International Journal of Latest Research in Engineering and Technology (IJLRET)
ISSN: 2454-5031
www.ijlret.com || Volume 02 - Issue 11 || November 2016 || PP. 01-05

FUZZY MERGING ALGORITHM WITH LYAPUNOV THEORY FOR NON-LINEAR SYSTEMS

S. Kayalvizhi[1], Gunasekar[2] and Thenmozhi[3]
[1]*Department of Sciences, Mahatma Gandhi University, Meghalaya*
[2]*Department of Computer and Information Science, Universiti Teknologi Petronas, Malaysia*
[3]*Department of electronics & Communication Engineering, NPR College of Engineering & Technology, Tamilnadu - 624 401*

Abstract: In this paper, Lyapunov Theory is combining with fuzzy merging algorithm to solve the non-linear solutions. The non-linear points are found using fuzzy c-means merging by reciprocating the distance. The Gaussian weights are used in combining the fuzzy merging with Lyapunov Theory. A vector equals the centers of two points; the weights are concentrated mostly on widely distributed region than the points in central regions. This provides the ideal solution for non-linear points by collecting the values associated with distributed regions.

Keywords: Fuzzy Merging Algorithm, Lyapunov Theory, Non-linear Points, Gaussian Weights

I. INTRODUCTION

Merging is the process of grouping the feature vectors in non-linear region into classes. This takes place in the form of a self-organizing mode.

The features vector, Q set be x and it is represented as: {x(q): q = 1,…,Q}. This feature vector has got N components that is given as: x(q) = (x1(q), …, xN(q)) has N components. The main process is considered here is merging that assigns K merges to the feature vector, Q. This is represented using minimum distance assignment principle: {c(k): k = 1, …, K}

The representatives of central points for merging or the prototype for merging is a crucial process. The feature vector, which is located at the longest distance from the central point region or distributed region i.e. at the region of merge center needs to have weights lesser than the closer points, whose weights are higher. The more distance points in the feature vector occurs due to errors in measurement or it also occurs due to the larger deviation in the process where object or the point is formed [9].

Arithmetic averaging is the simplest of all weighting method. This adds the featured vector in the merged region and in addition this takes the average of the merging prototype. This simple procedure makes it to run for a longer time in the initialization process of merging.

The main aim of the arithmetic averaging process is to give the individual same weights to the central located feature vectors. These are represented as outliers in the feature vector points. Median vectors are used instead of averaging vectors; this lowers the influence of the outlier in the non-linear pointed regions using the proposed strategy.

To provide more immunization to the outlier distance to be more representative, fuzzy weighted average is used as a representative prototype in the non-linear region and that is represented as:

$$Zn (k) = \Sigma\{q: q\in k\}\ wqkx(q)n;(1)$$

Instead of using a Boolean value as 1 or true representing merging operation or 0 or false, which does not belong to merging operation, wqk weight in Equation (1) is used to represent the membership to a merging operation in a partial range. This represents fuzzy weights and there are several methods to generate this fuzzy weight, once such method is represented here.

The fuzzy method is generated as a reciprocal of the distance vector in the non-linear points:

$$wqk = 1 / Dqk(2)$$

where, (wqk= 1 if Dqk = 0)

This is used in the process of fuzzy merging algorithm [2]. The distance between the prototype and the feature vector is large, the weights are assigned as small and when the feature vector and the prototype is small, the weights are assigned as large.

The Gaussian function is used in generating fuzzy weights and that is regarded as the most important way of merging procedure. This provides immune to the outliers and in addition provides suitable weights for the central and densely distributed points in the non-linear regions. This is used in fuzzy c-means and fuzzy merging algorithm.

In this research, a fuzzy c-means algorithm is merged over Lyapunov theory to obtain non-linear points in the distributed region. This results in the selection of best points in the distributed region rather than selecting a point in the central region. The use of Gaussian weights increases the advantage of selecting the points in the distributed region.

In this project, we implemented the fuzzy c-means (FCM) algorithm and the fuzzy Merging and merging algorithm in Java, applied the algorithms to several data sets and compared the weights of the two algorithms.

Section 2 describes the background of merging algorithms. Section 3 discusses the implementation of FCM algorithm and the results on data sets. Section 4 presents the results using the algorithm. Section 5 compares the fuzzy Gaussian weights in in the merged algorithm.

II. MERGING ALGORITHMS

The Merging procedure helps in grouping a set of sample from feature points or vectors. This is merged into K merges through appropriate similarity or dissimilarity function. This function is taken as distance from the central point of the merge to the featured vector.

A. k-means Algorithm

The k-means algorithm is suitable for assigning feature vector or point to get merged through a minimum distance assignment principle [5]. This procedure involves assignment of a new feature point x(q) to be merged with c(k). So, the distance between the point x(q) to the center of c(k) is the taken to be minimum for all K merges. The basic steps involved in k-means algorithm are described as follows:

- Put the first K feature vectors as initial centers
- Assign each sample vector to the merge with minimum distance assignment principle.
- Compute new average as new center for each merge
- If any center has changed, then go to step 2, else terminate.

The main advantage of this method is its efficacy, simplicity and self-organization. This is used as an initialization procedure in algorithms. Also, it offers certain disadvantages like the value of K has to be provided and this is a linear separation algorithm.

B. The ISODATA Algorithm

The algorithm uses the principle k-means algorithm that employs the process of elimination, split and merge operations and the process is described below [8]:

- Start with Kinit (initial number of merges) which is user-given. Assign the first Kinit samples as merge centers.
- Assign all samples to the merges by minimum distance principle.
- Eliminate merges that contain less than nmin feature vectors and reassign those vectors to other merges to yield K merges.
- Compute a new merge center as the average of all feature vectors in each merge.
- For each kth merge, compute the mean-squared error σn2(k) of each nth component xn over that merge and find the maximum σn*2(k) component mean-squared error over within merge k for over n = 1, …, N, where the index n* is for the maximum component.
- If there are not enough merges (Kinit < K/2) and this is not the last iteration, then if σmax(k) > σsplit for any merge k, split that merge into two.
- If this is an even iteration and Kinit > 2K, then compute all distances between merge centers. Merge the merges that are close than a given value.

The advantages of this algorithm include the self-organizing behavior and the flexibility of elimination in merging the values that are too small. Also, its ability to divide merges that are too dissimilar and its ability to merge merges that is sufficiently similar.

Also it offers other disadvantages like: the initialization of parameters by the users and it is not made pre-defined, the achievement of reasonable value can be achieved through small experimentation. The distance function helps in determining the ball shaped merges. The K value depends on the user parameters and mainly based on user value. The merging average is not the suitable prototype for the merging operations [9].

C. Fuzzy Merging Algorithms

The merging operation in fuzzy plays a major role in solving the problems associated with the areas of recognizing the patterns and identifying the fuzzy model. Several merging models in fuzzy is proposed and most operations are based on the criteria of distance measurement [6].

Fuzzy c-means is the widest algorithm in finding the non-linear points in computational systems. This uses a reciprocal distance function to measure the weight of fuzzy and the use of fuzzy c means is more efficient than other algorithms. This computes the Gaussian weights and a large initial prototype is used in this process. The addition of elimination, merge operations adds value of processing capability.

III. FUZZY MERGING (FCFM) ALGORITHM

A. The FCFM Algorithm

The FCFM algorithm [7] uses Gaussian weighted feature vectors to represent the prototypes. The algorithm starts with large numbers of merges and eliminates merges by distance or by size so the prototypes are much more representative. The FCFM algorithm computes fuzzy weights using Equation (10)

$$\alpha p(r) = \exp[-(xp - \mu(r))2/(2\sigma 2)(r)]/$$
$$\{\Sigma(m = 1,P)\exp[-(xm - \mu(r))2/(2\sigma 2)(r)]\}(10)$$

The FCFM algorithm does the following steps:

- Standardize the Q sample feature vectors and use a large Kinit.
- Eliminate the prototypes that are closer to other prototype than a distance threshold DThresh, which can be set by a user.
- Apply k-means as the first step to get the prototypes.
- Eliminate small merges.
- Loop in computing the fuzzy weights and MWFEV (modified weighted fuzzy expected value) for each merge to obtain the prototype and then assign all of the feature vectors to merges based on the minimum distance assignment. End the loop when the fuzzy centers do not change.
- Merge merges.

1) Initial K:

The FCFM algorithm uses a relatively large Kinit to thin out the prototypes. The default Kinit is calculated as:

$$Kinit = \max\{6N + 12\log 2Q,\ Q\}(11)$$

2) Modified Xie-Beni Validity.

The FCFM uses modified Xie-Beni validity same as in FCM.

3) Merging.

To obtain a more typical vector to represent a merge, the algorithm uses modified weighted fuzzy expected value as the prototypical value:

$$\mu(r+1) = \Sigma\{p=1, P\}\ \alpha p(r) \times p(12)$$

$\mu(r+1)$ is obtained by Picard iterations. The initial value $\mu(0)$ is the arithmetic average of the set of real vectors.

$$\alpha p(r) = \exp[-(xp - \mu(r))2/(2\sigma 2)(r)]/$$
$$\{\Sigma(m=1,P)\ \exp[-(xm - \mu(r))2/(2\sigma 2)(r)]\}(13)$$

σ^2 is the mean-square error.

The process computes the fuzzy weights and the MWFEV component wisely for each merge to obtain the merge center. It also computes the mean-square error σ^2 for each merge, and then assigns each feature vector to a merge by the minimum distance assignment principle.

For every pair of prototypes, the algorithm computes the distance between them. It finds the pair with shortest distance. If the pair meets the merge criteria, the two merges are merged into one.

B. Lyapunov theory

Lyapunov theory has emerged as a powerful method for designing controllers that are guaranteed to be stable or can maintain system operation within an intended range. In Lyapunov's “direct” or “second” method, key idea is to construct a scalar positive Lyapunov function comprising of system's state variables and then deriving a control law that makes this function's derivative negative along system trajectories. The resultant controller is guaranteed to be stable.

Lyapunov theory based Markov game fuzzy controller's action is an ‘annealed’ mix of the Fuzzy Markov game based action and Lyapunov theory based control action. The approach uses a rule-action pair visits dependent parameter $\mu k = 0, 1$ which governs the degree of hybridisation of the two mechanisms based on experiential information as:

$$\mu k = \{nk(i, ai, di)\}/\{no + nk(i, ai, di)\}$$

where, $no > 0$ is a constant and $nk(i, ai, di)$ is a count of the number of times the (rule-controller action-disturber action) tuple has been visited till instant k. Lyapunov theory based Markov game controller action is an ‘annealed’ combination of Lyapunov theory with fuzzy based action as per:

$$ai/m(sk) = (1 - \mu k)am(sk) + \mu kai(sk)$$

Under this control action ai/m(sk), inverted pendulum transitions to the next state sk+1 and generates cost:

$$ri/m(k) = øk + ø’k = 100$$

i.e., a high cost if the pendulum meets the failure condition, otherwise a quadratic cost to incentivize the pendulum to remain in the upright position.

The temporal difference error corresponding to the Markov game controller ΔQ gets modified:

$$\Delta Q = ri/m(k) + \gamma V(sk+1) - Q(sk)$$

We use reward ri/m(k) to update the candidate function C(k) on line. ΔQ is used in the q value update. We now briefly describe how the proposed controller arrives at a game theoretic stable policy. The FIS antecedent part receives current state, control action and disturber action to generate a rule firing strength vector.

FIS consequent part in generates rule specific q values. These q values are used to form a game matrix which is solved using linear programming to generate an optimal policy and optimal value.

The optimal policy (Linear programming solution) is mixed with a random policy to generate an EEP policy which in turn is used to derive Markov game control action. The values are also used in conjunction with an EEP policy to get global Q value. The Lyapunov module consists of a candidate function and a Lyapunov action generator. The Lyapunov theory based action and fuzzy based actions are combined in hybridizer to produce optimal non-linear point.

Under this action, system transitions to the next state and generates a reinforcement signal which indicates goodness or otherwise of the control action. This reinforcement signal is used to generate a TD error which is used to tune q values, thus making it adaptive. This entire process of matching current system state to FIS antecedents to generate rule firing strength vector, generating optimal control policy, rewards/cost, TD error, and q values tuning is repeated at the next state.

The sequential making procedure and associated refining of q values continues till either a goal state is reached or the system fails. The q values get progressively refined as the reinforcement learning (experiential) goes on, and the controller ultimately learns a “safe and stable” policy to counter parameter variations and disturbances.

C. Gaussian Weights

The weight of Gaussian is decided not only by the distance between the feature vector and the prototype, but also the distribution of the feature vectors in the cluster. When the feature vectors in a cluster are centrally and densely distributed around the center, the sigma value will be small. The features are close to prototype weigh much more than the farther features. The weight of a feature vector is related to the shape of the Gaussian. If a feature vector has equal distance from two prototypes, it weighs more on the more widely distributed cluster than on the centrally located cluster. Thus, Gaussian fuzzy weights are more immune to outliers and more representative than the other kind of fuzzy weights.

IV. THE RESULTS OF THE FCFM ALGORITHM

We run the FCFM on three data sets, which are iris data set, geological data and WBCD data set. The results are presented below.

For iris data, the initial number of prototypes was set to 150. The merge centers were initialized randomly. After eliminating merges by distance (0.28), K was reduced to 58. Reduce K to 21, 20, 18, 16, 14, 12 and 10 respectively by eliminating small merges with p = 1, 2, 3, 4, 5, 6 and 8. Then set iteration number of fuzzy Merging to 100. Merge with β = 0.5, 0.52, 0.54, 0.56 and 0.6, the K for each merge was 7, 4, 3, 3 and 2. Some more detailed results are in Table 1.

TABLE I. Result of the FCFM Algorithm

K	Merges	XB	σk
4	49, 34, 17, 50	0.825	0.204, 0.227, 0.134, 0.192
3	63, 37, 50	1.45	0.233, 0.228, 0.192
3	62, 38, 50	1.4	0.231, 0.229, 0.192
2	100, 50	5.82	0.323, 0.192

Table 2 shows the results on WBCD data set. The original prototypes were set to 200 and initialized using feature vectors. We started at deletion of close prototypes (d = 0.83), the prototype became 29. We then reduced the K to 14, 13, 11, 9 by eliminating small merges with p = 1, 2, 4 and 7. Set fuzzy iteration f = 40. We reduced the K to 7, 5, 4, 3, 2 by selecting β = 0.5, 0.6, 0.66, 0.7 and 0.8 in merge process.

TABLE II. Results on WBCD Data via the FCFM Algorithm

K	Merges	XB	σk
5	20, 18, 42, 14, 106	0.234	0.59, 0.83, 0.62, 0.77, 0.66
4	29, 18, 126, 27	0.292	0.59, 0.82, 0.72, 0.82
3	44, 123, 33	0.42	44, 123, 33
2	125, 75	1.082	75, 125

Table 3 presents the results on geological data set. The original prototypes were initialized using feature vectors. After deletion of close prototypes (d = 0.34), K reduced to 23. We eliminated small merges with p = 1, 2, 3, 4 and 5, K became to 14, 11, 9, 8 and 7. The fuzzy Merging iteration was set to 100. Merge the merges with β = 0.5, 0.55, 0.65, 0.85, 0.9, K reduced to 6, 5, 4, 3 and 2.

TABLE III. Results on Geological Data via the FCFM Algorithms

K	Merges	XB	σk
5	14, 8, 27, 12, 9	0.612	0.44, 0.32, 0.30, 0.40, 0.32
4	14, 10, 34, 12	0.995	0.42, 0.32, 0.45, 0.40
3	16, 21, 33	0.87	0.42, 0.60, 0.41
3	17, 53	2.39	0.47, 0.58

V. CONCLUSIONS AND FUTURE WORK

The merging algorithm with Lyapunov theory uses reciprocal distance for computing the fuzzy weights. Here, the feature vector has an equal distance from the merging central points. This weights the same when the two point merges and even if distributed region merges, there is no change in value. The two mergers are not differentiated with various distributions over the featured points. The fuzzy is made suitable to the points in the dataset. The main concentration is given to the points around the distributed regions in the space. The algorithm with merged theory helped in margining the two merges with its natural points which are merged to form larger boundaries. The use of Gaussian weights further provides more immune to the outliers. The use of Gaussian weights helps in reflecting the same values of distribution of features points in the merges. The feature vector with equi-distant point weights more on widely distributed mergers than the narrow regions.

REFERENCES

[1] E. Anderson, "The iris of the Gaspe peninsula" Bulletin American Iris Society, Vol. 59, 2-5, 1935.
[2] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York 1981.
[3] J. C. Bezdek, etc. Convergence Theory for Fuzzy c-Means: Counterexamples and Repairs, IEEE Trans. Syst., September/October 1987.
[4] Maria Colmenares & Olaf WolkenHauer, "An Introduction into Fuzzy Merging", http://www.csc.umist.ac.uk/computing/Merging.htm, July 1998, last update 03 July, 2000
[5] Marti Hearst, K-Means Merging, UCB SIMS, Fall 1998, http://www.sims.berkeley.edu/courses/is296a-3/f98/lectures/ui-bakground/sld025.htm.
[6] Uri Kroszynski and Jianjun Zhou, Fuzzy Merging Principles, Methods and Examples, IKS, December 1998
[7] Carl G. Looney: A Fuzzy Merging and Fuzzy Merging Algorithm, CS791q Class Notes, http://www.cs.unr.edu/~looney/.
[8] Carl G. Looney Pattern Recognition Using Neural Networks, Oxford University Press, N.Y., 1997.
[9] Carl G. Looney "Chapter 5. Fuzzy Merging and Merging", CS791q Class Notes, http://www.cs.unr.edu/~looney/.
[10] Ramze Rezaee M, Lelieveldt B P F and Reiber J H C, A New Merge Validity Index for the Fuzzy c-mean, Pattern Recognition Letters, (Netherlands) Mar 1998.
[11] M.J.Sabin, Convergence and Consistency of Fuzzy c-means /ISODATA Algorithms, IEEE Trans. Pattern Anal. Machine Intell., September 1987.