

Fuzzy Ant Clustering by Centroid Positioning

Parag M. Kanade and Lawrence O. Hall
Computer Science & Engineering Dept
University of South Florida,
Tampa FL 33620
<pkanade, hall>@csee.usf.edu

Abstract—We present a swarm intelligence based algorithm for data clustering. The algorithm uses ant colony optimization principles to find good partitions of the data. In the first stage of the algorithm ants move the cluster centers in feature space. The cluster centers found by the ants are evaluated using a reformulated Fuzzy C Means criterion. In the second stage the best cluster centers found are used as the initial cluster centers for the Fuzzy C Means (FCM) algorithm. Results on 8 datasets show that the partitions found by FCM using the ant initialization are better optimized than those from randomly initialized FCM. Hard C Means was also used in the second stage and the partitions from the algorithm are better optimized than those from randomly initialized Hard C Means.

I. INTRODUCTION

The goal of any clustering algorithm is to separate the data into self-similar clusters such that some measure of the inter-cluster distance is maximized and the intra-cluster distance is minimized. Clustering algorithms can be broadly classified as Hard, Fuzzy, Possibilistic, and Probabilistic [1]. Hard clustering algorithms assign individual data points to one cluster. This model is inappropriate for real data sets in which the boundaries between the clusters may be fuzzy. Fuzzy algorithms can assign data points partially to multiple clusters. The degree of membership in the clusters depends on the closeness of the data point to the cluster center.

The drawback of clustering algorithms like FCM which are based on the hill climbing heuristic is, prior knowledge of the number of clusters in the data is required and they have significant sensitivity to cluster center initialization. In [2] the authors proposed an algorithm that finds the number of clusters and provides good cluster centers. Their algorithm uses ant colony optimization principles as inspired by [3], [4], [5]. The algorithms developed using the principles of ant colony optimization are distributed, flexible, and robust [2]. Each individual ant is a simple agent with limited power, but with co-operation and stochastic iterative behavior a colony of ants can accomplish complex tasks. The ant colony optimization (ACO) algorithms simulate the behavior of ants and are based on the interactions of the ants with each other and the environment [6], [7], [8].

In the proposed algorithm, the stochastic property of ants is simulated to obtain good cluster centers. The ants move randomly in the feature space carrying a feature of a cluster center with them. After a fixed number of iterations the cluster centers are evaluated using the reformulation of FCM which leaves out the membership matrix [1]. After the ant stage the

best cluster centers obtained are used as the initial cluster centers for the Fuzzy C Means and Hard C Means algorithms. Results on 8 datasets show the superiority of the ant initialized algorithms over the randomly initialized algorithms. The reformulations for FCM and HCM are explained in Section II. Section III explains the ant based algorithm and the parameters required for the algorithm. The datasets used and their results are explained in Section IV. Conclusions and future work are presented in Section VII.

II. REFORMULATION OF CLUSTERING CRITERIA FOR FCM AND HCM

In [1] the authors proposed a reformulation of the optimization criteria used in a couple of common clustering objective functions. The original clustering functions minimize the objective function (1) to find good clusters.

$$J_m(U, \beta) = \sum_{i=1}^c \sum_{k=1}^n U_{ik}^m D_{ik}(x_k, \beta_i) \quad (1)$$

where

U_{ik} : Membership of the k^{th} object in the i^{th} cluster

β_i : The i^{th} cluster prototype

$m \geq 1$: The degree of fuzzification

$c \geq 2$: Number of clusters

n : Number of data points

$D_{ik}(x_k, \beta_i)$: Distance of x_k from the i^{th} cluster center

The reformulation replaces the membership matrix U with the necessary conditions which are satisfied by U . The reformulated version of J_m is denoted as R_m .

For the Hard clustering case the U optimization is over a crisp membership matrix. The necessary condition for U is given in equation 2. Equation 3 gives the necessary conditions for U , for the fuzzy case. The distance $D_{ik}(x_k, \beta_i)$ is denoted as D_{ik} .

$$\begin{aligned} U_{ik} &= 0 \text{ if } D_{ik} > \min (D_{1k}, D_{2k}, D_{3k}, \dots, D_{ck}) \\ &= 1 \text{ otherwise} \end{aligned} \quad (2)$$

$$U_{ik} = \frac{\left(D_{ik}^{\frac{1}{1-m}} \right)}{\left(\sum_{j=1}^c D_{jk}^{\frac{1}{1-m}} \right)} \quad (3)$$

The reformulations for hard and fuzzy optimization functions are given in equations 4 and 5 respectively. The function R depends only on the cluster prototype and not on the U matrix, whereas J depends on both the cluster prototype and the U matrix. The U matrix for the reformulated criterion can be easily computed using equation 2 or 3.

$$R_1(\beta) = \sum_{k=1}^n \min(D_{1k}, D_{2k}, \dots, D_{ck}) \quad (4)$$

$$R_m(\beta) = \sum_{k=1}^n \left(\sum_{i=1}^c D_{ik}^{\frac{1}{1-m}} \right)^{1-m} \quad (5)$$

III. FUZZY ANTS ALGORITHM

The proposed algorithm is motivated by swarm intelligence techniques. The ants co-ordinate to move cluster centers in feature space to search for optimal cluster centers. Initially the feature values are normalized between 0 and 1. Each ant is assigned to a particular feature of a cluster in a partition. The ants never change the feature, cluster or partition assigned to them, a pictorial view is given in Fig. 1. After randomly moving the cluster centers for a fixed number of iterations, called an epoch, the quality of the partition is evaluated by using the reformulated criterion 4 or 5. If the current partition is better than any of the previous partitions' in the ant's memory then the ant remembers this partition else the ant, with a given probability goes back to a better partition or continues from the current partition. This ensures that the ants do not remember a bad partition and erase a previously known good partition. Even if the ants change *good* cluster centers to *unreasonable* cluster centers, the ants can go back to the *good* cluster centers as the ants have a finite memory in which they keep the currently best known cluster centers. There are two directions for the random movement of the ant. The positive direction is when the ant is moving in the feature space from 0 to 1, and the negative direction is when the ant is moving in the feature space from 1 to 0. If during the random movement the ant reaches the end of the feature space the ant reverses the direction. After a fixed number of epochs the ants stop.

The data is partitioned using the centroids obtained from the best known R_m value. The nearest neighbor algorithm is used for assignment to a cluster. The cluster centers so obtained are then used as the initial cluster centers for the Fuzzy C Means or the Hard C Means algorithm. The ant based algorithm is presented in Fig. 2.

The values of the parameters used in the algorithm are shown in Table I.

IV. EXPERIMENTAL RESULTS

The algorithm was applied to six data sets: the Iris Plant dataset, Glass Identification dataset, Wine Recognition dataset, MRI dataset, Multiple Sclerosis dataset and the British Towns dataset. The datasets are described in Table II. The results obtained for the datasets are shown in Table III. All results are an average from 50 random initializations. The results for the FCM and HCM are the average results from 50 random

- 1) Normalize the feature values between 0 and 1. The normalization is linear. The minimum value of a particular feature is mapped to 0 and the maximum value of the feature is mapped to 1.
- 2) Initialize the ants with random initial values and with random direction. There are two directions, positive and negative. The positive direction means the ant is moving in the feature space from 0 to 1. The negative direction means the ant is moving in the feature space from 1 to 0. Clear the initial memory. The ants are initially assigned to a particular feature within a particular cluster of a particular partition. The ants never change the feature, cluster or the partition assigned to them.
- 3) Repeat
 - 3.1 For one epoch /* One epoch is n iterations of random ant movement */
 - 3.1.1 For all ants
 - 3.1.1.1 With a probability P_{rest} the ant rests for this epoch
 - 3.1.1.2 If the ant is not resting then with a probability $P_{continue}$ the ant continues in the same direction else it changes direction
 - 3.1.1.3 With a value between D_{min} and D_{max} the ant moves in the selected direction
 - 3.2 The new R_m value is calculated using the new cluster centers
 - 3.2.1 If the partition is better than any of the old partitions in memory then the worst partition is removed from the memory and this new partition is copied to the memories of the ants making up the partition
 - 3.2.2 If the partition is not better than any of the old partitions in memory

Then

With a probability $P_{ContinueCurrent}$ the ant continues with the current partition

Else

With a probability 0.6 the ant chooses to go back to the best known partition, with a probability 0.2 the ant goes back to the second best known partition, with a probability 0.1 the ant goes to the third best known partition, with a probability 0.075 the ant goes to the fourth best known partition and with a probability 0.025 the ant goes to the worst known partition.

Until Stopping criteria

The stopping criterion is the number of epochs.

Fig. 2. Fuzzy ant clustering with centroids algorithm

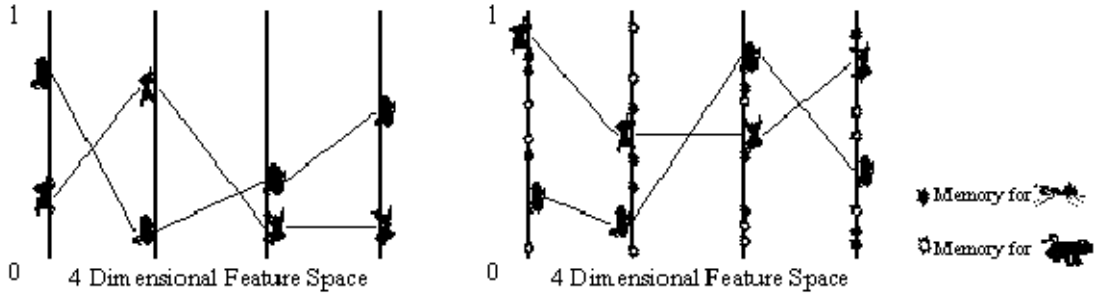


Fig. 1. Pictorial view of the algorithm in parallel co-ordinates

TABLE I
PARAMETER VALUES

Parameter	Value
Number of ants	30 Partitions ¹
Memory per ant	5
Iterations per epoch	50
Epochs	1000
P_{rest}	0.01
$P_{continue}$	0.75
$P_{ContinueCurrent}$	0.20
D_{min}	0.001
D_{max}	0.01

initializations. The glass dataset has been simplified to have just 2 classes window glass and non-window glass. The results for this modified dataset are also shown in Table 3. The age factor plays an important role in the Multiple Sclerosis dataset; the results considering the age feature and ignoring the age feature are also shown. Note, the R_m value is always less than or equal to that from randomly initialized FCM except for Glass (6 classes). Five datasets have a single extrema for the FCM algorithm. They converge to the same extrema, for all initializations tried here. This is reflected in Table III where we have the same values in columns 3 and 4 for the five datasets.

TABLE II
DATASETS

Dataset	# Examples	# Continuous Attributes	# Classes
Iris	150	4	3
Wine	178	13	3
Glass	214	9	6
MRI	65536	3	3
Multiple Sclerosis	98	5	2
British Towns	50	4	5

The parameters $Number\ of\ epochs$, D_{min} and D_{max} play an important role in determining the quality of the clusters found.

¹1 partition = number of clusters \times number of features per cluster

TABLE III
RESULTS FOR FUZZY C MEANS

Dataset	Min R_m found by Ants	R_m from FCM, ant Initialization	R_m from FCM, random Initialization
British Towns	1.6828	1.6033	1.6039
Iris	5.4271	5.2330	5.2330
Wine	33.0834	28.7158	28.7158
Glass (6 classes)	11.3827	7.2937	7.2917
Glass (2 classes)	25.8531	24.3932	24.3932
Multiple Sclerosis (with age)	6.9456	6.8538	6.8538
Multiple Sclerosis (ignoring age)	3.5704	3.5319	3.5319
MRI	311.2397	302.1302	303.289

By performing manual search, new parameters, which gave better results, were found. The values of the new parameters are shown in Table IV and the results obtained by using these modified parameters are shown in Table V. Significant improvements were observed for 3 datasets. For the British Towns dataset the average value for R_m after FCM decreased to 1.5999 from 1.6033, similarly for the Glass (6 classes) dataset the average value for R_m after FCM decreased to 7.2897 from 7.2937. This average value is better than that obtained from randomly initialized FCM. The average value for R_m after FCM for the MRI dataset decreased to 301.9198 from 302.1302.

TABLE IV
NEW PARAMETERS

Parameter	Old Value	New Value
Epochs	1000	2000
D_{min}	0.001	0.0001
D_{max}	0.01	0.001

V. HARD C MEANS

The ant algorithm was applied with the Hard C Means algorithm. The ants find the cluster centers and these centers

TABLE V
RESULTS FOR FCM OBTAINED FROM MODIFIED PARAMETERS

Dataset	Min R_m found by Ants	R_m from FCM, ant Initialization	R_m from FCM, random Initialization
British Towns	1.6051	1.5999	1.6039
Glass (6 classes)	9.2284	7.2897	7.2917
MRI	302.1188	301.9189	303.2894

are used as the initial centers for the Hard C Means algorithm. The parameter values were those shown in Table I.

TABLE VI
RESULTS FOR HARD C MEANS

Dataset	Min R_m found by Ants	R_m from HCM, ant Initialization	R_m from HCM, random Initialization
British Towns	5.5202	3.6260	3.4339
Iris	7.0055	6.9981	8.2516
Wine	52.8098	50.4573	48.9792
Glass (6 classes)	28.1317	24.3770	21.1165
Glass (2 classes)	34.2488	34.1352	36.9132
Multiple Sclerosis (with age)	10.2201	10.2016	10.3548
Multiple Sclerosis (ignoring age)	4.6406	4.6381	4.7759
MRI	433.2752	432.7499	452.384

From Table VI we see that the algorithm gives better results than randomly initialized HCM for 5 of the 8 datasets tested. Changing the parameter values can improve the results. By performing a search in the parameter space we got parameter values that resulted in better partitions. Tables VII and VIII show the variation in the results obtained by changing the number of ants per partition and epochs for the British Towns and Wine datasets. From the tables we see that as the number of epochs increase, the minimum R_m found by the ants decreases, this is to be expected because as the number of epochs increases, the ants get more time to refine the centroids found. Also as the number of ants increases, better R_m values are found. Table IX shows the results obtained for different MRI slices, the parameter values used for the MRI dataset are tabulated in Table IV and the ants per partition were 50. We can see the ant generated partitions all have lower R_m values.

VI. EXECUTION TIME

The variation of the minimum R_m found by the ants by changing the ants per partition for MR slice # 35 is shown in Table X. As the number of ants increases the minimum R_m found decreases, but at the cost of increased execution time. As the ants increase, more search space is explored and we get better R_m values.

The execution time for the British Towns and the MRI dataset is shown in Table XI. These two datasets were chosen

TABLE VII
RESULTS FOR THE BRITISH TOWNS DATASET

Ants per partition	Epochs	Min R_m found by Ants	R_m from HCM,Ant Initialization	R_m from HCM,Random Initialization
50	2000	5.3658	3.5812	3.4339
50	4000	4.5048	3.5701	3.4339
75	2000	5.1691	3.6134	3.4339
100	2000	3.0835	3.0661	3.4339

TABLE VIII
RESULTS FOR THE WINE DATASET

Ants per partition	Epochs	Min R_m found by Ants	R_m from HCM,Ant Initialization	R_m from HCM,Random Initialization
50	2000	52.8003	49.2405	48.9792
50	4000	51.0631	49.2604	48.9792
75	2000	50.1879	49.2604	48.9792
75	3000	49.9230	48.9748	48.9792
100	2000	49.6415	48.9716	48.9792
100	4000	49.2076	48.9697	48.9792

because British Towns' dataset is the smallest (in terms of number of examples) dataset and the MRI dataset is the largest dataset used in the study. The values are an average from 20000 epochs and 5 experiments for the British Towns' dataset and from 6000 epochs and 3 experiments for the MRI dataset, for FCM the values are an average from 50 runs of random initializations. One experiment consists of the ant stage and the following Fuzzy C Means or the Hard C Means stage. The time required for one epoch for the British towns dataset was more than that for the MRI dataset as there are more classes, and hence more ants, in the British towns dataset. The time required for the entire experiment was more for the MRI dataset as there are more examples in the MRI dataset. The experiments were performed on an Intel Pentium 4 2.53 GHz processor with 512 KB cache and 2 GB of main memory.

VII. CONCLUSIONS AND FUTURE WORK

The algorithm is based on relocating the cluster centroids in the feature space. The ants move the cluster centers, not

TABLE IX
RESULTS FOR THE MRI DATASET

Slice #	Min R_m found by Ants	R_m from HCM,Ant Initialization	R_m from HCM,Random Initialization
20	853.7991	851.8342	882.5732
35	919.8082	917.6636	927.6961
45	839.1622	838.0175	851.3756
46	842.5583	841.3414	847.0730
47	796.8415	795.6057	834.1028

TABLE X

VARIATION OF R_m WITH THE NUMBER OF ANTS FOR SLICE # 35 OF MRI DATASET

Ants per partition	Min R_m found by Ants	R_m from HCM,Ant Initialization	R_m from HCM,Random Initialization
50	919.8082	917.6636	927.6961
75	912.0365	909.9324	927.6961
100	910.0054	907.9996	927.6961

TABLE XI
EXECUTION TIME

Dataset	Ants per Partition	Time for one Epoch (millisecs)	Time for one Experiment (seconds)	Time for FCM (seconds)
British Towns	50	14.9645	97	0.0134
	75	22.2210	108.8380	
	100	29.7060	120.896	
MRI	50	6.7333	540.72	12.5964
	75	10.1117	811.0133	

the objects, in feature space to find a good partition for the data. The algorithm does not use the object merging criterion, which makes it independent of the threshold for merging the objects. Also, there are less controlling parameters than the previous ant based clustering algorithms [2].

Results from 8 datasets show the superiority of our algorithm over the randomly initialized FCM and HCM algorithms. For comparison purposes, Tables XII, XIII and XIV show the frequency of occurrence of different extrema for the ant initialized FCM and HCM algorithms and the randomly initialized FCM and HCM algorithms. The ant initialized FCM algorithm always finds better extrema for the MRI dataset and for the British Towns' dataset the ant initialized algorithm finds better extrema 49 out of 50 times. The ant initialized HCM algorithm finds better extrema for the Iris dataset all the time and for the Glass (2 class) dataset a majority of the time. For the different MRI slices, the ant initialized HCM algorithm finds better extrema most of the time. In [9], a Genetic programming approach was used to optimize the clustering criteria, the genetic approach for Hard C Means, found better extrema 64% of the time for the Iris dataset. The ant initialized HCM finds better extrema all the time.

The number of ants per partition is an important parameter of the algorithm. The quality of the partition improves as number of ants increase, but the improvement comes at the expense of increased execution time. Future work should focus on automatically finding the number of ants per partition and the number of clusters. In this direction, the algorithm proposed in [2] can be used to find the number of clusters.

In the algorithm the number of ants is an important parameter, and also the initial number of clusters in the data

TABLE XII

FREQUENCY OF DIFFERENT EXTREMA FROM FCM, FOR BRITISH TOWNS AND MRI DATASET

Dataset	Extrema	Frequency FCM, Ant Initialization	Frequency FCM, Random Initialization
British Towns	1.5999	49	16
	1.6037	1	18
	1.6081	0	16
MRI	301.9195	50	37
	307.1898	0	13

TABLE XIII

FREQUENCY OF DIFFERENT EXTREMA FROM HCM, FOR GLASS (2 CLASS) AND IRIS DATASET

Dataset	Extrema	Frequency HCM, Ant Initialization	Frequency HCM, Random Initialization
Glass (2 class)	34.1320	19	3
	34.1343	11	19
	34.1372	19	15
	34.1658	1	5
Iris	6.9981	50	23
	7.1386	0	14
	10.9083	0	5
	12.1437	0	8

TABLE XIV

FREQUENCY OF DIFFERENT EXTREMA FROM HCM, MRI DATASET

Slice #	Extrema	Frequency HCM, Ant Initialization	Frequency HCM, Random Initialization
20	841.3414	50	44
	889.1043	0	6
35	930.1677	45	30
	951.4871	5	15
	1003.492	0	5
45	838.0175	50	41
	912.2289	0	9
46	841.3414	50	45
	889.1043	0	5
47	795.3043	35	27
	796.2459	15	13
	970.5483	0	10

is required. Future work should focus on setting these values automatically. We compared the results from the algorithms with randomly initialized FCM and HCM. In the future results can be compared with cleverly initialized FCM and HCM algorithms. The initial placement of the ants is also random, future work might focus on giving clever initializations to the ants. The stopping criteria is currently based on the number of iterations, future work should concentrate on automatically stopping the ants based on their progress.

ACKNOWLEDGEMENT

This research partially supported by The National Institutes of Health via a bioengineering research partnership under grant number 1 R01 EB00822-01.

REFERENCES

- [1] R. J. Hathway and J. C. Bezdek, "Optimization of clustering criteria by reformulation," *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 2, pp. 241–245, May 1995.
- [2] P. M. Kanade and L. O. Hall, "Fuzzy ants as a clustering concept," *North American Fuzzy Information Processing Society, NAFIPS 2003, 22nd International Conference of the*, pp. 227–232, 2003.
- [3] W. Bin and S. Zhongzhi, "A clustering algorithm based on swarm intelligence," *Info-tech and Info-net, 2001 Proceedings. ICII 2001*, vol. 3, pp. 58–66, 2001.
- [4] W. Bin, Z. Yi, L. Shaohui, and S. Zhongzhi, "Csim: a document clustering algorithm based on swarm intelligence," *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 1, pp. 477–482, May 2002.
- [5] N. Monmarché, M. Slimane, and G. Venturini, "On improving clustering in numerical databases with artificial ants," in *5th European Conference on Artificial Life (ECAL'99), Lecture Notes in Artificial Intelligence*, D. Floreano, J. Nicoud, and F. Mondala, Eds., vol. 1674. Lausanne, Switzerland: Springer-Verlag, Sep 1999, pp. 626–635.
- [6] M. Doriga, V. Maniezzo, and A. Colomi, "Ant system: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, Feb 1996.
- [7] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence From Natural to Artificial Systems*. New York, New York: Oxford University Press, 1999.
- [8] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas, "Data mining with an ant colony optimization algorithm," *IEEE Transactions on Evolutionary Computing*, vol. 6, no. 4, pp. 321–332, Aug 2002.
- [9] L. O. Hall, I. B. Özyurt, and J. C. Bezdek, "Clustering with a genetically optimized approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 2, pp. 103–112, 1999.