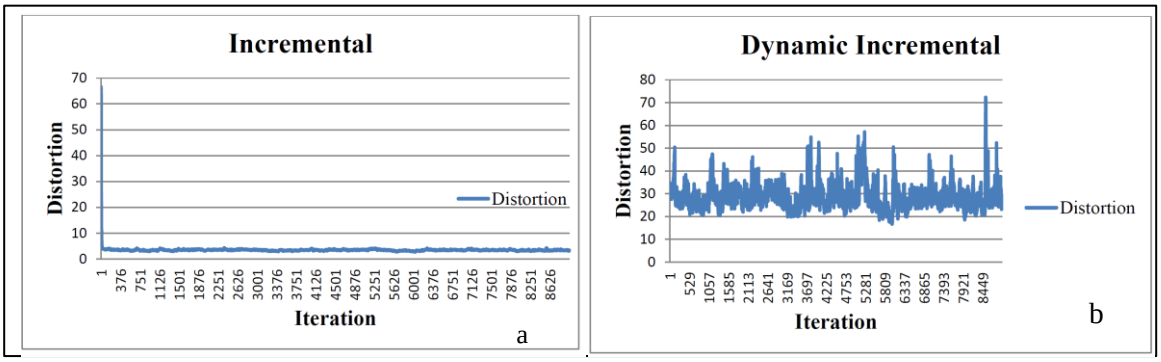

Figure 4. Incremental and Dynamic Incremental Clustering of the Image "Neighborhood"

Figure 5. Original and Reconstructed Versions of the Image "Neighborhood"

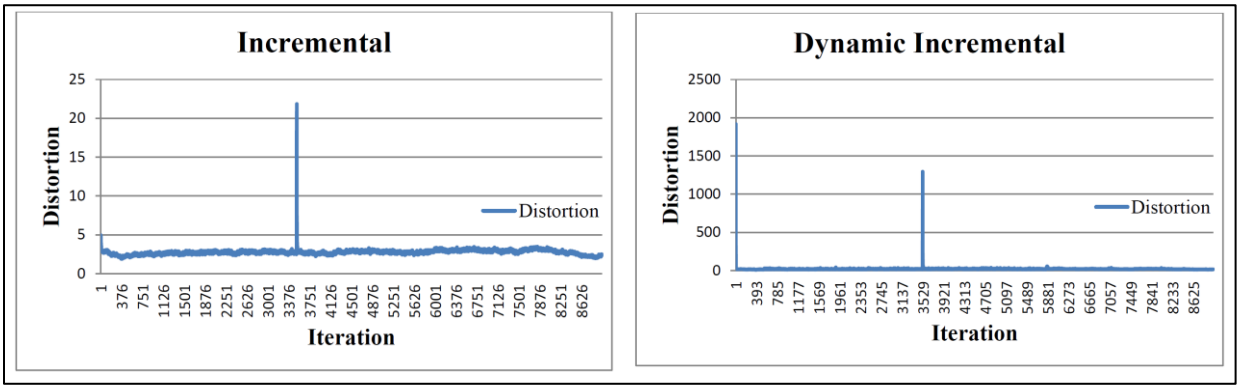

Figure 6. Incremental and Dynamic Incremental Clustering of the Image "Park"

 ISBN: 978-1-61208-343-8

Figure 7. Original and Reconstructed Versions of the Image "Park"

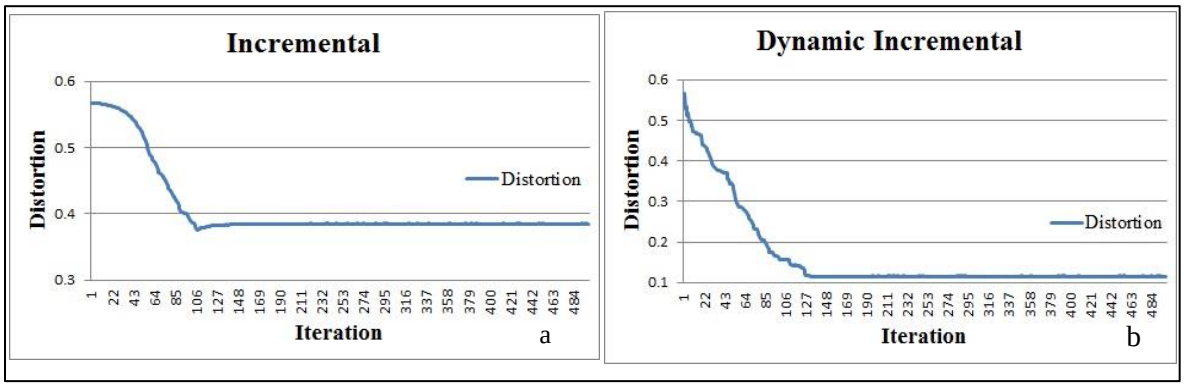

Figure 8. Incremental and Dynamic Incremental Clustering of Synthetic Data

ACKNOWLEDGMENT

This material is based in part upon work supported by the National Science Foundation under Grant Nos. I/UCRC IIP-1338922, AIR IIP-1237818, SBIR IIP-1330943, III-Large IIS-1213026, MRI CNS-0821345, MRI CNS-1126619, CREST HRD-0833093, I/UCRC IIP-0829576, MRI CNS-0959985, and FRP IIP-1230661.

REFERENCES

[1] D. E. Tamir and A. Kandel, "The Pyramid Fuzzy C-means Algorithm," International Journal of Computational Intelligence in Control, vol. 2 iss: 2, Dec. 2010, pp. 270-302.

[2] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proc. 5th Berkeley Symposium on Mathematical Statistics and Probablity, vol. 1, University of California Press, 1967, pp. 281-297.

[3] J. T. Tou and R. C. Gonzalez, Pattern Recognition Principles. London: Addison-Wesley, 1974.

[4] P. Berkhin, "Survey of Clustering Data Mining Techniques," Technical Report, Accrue Software, San Jose, CA, 2002.

[5] A. A. El-Gamal, "Using Simulated Annealing to Design Good Codes," IEEE Transactions on Information Theory, vol. 33 iss: 1, Jan. 1987, pp. 116-123.

[6] T. Kohonen, (1991) Self-organizing maps: optimization approaches. In T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, (eds), Artificial Neural Networks. Proceedings of ICANN'91, International Conference on Artificial Neural Networks, vol. II, 1991, pp. 981-990, North-Holland, Amsterdam.

[7] P. S. Bradley, U. M. Fayyad, and C. A. Reina, "Scaling Clustering Algorithms to Large Databases," in Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD98), R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro (eds), AAAI Press, Menlo Park, CA, 1998, pp. 9-15.

[8] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," in STOC '97 Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, ACM, New York, NY, USA, May 1997, pp. 626-635.

[9] F. Farnstrom, J. Lewis, and C. Elkan, "Scalability for clustering algorithms revisited," in ACM SIGKDD Explorations Newsletter, ACM, New York, NY, USA, vol. 2 iss: 1, June 2000, pp. 51-57.

[10] C. Gupta and R. Grossman, "GenIc: A Single Pass Generalized Incremental Algorithm for Clustering," in the Fourth (SIAM) International Conference on Data Mining (SDM04), 2004, pp. 147-153.

[11] J. Lin, M. Vlachos, E. Keogh, "Iterative Incremental Clustering of Time Series," Advances in Database Technology - EDBT 2004, vol. 2992, March 2004, pp. 106-122.

[12] D. E. Tamir, C. Park, and W. Yoo, "The validity of pyramid K-means clustering," in Proc. SPIE 6700 Mathematics of Data/Image Pattern Recognition, Compression, Coding, and Encryption X, with Applications, 67000D, Sept. 2007, pp. 658-675, doi:10.1117/12.735436.

[13] D. E. Tamir, Cluster Validity of Multi Resolution Competitive Neural Networks. International Conference on Artificial Intelligence, 2007, pp. 303-312.

 ISBN: 978-1-61208-343-8

[14] G. B. Coleman and H. C. Andrews, "Image segmentation by clustering," in Proceedings of the IEEE, vol. 67 iss: 5, May 1979, pp. 773-785.

[15] W. H. Equitz, "A new vector quantization clustering algorithm," in Acoustics, Speech and Signal Processing, IEEE Transactions, vol. 37 iss: 10, Oct. 1989, pp. 1568-1575.

[16] Y. Linde, A. Buzo, and R. Gray, "An Algorithm for Vector Quantizer Design," in IEEE Transactions on Communications, vol. 28 iss: 1, Jan. 1980, pp. 84-95.

[17] H. V. Lung and J.-M. Kim, "A generalized spatial fuzzy C-means algorithm for medical image segmentation," in Proceedings of the 18th International Conference on Fuzzy Systems, Jeju Island, Korea, Aug. 2009, pp. 409-414. ISSN: 1098-7584, doi:10.1109/FUZZY.2009.5276878.

[18] S. Das and A. Konar, "Automatic image pixel clustering with an improved differential evolution," Applied Soft Computing, vol. 9 iss: 1, Jan. 2009, pp. 226-236, doi:10.1016/j.asoc.2007.12.008.

[19] X. Xu, J. Jager, and H.-P. Kriegel, "A Fast Parallel Clustering Algorithm for Large Spatial Databases," Data Mining and Knowledge Discovery, vol. 3 iss: 3, Sept. 1999, pp. 263-290.

[20] P. Heckbert, "Color image quantization for frame buffer display," ACM Transactions on Computer Graphics, vol. 16 iss: 3, July 1982, pp. 297-307.

[21] S. Lloyd, "Least squares quantization in PCM," IEEE Transactions on Information Theory, vol. 28 iss: 2, Mar. 1982, pp. 129-137.

[22] M. Garey, D. Johnson, and H. Witsenhausen, "The complexity of the generalized Lloyd-Max Problem (Corresp.)," IEEE Transactions on Information Theory, vol. 28 iss: 2, Mar. 1982, pp. 255-256.

[23] G. H. Ball and J. D. Hall, "ISODATA, A Novel Method of Data Analysis and Pattern Classification," Technical Report, SRI International, Stanford 1965.

[24] J. C. Bezdek, "A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-2 iss: 1, Jan. 1980, pp. 1-8.

[25] D. Cheng, R. Kannan, S. Vempala, and G. Wang, "A divide-and-merge methodology for clustering," ACM Transactions on Database Systems (TODS), vol. 31 iss: 4, Dec. 2006, pp. 1499-1525.

[26] C. Alippi and R. Cucchiara, "Cluster partitioning in image analysis classification: a genetic algorithm approach," in Proceedings of the Computer Systems and Software Engineering, CompEuro '92, May 1992, pp. 139-144, doi: 10.1109/CMPEUR.1992.218520.

[27] S. Young, I. Arel, T. P. Karnowski, D. Rose, "A Fast and Stable Incremental Clustering Algorithm," 2010 Seventh International Conference on Information Technology: New Generations (ITNG), April 2010, pp.204-209.

[28] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithm, New York, USA: Plenum Press, 1981.

[29] A. Kandel, Fuzzy Mathematical Techniques with Applications, New York: Addison Wesley, 1987.

[30] A. M. Bensaid, L. O. Hall, J. C. Bezdek, and L. P. Clarke, "Partially supervised clustering for image segmentation," Pattern Recognition, vol. 29 iss: 5, May 1996, pp. 859-871.

[31] T. W. Cheng, D. B. Goldgof, and L. O. Hall, "Fast fuzzy clustering," Fuzzy Sets and Systems, vol. 93 iss: 1, Jan. 1998, pp. 49-56.

[32] F. Höppner, F. Klawonn, R. Kruse, and T. Runkler, Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition, New York: Wiley, 1999.

[33] S. K. Pal and D. K. D. Majumder, Fuzzy Mathematical Approach to Pattern Recognition, New York: Wiley, 1986.

[34] L. Ma and R. C. Stauton, "A modified fuzzy C-means image segmentation algorithm for use with uneven illumination patterns," Pattern Recognition, vol. 40 iss: 11, Nov. 2007, pp. 3005-3011.

[35] R. R. Yager, S. Ovchinnikov, R. M. Tong, and H. T. Nguyen, Fuzzy Sets and Applications: Selected Papers by L. A. Zadeh. New York: Wiley, 1987.

[36] J. Y. Hsiao and A. Sawchuk, "Unsupervised textured image segmentation using feature smoothing probabilistic relaxation techniques," Computer Vision, Graphics, and Image Processing, vol. 48 iss: 1, Oct. 1989, pp. 1-21.

[37] J. F. Huang, "Image Segmentation Using Discrete Cosine Transform and K-Means Clustering Algorithm," in Computer Science. 1991, Florida Institute of Technology: Melbourne, Florida.

[38] Y. Zhu, "Image Segmentation for Color Separation," in Computer Science. 1991, Florida Institute of Technology: Melbourne, Florida.

[39] P. Hore, L. W. Hall, and D. B. Goldgof, "Speedup of Fuzzy Clustering Through Stream Processing on Graphics Processing Units," in IEEE International Conference on Fuzzy Systems, London 2007.

[40] A. Rosenfeld, "Pyramids: Multi-resolution Image Analysis," 3rd Scandinavian Conference on Image Analysis, June 1983, pp. 23-28.

[41] S. Kasif and A. Rosenfeld, "Pyramid linking is a special case of ISODATA," IEEE Transactions on Systems, Man and Cybernetics, vol. SMC-13 iss: 1, Jan.-Feb. 1983, pp. 84-85.

[42] J. C. Tilton, "Multi-resolution Spatially Constrained Clustering of Remotely Sensed Data on the Massively Parallel Processor," IGARSS Symposium, 1984, pp. 661-666.

[43] E. Lughofer, "Dynamic Evolving Cluster Models Using On-line Split-and-Merge Operations," in Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops (ICMLA), vol. 2, Dec. 2011, pp. 20-26.

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

[45] R. L. Cannon, J. V. Dave, and J. C. Bezdek, "Efficient Implementation of the Fuzzy C-Means Clustering Algorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8 iss: 2, March 1986, pp. 248-255.

[46] J. F. Kolen and T. Hutcheson, "Reducing the time complexity of the fuzzy C-means algorithm," IEEE Transactions on Fuzzy Systems, vol. 10 iss: 2, Apr. 2002, pp. 263-267.

[47] D. Altman, "Efficient fuzzy clustering of multi-spectral images," in IEEE 1999 International, IGARSS '99 Proceedings, Geoscience and Remote Sensing Symposium, vol. 3, Jun.-Jul. 1999, pp. 1594-1596.

[48] S. Eschrich, J. Ke, L. O. Hall, and D. B. Goldgof, "Fast accurate fuzzy clustering through data reduction," IEEE Transactions on Fuzzy Systems, vol. 11 iss: 2, Apr. 2003, pp. 262-270.

[49] N. R. Pal and J. C. Bezdek, "Complexity reduction for "large image" processing," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 32 iss: 5, Oct. 2002, pp. 598-611.

[50] S. Rahimi, M. Zargham, A. Thakre, and D. Chhillar, "A parallel Fuzzy C-Mean algorithm for image segmentation," IEEE Annual Meeting of the Fuzzy Information (Processing NAFIPS '04), vol. 1, June 2004, pp. 234-237.

[51] A. Petrosino and M. Verde, "P-AFLC: a parallel scalable fuzzy clustering algorithm," in Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), vol. 1, Aug. 2004, pp. 809-812, ISSN: 1051-4651, doi: 10.1109/ICPR.2004.1334340.

[52] D. T. Anderson, R. H. Luke, and J. M. Keller, "Speedup of Fuzzy Clustering Through Stream Processing on Graphics Processing Units," IEEE Transactions on Fuzzy Systems, vol. 16 iss: 4, Aug. 2008, pp. 1101-1106.

 ISBN: 978-1-61208-343-8