Proceedings of the 2007 IEEE Symposium on
Foundations of Computational Intelligence (FOCI 2007)

Fuzzy Clustering and Mapping of Ordinal Values to Numerical

Mahnhoon Lee
Computational Intelligence Group
Thompson Rivers University, Canada
mlee@tru.ca

Roelof K. Brouwer, *Senior Member*, *IEEE*
Visiting Professor
University of Stellenbosch, South Africa
rkbrouwer@ieee.org

Abstract – Classification of object is considered to be the first step in many computationally intelligent systems. Objects are categorized according to their features or characteristics. Objects in the same category can be clustered into groups according to the dissimilarity in terms of their features. These groups reveal some knowledge about the objects by their partitions. Features can be numerical, ordinal or nominal. There has not been a good way to measure the dissimilarity among ordinal values, which is required for clustering. We present a novel algorithm for developing a mapping of ordinal values to numerical values for which a measure of dissimilarity exists. The algorithm is made part of the fuzzy c-means clustering algorithm. The modified algorithm finds better partitioning into clusters as well as an ordinal-numerical mapping that reveals the hidden structural knowledge of the ordinal feature. Simulations show the method to be quite effective.

I. Introduction

It is natural to consider identification of object types as one of the first steps of knowledge acquisition and learning about the surrounding environments. Objects can be classified into categories based on features that describe the objects. Objects in the same category have similar features and can be further partitioned into clusters (or called groups) according to their dissimilarity or similarity in terms of features. The clusters reveal internal structure of a set of the objects through partitioning. Clusters are information granules, and hence they can be used in computationally intelligent systems as units of learning and reasoning. Therefore the computation of dissimilarity or similarity between objects plays a very important role in computational intelligence.

Clustering [1, 2] on the basis of three types of features is generally used – numerical, ordinal and nominal. Numerical feature values lie in the real number space. Values of an ordinal feature are labels such as *Infant*, *Child* and *Adult*, with a total ordering among values of a particular feature. Nominal feature values are labels such as *Red*, *Blue* and *Yellow*, with only the relationship of equality among different values.

Numerical feature values are easily handled in clustering because they can be summed up and be compared. Also the center of a set of values can be computed. The dissimilarity or similarity between two numerical values can be easily computed by taking the difference. Nominal feature values do not have any relationship except for equality among them, and hence the dissimilarity or similarity between two values cannot be readily defined. The dissimilarity between two different values, e.g., *Red* and *Blue*, can be defined as one, and the dissimilarity between the same values can be defined as zero. Ordinal feature values are similar to nominal values in that arithmetic operators do not make sense. Also finding centers of sets of values is not obvious. The difference between ordinal values and nominal values is that ordinal values of an ordinal feature have a total ordering. For example, ordinal values *Infant*, *Child* and *Adult* of the ordinal feature of *AgeLevel* have the total ordering *Infant* < *Child* < *Adult*. This ordering gives us not only more information about the ordinal values but also a tool to compare them. We can say that the dissimilarity between *Infant* and *Child* is less than the dissimilarity between *Infant* and *Adult*. However, we cannot say how much one ordinal value is bigger than another one. Hence we do not have a good way for comparing which one is smaller or larger among the dissimilarities, e.g., the dissimilarity between *Infant* and *Child*, and the dissimilarity between *Child* and *Adult*.

The objective of this paper is to present a reasonable algorithm to map ordinal values of an ordinal feature to numerical values. We will call the one-to-one mapping "*ordinal-numerical mapping*" in the remainder of this paper. Once the ordinal values are mapped to numerical values, the dissimilarity between two ordinal values can be formulated like numerical values. The dissimilarity between two ordinal values is replaced by the dissimilarity between their images. Furthermore we can also readily associate fuzzy sets for the ordinal values, with their mapped numerical values located on the universe of discourse as centers of the fuzzy sets. One of the authors of this paper published an article about fuzzy set covering of ordinal feature values [3].

There can be various ordinal-numerical mappings for the same ordinal feature for different categories of objects. The mapping is dependent on the categories of objects investigated. For example, the mapping of *Infant*, *Child* and *Adult* to numerical values can be different for different species *Dog* and *Cat*. Our algorithm extracts the internal structure of ordinal values, which is also related to clusters of given objects. That internal structure is turned into an ordinal-numerical mapping. The algorithm is part of the fuzzy c-means clustering algorithm [4, 5] to improve the quality of clustering. The fuzzy c-means clustering algorithm is known to find good quality clusters quickly and to be noise tolerant.

In the next section, we briefly review the framework of the fuzzy c-means clustering algorithm. In section III, the problem of the mapping of ordinal values to numerical is

explained. The algorithm of the ordinal-numerical mapping is explained in section IV. Simulation results follow in section V. We conclude this paper in section VI.

II. Fuzzy C-MEANS Clustering Algorithm

The fuzzy c-means clustering algorithm for patterns consisting of numerical feature values is defined by minimization of the objective function (2.1)

$$Q = \sum_{i=1}^{c}\sum_{j=1}^{N} u_{i,j}^{m} \left\| X_j - V_i \right\|^2 \tag{2.1}$$

subject to

$$\begin{aligned} & u_{i,j} > 0 \ \forall i = 1,...,c; \forall j = 1,...,N \\ & \sum_{i=1}^{c} u_{i,j} = 1 \ \forall j = 1,...,N \\ & \sum_{j=1}^{N} u_{i,j} > 0 \ \forall i = 1,...,c \\ & m > 1 \end{aligned} \tag{2.2}$$

where $X = \{X_1,...,X_N\}$ is a data set of N patterns; c is the number of clusters; $u_{i,j}$ is the degree of membership of pattern X_j in the i^{th} cluster; V_i is the center (or called prototype) of the i^{th} cluster; and m is the indicator of fuzziness of clusters.

Only a pseudo minimum of the above objective function can be found by keeping either the membership at a particular value and finding values for the centers or fixing the centers and solving for the membership. If this is done alternatively, the optimal values for both may be approximated.

$$u_{s,t} = \frac{1}{\sum_{i=1}^{c}\left(\frac{\left\|X_t - V_s\right\|}{\left\|X_t - V_i\right\|}\right)^{\frac{2}{m-1}}}, \text{ for given } V_i, \ \forall i = 1,...,c \tag{2.3}$$

$$V_s = \frac{\sum_{j=1}^{N} u_{s,j}^{m} X_j}{\sum_{j=1}^{N} u_{s,j}^{m}}, \text{ for given } u_{s,j}, \ \forall j = 1,...,N \tag{2.4}$$

The above two formulas are computed repeatedly in a loop until some termination criteria are met. Here is the fuzzy c-means clustering algorithm.

Algorithm 1. (Fuzzy c-Means Clustering)
Initialization:
Initialize V_i for $i = 1,...,c$, with random numbers;
Repeat:
(Step 1) Compute $u_{i,j} \ \forall i = 1,...,c; \forall j = 1,...,N$; using (2.3)
(Step 2) Compute $V_i \ \forall i = 1,...,c$; using (2.4)
while $\varepsilon \le \max_{i,j}\left\{\left|old\_u_{i,j} - new\_u_{i,j}\right|\right\}$;

III. PROBLEM STATEMENT

Let the set of N patterns representing the objects be $X = \{X_1,...,X_N\}$. Without loss of generality we assume that there is only one ordinal feature with the rest being numerical. Let $L = \{l_1,...,l_n\}$ be the domain of the ordinal feature, satisfying $l_1 < l_2 < ... < l_n$, and let x_i be the ordinal feature value for a pattern X_i. We can assume that L is equal to $\{x_i \mid i = 1,...,N\}$ without loss of generality.

Consider a mapping, $g(\cdot)$, from ordinal values into the unit interval $[0,1]$, as defined in (3.1).

$$\begin{aligned} g : L &\to [0,1] \\ l_i &\mapsto a_i \end{aligned} \tag{3.1}$$

satisfying

$$\forall i = 1,...,n-1, \qquad 0 < a_i < a_{i+1} < 1 \tag{3.2}$$

Figure 1 shows the relations among the ordinal feature values and their mapped numerical values. a_0 and a_{n+1} are 0 and 1 respectively. For example, the ordinal feature x_i with value l_1 of a pattern X_i is mapped to a_1.

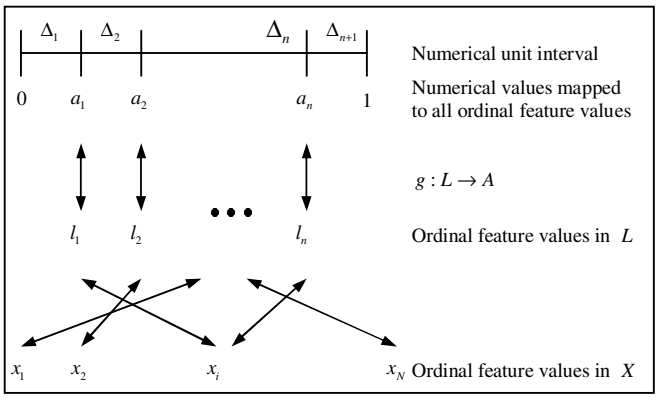

Fig. 1. Relation between ordinal feature values and their mapped numerical values

From the assumption that L is equal to $\{x_i \mid i = 1,...,N\}$, we can have the following property,

$$\forall k = 1,...,n, \ \exists x_p \ni x_p = l_k \text{ and } g(l_k) = a_k \tag{3.3}$$

The above property (3.3) is used in the development of the modified fuzzy algorithm in the next section.

Dissimilarity can now be expressed as in (3.4), using the mapping function $g(\cdot)$ defined in (3.1).

$$dis(x_s, x_t) = dis(l_i, l_j) \triangleq \left|g(l_i) - g(l_j)\right| = a_j - a_i \tag{3.4}$$

The values of x_s and x_t are l_i and l_j respectively with $i < j$. For example, $dis(x_2, x_i) = dis(l_1, l_2) = a_2 - a_1$ in Figure 1. Thus the ordinal values have been replaced by as numerical values as defined by the ordinal-numerical mapping $g(\cdot)$. What is novel is that the numerical equivalents are not pre-assigned but rather are determined as part of the clustering process as will be seen later.

IV. MODIFIED FUZZY CLUSTERING ALGORITHM INCLUDING ORDINAL-NUMERICAL MAPPING

Clustering is done with an arbitrary initial ordinal-numerical mapping for each ordinal feature. As a part of clustering process the ordinal-numerical mappings are updated followed by the determination of new clusters. This is repeated until there is no significant improvement. As the results of the modified fuzzy clustering algorithm, improved clusters in terms of known clustering performance measures explained in [6-8] are found using updated ordinal-numerical mappings. This method does not require the assumption of equal distances between ordinal values.

Since the fuzzy c-means clustering algorithm [4, 5] finds clusters effectively even in the presence of noises, the existing algorithm is modified to include determination of the ordinal-numerical mappings. Details of the modified clustering algorithm now follow.

We assume that all the numerical feature values are normalized in the unit interval $[0,1]$. Even though the algorithm is described in terms of one ordinal feature, it is readily generalized to patterns with more than one ordinal feature.

Algorithm 2. (Adaptive Fuzzy Clustering)

Let

c be the given number of clusters;

$x_1,...,x_N$ be the ordinal feature values for N patterns;

$L=\{l_1,...,l_n\}$ be the domain of the ordinal feature covering all $x_1,...,x_N$, with the total ordering $l_1<...<l_n$;

$\varepsilon>0$ be the termination criterion;

Initialization:

Initialize the ordinal-numerical mapping $g: L \to A=\{a_1,...,a_n\}$ $\forall i=1,...,n,\ a_i=\dfrac{i}{n+1}$, where a_i corresponds to l_i;

Initialize the cluster centers $V_i, \forall i=1,...,c$, with random numbers in $[0,1]$;

Repeat:

(Step 1) Compute $u_{i,j}, \forall i=1,...,c, \forall j=1,...,N$ using (2.3);

(Step 2) Compute $V_i, \forall i=1,...,c$, using (2.4);

(Step 3) Update the ordinal-numerical mapping $g(\cdot)$, i.e., $a_i, \forall i=1,...,n$;

while $\varepsilon \le \max_{i,j}\{|old\_u_{i,j}-new\_u_{i,j}|\}$;

In the computation of $u_{i,j}$ and V_i in Step 1 and Step 2 of the above algorithm, Formulas (2.3) and (2.4) of the fuzzy c-means clustering algorithm are used, with the dissimilarity (3.4) defined on the ordinal-numerical mapping $g(\cdot)$. In Step 3, the ordinal-numerical mapping is adapted so that the quality of clusters found in Steps 1 and 2 improves.

An objective function (3.5) restricted to the ordinal feature, not all the features, is used in Step 3 in Algorithm 2, in order to find a proper ordinal-numerical mapping $g(\cdot)$. In the objective function, the ordinal feature values $x_1,...,x_N$ of N patterns, the memberships $u_{i,j}$ and the numerical values v_i of the cluster centers V_i are assumed. (Note that the objective function (3.5) is independent of the objective function (2.1) used in Steps 1 and 2. V_i is a vector containing the numerical value v_i, and v_i represents all the ordinal feature values in the i^{th} cluster. v_i is the weighted center of mapped numerical values in the ordinal feature.)

$$Q(x,u,v;a)=\sum_{i=1}^{c}\sum_{j=1}^{N}u_{i,j}^{m}\left(g(x_j)-v_i\right)^2 \tag{3.5}$$

where

c	number of clusters	
n	number of labels in the ordinal feature	
N	number of patterns	
m	fuzziness of clusters	
$g(\cdot)$	ordinal-numerical mapping	(3.6)
v_i	weighted center of the mapped numerical values of the ordinal feature values in the i^{th} cluster	
x_j	ordinal feature value of the j^{th} pattern X_j	
$u_{i,j}$	degree of membership of X_j in the i^{th} cluster	

subject to

$$0<a_1<...<a_n<1 \tag{3.7}$$

Function (3.5) can be rewritten as follows.

$$\begin{aligned} Q&=\sum_{i=1}^{c}\sum_{j=1}^{N}u_{i,j}^{m}\left(g(x_j)-v_i\right)^2 \\ &=\sum_{i=1}^{c}\sum_{l=1}^{n}\sum_{j=1}^{N}u_{i,j}^{m}\phi_{j,l}\left(a_l-v_i\right)^2 \\ &=\sum_{i=1}^{c}\sum_{l=1}^{n}\mu_{i,l}\left(a_l-v_i\right)^2 \end{aligned} \tag{3.8}$$

where

$$\begin{aligned} \phi_{j,l}&=\begin{cases}1 & g(x_j)=a_l\\ 0 & g(x_j)\neq a_l\end{cases} \\ \mu_{i,l}&=\sum_{j=1}^{N}u_{i,j}^{m}\phi_{j,l} \end{aligned} \tag{3.9}$$

Note that $x_p \ni g(x_p)=a_k, \forall k=1,...,n$, always exists due to the assumption (3.3). (For the convenience of explanation, we will keep using the same indices i for clusters, j for patterns, l for labels in the ordinal feature and k for Δ's in the remainders of the paper as long as there is no confusion.)

Our concern now is to find $a_l, \forall l=1,...,n$, which minimize the objective function (3.8) with the constraints (3.7), using the method of Lagrange multipliers. However, it is not simple to use the method of Lagrange multipliers, with the inequality constraints $0<a_1<...<a_n<1$ in (3.7). We convert the objective function (3.8) into another form in which the inequality constraints $0<a_1<...<a_n<1$ in (3.7) become a unity constrain, so that the method of Lagrange multipliers can be easily used. For that purpose, we first introduce $\Delta_1,...,\Delta_{n+1}$ as shown in Figure 1. They are defined as (3.10).

$$\Delta_k=\begin{cases}a_k-a_{k-1} & \forall k=1,...,n\\ 1-a_n & k=n+1\end{cases} \tag{3.10}$$

Then the objective function (3.8) is converted as follows, with the new unity constraint in (3.12).

$$Q = \sum_{i=1}^{c}\sum_{l=1}^{n}\mu_{il}\left(\sum_{k=1}^{l}\Delta_k - v_i\right)^2 \tag{3.11}$$

subject to

$$\begin{aligned} &0 < \Delta_k, \forall k = 1,...,n+1 \\ &\sum_{k=1}^{n+1}\Delta_k = 1 \end{aligned} \tag{3.12}$$

Once we find the minimal solutions $\Delta_1,...,\Delta_{n+1}$ with the above constraints (3.12), satisfying the objective function (3.11), we can easily compute $a_l, \forall l = 1,...,n$, which minimize the objective function (3.8).

In the computation of the approximate solutions in Step 3 of Algorithm 2, we do not consider the positivity constraints in (3.12) in order to use the Lagrange multipliers method. Only the unity constraint is considered. Once the solutions are found, they are examined whether the positivity constraints are hold. If the positivity constraints are not hold, the new solutions are simply discarded. Experiments showed that the chance to get the solutions of non-positive values is low.

In order to find the minimal solutions $\Delta_1,...,\Delta_{n+1}$, we add the unity constraint (3.12) with the Lagrange multiplier giving us (3.13).

$$R = \sum_{i=1}^{c}\sum_{l=1}^{n}\mu_{i,l}\left(\sum_{k=1}^{l}\Delta_k - v_i\right)^2 - \lambda\left(\sum_{k=1}^{n+1}\Delta_k - 1\right) \tag{3.13}$$

The partial derivatives of R with respect to $\Delta_t, \forall t = 1,...,n$, are given by (3.14).

$$\frac{\partial R}{\partial \Delta_t} = 2\sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\left(\Delta_t + \sum_{k=1;k\neq t}^{l}\Delta_k - v_i\right) - \lambda \tag{3.14}$$

We have the linear equations with respect to Δ_t from $\frac{\partial R}{\partial \Delta_t} = 0$, and the linear equations can be arranged as follows.

$$\frac{\partial R}{\partial \Delta_t} = 0 \;\Rightarrow\; 2\sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\left(\Delta_t + \sum_{k=1;k\neq t}^{l}\Delta_k - v_i\right) - \lambda = 0 \tag{3.15}$$

Now we have $n+1$ linear equations – n equations in (3.15) and the unity constraint in (3.12). We do not solve directly these $n+1$ linear equations to get approximate solutions for $n+2$ variables – $\Delta_t, \forall t = 1,...,n+1$, and λ . We further simplify the above equations in (3.15) as follows.

$$2\sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\left(\Delta_t^{(T+1)} + \sum_{k=1;k\neq t}^{l}\Delta_k^{(T)} - v_i\right) - \lambda = 0 \tag{3.16}$$

where $\Delta_k^{(T)}, \forall k = 1,...,n+1,$ is the solution at the previous T^{th} iteration, and $\Delta_t^{(T+1)}$ is the solution at the current $(T+1)^{\text{th}}$ iteration in Algorithm 2. $\Delta_k^{(T)}, \forall k = 1,...,n+1,$ are used as given values when $\Delta_t^{(T+1)}$ is computed. The unity constraint in (3.12) is also changed slightly as follows.

$$\sum_{k=1}^{n}\Delta_k^{(T+1)} = 1 - \Delta_{n+1}^{(T)} \tag{3.17}$$

Then $\Delta_t^{(T+1)}$ is driven from (3.16) as follows.

$$\begin{aligned} &2\sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\left(\Delta_t^{(T+1)} + \sum_{k=1;k\neq t}^{l}\Delta_k^{(T)} - v_i\right) - \lambda = 0 \\ \Rightarrow\quad &\sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\Delta_t^{(T+1)} + \sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\left(\sum_{k=1;k\neq t}^{l}\Delta_k^{(T)} - v_i\right) = \frac{\lambda}{2} \\ \Rightarrow\quad &\Delta_t^{(T+1)} = \frac{\lambda}{2\xi_t} - \zeta_t^{(T)} \end{aligned} \tag{3.18}$$

where

$$\begin{aligned} \xi_t &\triangleq \sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l} \\ \zeta_t^{(T)} &\triangleq \frac{\sum_{i=1}^{c}\sum_{l=t}^{n}\mu_{i,l}\left(\sum_{k=1;k\neq t}^{l}\Delta_k^{(T)} - v_i\right)}{\xi_t} \end{aligned} \tag{3.19}$$

We can easily see $\xi_t > 0$ from the definition of $\mu_{i,l}$ in (3.9) and the positivity of $u_{i,j}$ in (2.2).

We can find λ first from the above formula (3.18) by applying the unity constraint (3.17) of $\Delta_t^{(T+1)}$'s, as follows.

$$\begin{aligned} &\Delta_t^{(T+1)} = \frac{\lambda}{2\xi_t} - \zeta_t^{(T)} \\ \Rightarrow\quad &\sum_{k=1}^{n}\left(\frac{\lambda}{2\xi_k} - \zeta_k^{(T)}\right) = 1 - \Delta_{n+1}^{(T)} \\ \Rightarrow\quad &\lambda = 2\frac{1 - \Delta_{n+1}^{(T)} + \sum_{k=1}^{n}\zeta_k^{(T)}}{\sum_{k=1}^{n}\frac{1}{\xi_k}} \end{aligned} \tag{3.20}$$

This λ is applied back into the formula (3.18) to find the new $\Delta_t^{(T+1)}$, as follows.

$$\Delta_t^{(T+1)} = \frac{\lambda}{2\xi_t} - \zeta_t^{(T)} = \frac{1}{\xi_t}\frac{1 - \Delta_{n+1}^{(T)} + \sum_{k=1}^{n}\zeta_k^{(T)}}{\sum_{k=1}^{n}\frac{1}{\xi_k}} - \zeta_t^{(T)} \tag{3.21}$$

From the new $\Delta_t^{(T+1)}, \forall t = 1,...,n$, computed in Formula (3.21), $\Delta_{n+1}^{(T+1)}$ is computed as follows.

$$\Delta_{n+1}^{(T+1)} = 1 - \sum_{k=1}^{n}\Delta_k^{(T+1)} \tag{3.22}$$

The formulas (3.21) and (3.22) give us the approximate solutions minimizing (3.11) with only the unity constraint in (3.12). Because the positivity constraints in (3.12) are not used in the computation, some of the found solutions can be equal to zero or negative. In that case, the found solutions are not used and the previous solutions are preserved. As mentioned earlier, the chance that the found solutions are non-positive is small according to the experiments.

V. Experimental Results

We used three synthetic data sets for the experiments. All the three synthetic data sets have 500 patterns of one ordinal feature and one numerical feature. The ordinal feature has five ordinal values. The three data sets have ten, five and two clusters respectively. The simulation ran ten times and averages were obtained, with the very small termination threshold $\varepsilon = 0.00001$.

First we compared the effectiveness of our algorithm (denoted AFCM) to the fuzzy c-means clustering algorithm (denoted FCM,) which uses the initial ordinal-numerical mapping. The comparison was done with the three performance measures – partition coefficient (denoted BPC) in [6], partition entropy in [7] and another measure partition index (denoted XBPI) in [8]. Table 1 shows the effectiveness of our modified fuzzy clustering algorithm. The number 10, 5 and 2 mean the number of clusters in the data sets. (Note that the more, BPC the better; the less BPE, the better; and the less XBPI, the better.)

	FCM-10	AFCM-10	FCM-5	AFCM-5	FCM-2	AFCM-2
BPC	0.61	**0.70**	0.68	**0.79**	0.935	**0.956**
BPE	0.91	**0.70**	0.64	**0.43**	0.137	**0.103**
XBPI	0.15	**0.09**	0.12	**0.07**	0.041	**0.012**

Table 1. Performance comparison results. The numbers 10, 5 and 2 mean the number of clusters in the data sets.

Next, Figures 2 and 3 show the found ordinal-numerical mappings in two different data sets of clustering numbers 10 and 5 respectively. The ordinal values in both cases started with equally distributed five numerical values, but the experimental results show that three ordinal values would be enough in the first data set of ten clusters, and two ordinal values would be enough in the second data set of five clusters. Figure 4 shows the five clusters found using FCM with the same second data set.

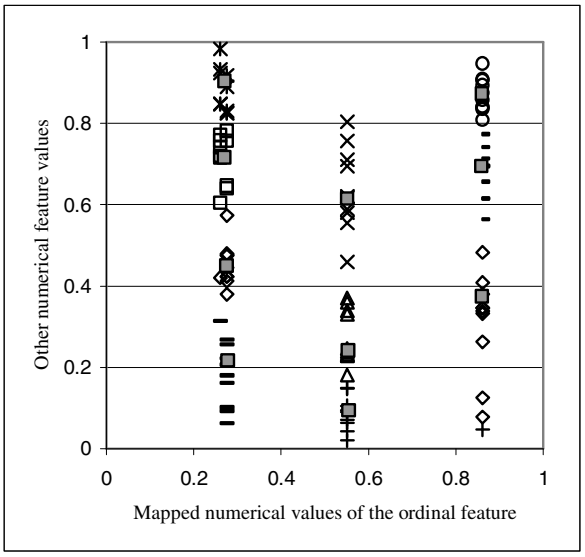

Fig. 2. Found ordinal-numerical mapping in a data set of ten clusters, using AFCM. The solid black squares are the centers of the clusters.

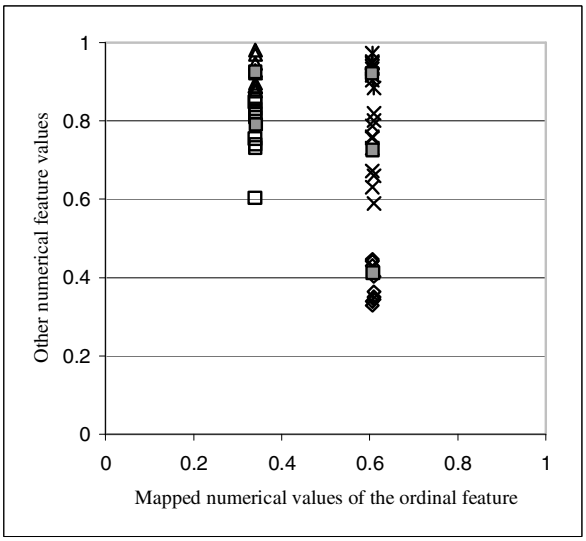

Fig. 3. Found ordinal-numerical mapping in a data set of five clusters, using AFCM. The solid black squares are the centers of the clusters.

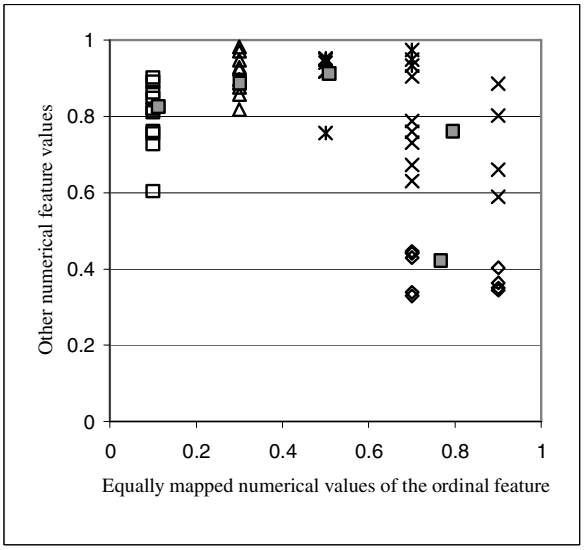

Figure 4. Clusters in the data set of five clusters, using FCM. The same data set is used in Figure 3 with AFCM to find the ordinal-numerical mapping. The solid black squares are the centers of the clusters.

VI. CONCLUSION

We presented a novel modified fuzzy c-means clustering algorithm that includes the mapping of ordinal feature values to numerical values as part of the clustering process. The mapped numerical values obtained from the ordinal feature values are used in the measurement of dissimilarity among ordinal feature values. The simulations show the modified clustering algorithm finds improved partitioning into clusters. A secondary benefit is the production of the ordinal to numerical mapping in its own right and the purpose of clustering is then to find this mapping. It is as if the inverse of the unknown real to ordinal mapping has been found. Of

course it is also possible that the underlying real to ordinal mapping is a many to one mapping. The found ordinal-numerical mapping provides additional information about the ordinal features in a given data set, which implies new updated ordinal features.

REFERENCES

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data Clustering: A Review," *ACM Computing Survey*, vol. 31, 1999.

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

[3] R. K. Brouwer, "Fuzzy Set Covering of a Set of Ordinal Attributes without Parameter Sharing," *Fuzzy Sets and Systems*, vol. 157, pp. 1775-1786, 2006.

[4] J. C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in detecting Compact Well-Separated Clusters," *Journal of Cybernetics*, vol. 3, pp. 32-57, 1973.

[5] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum Press, 1981.

[6] J. C. Bezdek, "Cluster Validity with Fuzzy Sets," *Journal of Cybern*, vol. 3, pp. 58-73, 1974.

[7] J. C. Bezdek, "Mathematical Models for Systematic and Taxonomy," presented at International Conference on Numerical Taxonomy, San Francisco, 1975.

[8] X. L. Xie and G. A. Beni, "Validity Measures for Fuzzy Clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 841-846, 1991.