Available online at http://www.journalijdr.com

IJDR

International Journal of
DEVELOPMENT RESEARCH

ISSN: 2230-9926

International Journal of Development Research
Vol. 06, Issue, 05, pp. 7838-7842, May, 2016

Full Length Research Article

APPLICATIONS OF FUZZY LOGIC APPROACH IN IMAGE SEGMENTATION

Suranjan Dhar, *Arnab Acharyya and Arijit Sarker

Technique Polytechnic Institute, Panchrokhi, Sugandhya, Hooghly

ARTICLE INFO

Article History:
Received 16th February, 2016
Received in revised form
24th March, 2016
Accepted 27th April, 2016
Published online 31st May, 2016

Key Words:
Image segmentation,
Clustering, Fuzzy c-means,
Thresholding,
Edge detection.

ABSTRACT

This is a review paper on image segmentation. Basically we have tried to focus our work on image segmentation using fuzzy logic approach. In this paper we have discussed some basic concept of image segmentation and the application of fuzzy c-means algorithm in image segmentation. We have implemented this algorithm using MATLAB. This is basically the implementation part and now we are working on the efficiency respect to other fuzzy clustering algorithms which can be used for image segmentation.

INTRODUCTION

Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as super-pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics. The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristics. When applied to a stack of images, typical in medical imaging, the resulting contours after image segmentation can be used to create 3D reconstructions with the help of interpolation algorithms like marching cubes. Some of the practical applications of image segmentation are- (a) Content-based image retrieval, (b) Machine vision, (c) Medical imaging, (d) Object detection, (e) Recognition tasks, (f) Traffic control system, (g) Video surveillance.

Corresponding author: Arnab Acharyya
Technique Polytechnic Institute, Panchrokhi, Sugandhya, Hooghly

Different methods of image segmentation

Thresholding

The simplest method of image segmentation is called the thresholding method. This method is based on a clip-level (or a threshold value) to turn a gray-scale image into a binary image. There is also a balanced histogram thresholding. The key of this method is to select the threshold value (or values when multiple-levels are selected). Several popular methods are used in industry including the maximum entropy method, Otsu's method (maximum variance), and k-means clustering. Recently, methods have been developed for thresholding computed tomography (CT) images. The key idea is that, unlike Otsu's method, the thresholds are derived from the radiographs instead of the (reconstructed) image. New methods suggested the usage of multi-dimensional fuzzy rule-based non-linear thresholds. In these works decision over each pixel's membership to a segment is based on multi-dimensional rules derived from fuzzy logic and evolutionary algorithms based on image lighting environment and application.

Clustering methods

The K-means algorithm is an iterative technique that is used to partition an image into *K* clusters. The basic algorithm is

- Pick K cluster centers, either randomly or based on some heuristic
- Assign each pixel in the image to the cluster that minimizes the distance between the pixel and the cluster center
- Re-compute the cluster centers by averaging all of the pixels in the cluster
- Repeat steps 2 and 3 until convergence is attained (i.e. no pixels change clusters)

In this case, distance is the squared or absolute difference between a pixel and a cluster center. The difference is typically based on pixel color, intensity, texture, and location, or a weighted combination of these factors. *K* can be selected manually, randomly, or by a heuristic. This algorithm is guaranteed to converge, but it may not return the optimal solution. The quality of the solution depends on the initial set of clusters and the value of *K*.

Compression-based methods

Compression based methods postulate that the optimal segmentation is the one that minimizes, over all possible segmentations, the coding length of the data. The connection between these two concepts is that segmentation tries to find patterns in an image and any regularity in the image can be used to compress it. The method describes each segment by its texture and boundary shape. Each of these components is modeled by a probability distribution function and its coding length is computed as follows:

- The boundary encoding leverages the fact that regions in natural images tend to have a smooth contour. This prior is used by Huffman coding to encode the difference chain code of the contours in an image. Thus, the smoother a boundary is, the shorter coding length it attains.
- Texture is encoded by lossy compression in a way similar to minimum description length (MDL) principle, but here the length of the data given the model is approximated by the number of samples times the entropy of the model. The texture in each region is modeled by a multivariate normal distribution whose entropy has closed form expression. An interesting property of this model is that the estimated entropy bounds the true entropy of the data from above. This is because among all distributions with a given mean and covariance, normal distribution has the largest entropy. Thus, the true coding length cannot be more than what the algorithm tries to minimize.

Histogram-based methods

Histogram-based methods are very efficient compared to other image segmentation methods because they typically require only one pass through the pixels. In this technique, a histogram is computed from all of the pixels in the image, and the peaks and valleys in the histogram are used to locate the clusters in the image. Color or intensity can be used as the measure. A refinement of this technique is to recursively apply the histogram-seeking method to clusters in the image in order to divide them into smaller clusters. This operation is repeated with smaller and smaller clusters until no more clusters are formed.

One disadvantage of the histogram-seeking method is that it may be difficult to identify significant peaks and valleys in the image. Histogram-based approaches can also be quickly adapted to apply to multiple frames, while maintaining their single pass efficiency. The histogram can be done in multiple fashions when multiple frames are considered. The same approach that is taken with one frame can be applied to multiple, and after the results are merged, peaks and valleys that were previously difficult to identify are more likely to be distinguishable. The histogram can also be applied on a per-pixel basis where the resulting information is used to determine the most frequent color for the pixel location. This approach of segmentation is based on active objects and a static environment, resulting in a different type of segmentation useful in Video tracking.

Edge detection

Edge detection is a well-developed field on its own within image processing. Region boundaries and edges are closely related, since there is often a sharp adjustment in intensity at the region boundaries. Edge detection techniques have therefore been used as the base of another segmentation technique. The edges identified by edge detection are often disconnected. To segment an object from an image however, one needs closed region boundaries. The desired edges are the boundaries between such objects or spatial-taxons. Spatial-taxons are information granules, consisting of a crisp pixel region, stationed at abstraction levels within hierarchical nested scene architecture. They are similar to the Gestalt psychological designation of figure-ground, but are extended to include foreground, object groups, objects and salient object parts. Edge detection methods can be applied to the spatial-taxon region. Segmentation methods can also be applied to edges obtained from edge detectors. Lindeberg and Li developed an integrated method that segments edges into straight and curved edge segments for parts-based object recognition, based on a minimum description length (M_{DL}) criterion that was optimized by a split-and-merge-like method with candidate breakpoints obtained from complementary junction cues to obtain more likely points at which to consider partitions into different segments.

Segmentation with fuzzy logic

Clustering Method

Fuzzy clustering is the partitioning of a collection of data into fuzzy subsets or clusters based on similarities between the data [Passino and Yurkovich, 1998]. The CM, like the other methods described previously, develops a fuzzy estimation model, to predict the output given the input. The algorithm forms rules (or clusters) with training data using a nearest neighbor approach for the fuzzy system. This is demonstrated in the following example where the same training data set used. Recall that these data consist of two inputs *(n* = 2) and one output for each data-tuple. Again we employ Gaussian membership functions for the input fuzzy sets, and delta functions for the output functions. In addition, we make use of center-average de-fuzzification and product premise for developing our fuzzy model, which is given by *f(x\6).* However, for the CM we employ slightly different variables.

Passino and Yurkovich [1998] make the following recommendations on *a*:

- A small *a* provides narrow membership functions that may yield a less smooth fuzzy system, which may cause the fuzzy system mapping not to generalize well for the data points in the training set.
- Increasing the parameter *a* will result in a smoother fuzzy system mapping.

Cluster Analysis

Clustering refers to identifying the number of subclasses of *c* clusters in a data universe X comprising *n* data samples, and partitioning X into *c* clusters ($2 < c < n$). $c = 1$ denotes rejection of the hypothesis that there are clusters in the data, whereas $c = n$ constitutes the trivial case where each sample is in a "cluster" by itself. There are two kinds of c-partitions of data: hard (or crisp) and soft (or fuzzy). For numerical data, one assumes that the members of each cluster bear more mathematical similarity to each other than to members of other clusters. Two important issues to consider in this regard are how to measure the similarity between pairs of observations and how to evaluate the partitions once they are formed. One of the simplest similarity measures is distance between pairs of feature vectors in the feature space. If one can determine a suitable distance measure and compute the distance between all pairs of observations, then one may expect that the distance between points in the same cluster will be considerably less than the distance between points in different clusters. Several circumstances, however, mitigate the general utility of this approach, such as the combination of values of incompatible features, as would be the case, for example, when different features have significantly different scales. The clustering method described in this chapter defines "optimum" partitions through a global criterion function that measures the extent to which candidate partitions optimize a weighted sum of squared errors between data points and *cluster centers* in feature space. Many other clustering algorithms have been proposed for distinguishing substructure in high-dimensional data (Bezdek *et al.,* 1986). It is emphasized here that the method of clustering must be closely matched with the particular data under study for successful interpretation of substructure in the data.

Cluster Validity

In many cases, the number *c* of clusters in the data is known. In other cases, however, it may be reasonable to expect cluster substructure at more than one value of c. In this situation it is necessary to identify the value of *c* that gives the most plausible number of clusters in the data for the analysis at hand. This problem is known as *cluster validity* (Duda and Hart, 1973; Bezdek, 1981).

If the data used are labeled, there is a unique and absolute measure of cluster validity: the *c* that is given. For unlabeled data, no absolute measure of clustering validity exists. Although the importance of these differences is not known, it is clear that the features nominated should be sensitive to the phenomena of interest and not to other variations that might not matter to the applications at hand.

c –Means Clustering

Bezdek (1981) developed an extremely powerful classification method to accommodate fuzzy data. It is an extension of a method known as *c-means*, or *hard c-means*, when employed in a crisp classification sense. To introduce this method, we define a sample set of *n* data samples that we wish to classify:

$X = \{X1, X2, X3, \ldots, X\,n\}$.

Each data sample, Xi, is defined by *m* features, that is,

$Xi = \{Xi1, Xi2, Xi3, \ldots, Xi\,n\}$.

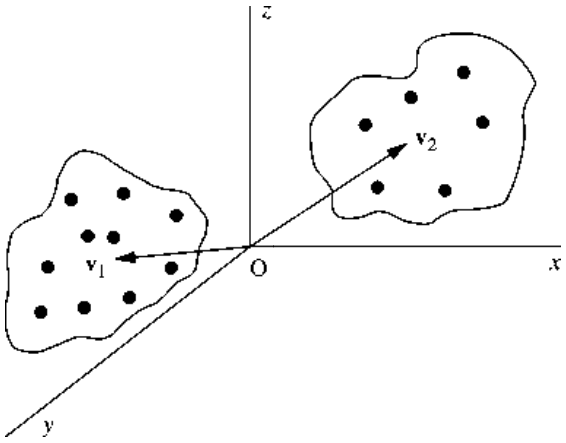

Fig 1. Cluster idea with hard c-means

where each x, in the universe X is an *m*-dimensional vector of *m* elements or *m* features. Since the *m* features all can have different units, in general, we have to normalize each of the features to a unified scale before classification. In a geometric sense, each x, is a *point* in *m* -dimensional feature space, and the universe of the data sample, X, is a *point set* with *n* elements in the sample space.

Hard *c* -Means (HCM)

HCM is used to classify data in a crisp sense. By this we mean that each data point will be assigned to one, and only one, data cluster. In this sense these clusters are also called *partitions* - that is, partitions of the data. Define a family of sets {A**,** i = 1, 2,... ,c} as a hard c-partition of X, where the following set-theoretic forms apply to the partition:

$\bigcup_{i=1}^{c} A_i = X$

$A_i \cap A_j = \phi$ all $\boldsymbol{i \neq j}$.

$\phi \subset A_i \subset X$ all i.

Again, where $X = \{x_1, x_2, x_3, \ldots, x_n\}$ is a finite set space comprising the universe of data samples, and ***c*** is the number of classes, or partitions, or clusters, into which we want to classify the data. We note the obvious-2<c<n, where $\boldsymbol{c = n}$ classes just places each data sample into its own class, and $\boldsymbol{c =}$ 1 places all data samples into the same class; neither case requires any effort in classification, and both are intrinsically uninteresting. The above equation express the fact that the set of all classes exhausts the universe of data samples and indicates that none of the classes overlap in the sense that a data sample can belong to more than one class and simply

express that a class cannot be empty and it cannot contain all the data samples.

System design and algorithm

Fuzzy c-Means Algorithm

To describe a method to determine the fuzzy **c**-partition matrix U for grouping a collection of *n* data sets into *c* classes, we define an objective function J_m for a fuzzy **c**-partition:

$J_m(U,v) = \sum_{k=1}^{n}\sum_{i=1}^{c}(\mu_{ik})^{m'}(d_{ik})^2$ ..... (4.1)
$d_{ik} = d(x_k \quad v_{i)} = [\sum_{j=1}^{m}(x_{kj-}v_{ij})^2]^{1/2}$... (4.2)

and where $\boldsymbol{\mu_{ik}}$ is the membership of the k[th] data point in the i[th] class. As with crisp classification, the function J_m can have a large number of values, the smallest one associated with the *best* clustering. Because of the large number of possible values (now infinite because of the infinite cardinality of fuzzy sets) we seek to find the best possible, or optimum, solution without resorting to an exhaustive, or expensive, search. The distance measure, d_{ik} in Equation (4.2), is again a Euclidean distance between the ith cluster center and the kth data set (data point in *m* space). A new parameter is introduced in Equation (4.1) called a *weighting parameter*, *m'* (Bezdek, 1981). This value has a range $m' \in [1,\infty)$. This parameter controls the amount of fuzziness in the classification process and is discussed shortly. Also, as before, $\boldsymbol{v_i}$ is the ith cluster center, which is described by *m* features (m coordinates) and can be arranged in vector form as before, $\mathbf{V_i}$= {v_{i1}, v_{i2}, ……, v_{im}}. Each of the cluster coordinates for each class can be calculated in a manner similar to the calculation in the crisp case:

$\boldsymbol{v_{ij}} = \sum_{k=1}^{n}\boldsymbol{\mu_{ik}^{m'}}.\boldsymbol{x_{ki}} \;/\; \sum_{k=1}^{n}\boldsymbol{\mu_{ik}^{m'}}$ ……(4.3)

where *j* is a variable on the feature space, that is, *j = 1, 2,... ,m.*

As in the crisp case, the optimum fuzzy ***c***-partition will be the smallest of the partitions described in Equation (4.1), that is,

Jm(U*, v*) = min J(U, v)…… (4.4)

with many optimization processes, the solution to Equation (4.4) cannot be guaranteed to be a global optimum, that is, the best of the best. What we seek is the best solution available within a prespecified level of accuracy. An effective algorithm for fuzzy classification, called *iterative optimization*, was proposed by Bezdek (1981). The steps in this algorithm are as follows:

- Fix *c* ($2 \le c < n$) and select a value for parameter *m'*. Initialize the partition matrix, $U^{(0)}$. Each step in this algorithm will be labeled *r*, where *r* = 0,1,2,...
- Calculate the *c* centers $\{\boldsymbol{v_i^{(r)}}\}$ for each step.
- Update the partition matrix for the *r*th step, $U^{(r)}$, as follows:

$$\mu_{ik}^{(r+1)} = \left[\sum_{j=1}^{c}\left(\frac{d_{ik}^{(r)}}{d_{jk}^{(r)}}\right)^{2/(m'-1)}\right]^{-1}, \quad \text{for } I_k = \quad \text{..... (4.5a)}$$

or $\mu_{ik}^{(r+1)} = 0$ for all classes I where $I \in I_k$ ..... (4.5b)

Where $\mathbf{I_k} = \{ i \mid 2 \le c < n;\ d_{ik}^{(r)} = 0\}$ ..... (4.6)

and $I_k = \{1, 2, \ldots, c\} - I_{k,}$ ..... (4.7)

and $\sum_{i\in I_k}\mu_{ik}^{(r+1)} = 1$ ……..... (4.8)

- If $\|U^{(r+1)} - U^{(r)}\| < \boldsymbol{\varepsilon_L}$, stop; otherwise set $\boldsymbol{r} = \boldsymbol{r} + 1$ and return to step 2.

In step 4, we compare a matrix norm || of two successive fuzzy partitions to a prescribed level of accuracy, $\boldsymbol{\varepsilon_L}$, to determine whether the solution is good enough. In step 3, there is a considerable amount of logic involved in Equations (4.5)-(4.8). Equation (4.5a) is straightforward enough, except when the variable d_{jk} is zero. Since this variable is in the denominator of a fraction, the operation is undefined mathematically, and computer calculations are abruptly halted. So the parameters I_k and I_k comprise a bookkeeping system to handle situations when some of the distance measures, d_ij, are zero, or extremely small in a computational sense. If a zero value is detected, Equation (4.5b) sets the membership for that partition value to be zero. Equations (4.6) and (4.7) describe the bookkeeping parameters I_k and I_k, respectively, for each of the classes. Equation (4.8) simply says that all the nonzero partition elements in each column of the fuzzy classification partition, U, sum to unity.

Flow Diagram

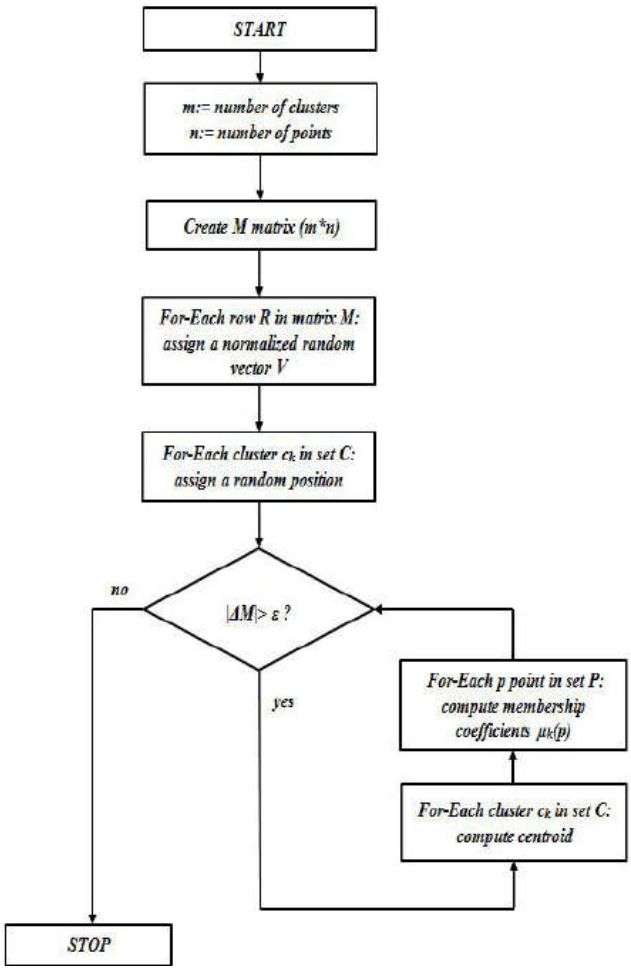

Fig. 2. Flow Chart of Fuzzy C-Mean Algorithm

RESULTS

Input to the System

Fig 3. Input Image (Cameraman)

Output from the System

Fig. 4. Input Image (Cameraman)

Acknowledgements

We are thankful to the Research & Development Cell of Technique Polytechnic Institute for their help and support. Our departmental faculty members and our departmental In-Charge, Mr. P.K.Mitra helped us a lot to carry on this project. We also express our heartiest gratitude to Prof. Dr. Bimal Kr. Datta, H.O.D, CSE of BBIT for his excellent seminar on Image Processing which motivates us for this project. Last but not the least we are thankful to our families, without their encouragement it will not be a possible task for us.

REFERENCES

Nguyen, H.P., Walker, E.A. A First Course in Fuzzy Logic. second ed. Chapman and Hall/CRC, Press, FL; 1999.

Zadeh, L.A. Fuzzy sets**.** Inf. Control. 1965;8:338–353.

Nguyen, H.P., Walker, E.A. A First Course in Fuzzy Logic. second ed. Chapman and Hall/CRC, Press, FL; 1999.

Kosko, B. Fuzzy Engineering. Prentice Hall, Englewood Cliffs, NJ; 1997.

Gene Bylinsky*, Computers That Learn By Doing*, Fortune, September 6, 1993.

Dennis Collins, *BrainMaker: Strange, Captivating, Easy to Use*, California Computer News, July, 1990.

John C. Dvorak, *Best of 1990: BrainMaker Professional, Version 1.5*, PC Magazine, January 15, 1991.

John C. Dvorak, *Inside Track*, PC Magazine, May 29, 1990

Acharyya A. and Mitra D., "Fuzzy Logic Approach towards Complex Solutions: A Review", International Journal of Advanced Computer Engineering and Communication Technology (IJACECT), ISSN (Print): 2319-2526, Volume-4, Issue-2, 2015

****