Proceedings of The National Conference
On Undergraduate Research (NCUR) 2008
Salisbury University
Salisbury, Maryland
April 10 - 12, 2008

A Methodology to Determine the Number of Clusters in Unsupervised Hyperspectral Image Classification

Axel Y. Rivera (Luis G. Jaimes)
Department of Mathematics
University of Puerto Rico at Humacao
Humacao, PR 00791

Faculty Advisor: Luis G. Jaimes

Abstract

A hyperspectral image is a collection of images using a large number of channels from the spectrum interval. These images are used to identify different natural phenomenon like variation of vegetation, minerals, etc. It is difficult to identify the components in these images. One of the principal problems in unsupervised classification is to determinate the natural clusters in which the data is distributed and their number. The focus of this paper is to provide a methodology to determine the number of clusters when unsupervised classifications over hyperspectral images were made. The method implemented for this purpose consists in using different clustering algorithms (hierarchies and partitioning) in combination with different validation methods (external and internal). These combinations gave clues about possible best number of clusters. The first experiments were done with some hyperspectral images reduced for algorithm calibration purposes. These experiments were successful using images that contain well defined and separated objects, also with images that contain objects that are overlapped by other objects.
Keywords: Clustering, Data Mining, Hyperspectral Images

1. Introduction

Hyperspectral images, that are taken most of the times from airborne scanners or satellites, are a combination of images taken in different light frequencies. These light frequencies are based in the electromagnetic spectrum chart and are used to identify things that can not be seen by human eyes. Thanks to this configuration, these images provide useful information like variation of vegetation, minerals, etc. This information can be used for moisture studies, forests ignitions and others.

These images are composed from one hundred to three hundred bands. Each band is the same picture in different light frequencies and each pixel, depending from the sensor, can represent approximately thirty by thirty square meters. Because of these properties, these images are composed a huge collection of data. A classification method is needed because the human eye can not identify objects at this level and the quantity of data is massive. The classification methods that are used often are: supervised and unsupervised (clustering) classification.

The supervised classification is bases on classifying n objects in different groups. These groups have user-defined training sets, it means, the user defines the properties of the groups where the objects are going to be classified.

Unlike the supervised classification, the unsupervised classification, or clustering, tries to group n objects in k partitions without any user-defined training sets, in other words, the objects are classified by themselves and not by user defined properties. These properties leave an opening to the problem of determining the natural clusters in which the data is distributed and their number.

2. Preprocessing

The hyperspectral images are composed by many bands. This collection of data is organized as a stack of pictures from lowest to highest wave length. After the data is organized in this form, it forms a three dimensional matrix (cube). This cube is called the hyperspectral cube.

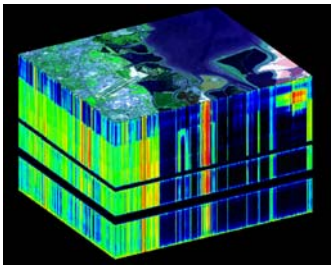

Figure 1. Example of a Hyperspectral Cube

In this cube a pixel can be seen as an array of data. This property helps to transform the cube into a matrix. In this matrix each row represents a pixel and each column represents the value of the pixel in each band. This transformation helps to apply the clustering algorithms that are going to be used.

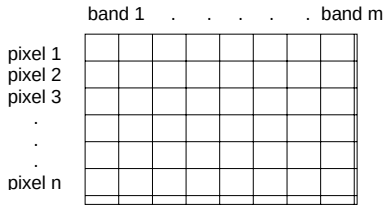

Figure 2. Transformed Matrix

3. **Algorithms**

3.1 **clustering algorithms**

A clustering algorithm is a partitioning method that tries to group a data set into subsets without any user-defined properties, it means, the data arrange by themselves. Data clustering is a common technique for statistical data analysis, which is used in many fields, including machine learning, data mining, pattern recognition, image analysis and bioinformatics. The data that is each cluster (partition) share some properties in common, most of the times the distance between the object and the cluster center.

3.1.1 *k-means*

$$K_{mn}(S) = \sum_{i=1}^{K} \sum_{X_n \in S_i} ((X_n - C_i)^2) \tag{1}$$

$\mathrm{K} = \text{clusters quantity}, \mathrm{S}_j \subset \mathrm{K}, \mathrm{xn} = \text{data vector}, \mathrm{M}_j = \text{centroid}$

The K-Means clustering method is an algorithm that groups n objects in k partitions, where $k < n$. The objective that it tries to achieve is to minimize total intra-cluster variance, or, the squared error function. The centers of the clusters in this algorithm are based on centroids. A centroid is an approximation of a subset center and is given by the average of the sum of all elements in the subset. A centroid is defined as:

$$X = \{x_1, x_2, x_3, ..., x_n\}, n > 0$$
$$C_x = \frac{x_1 + x_2 + x_3 + ... + x_n}{n} \quad (2)$$

3.1.2 *k-medoids*

$$K_{md}(R) = \frac{1}{(P)} \sum_{i=1, p \in P}^{K} \left((p - M_i(P))^2 \right) \quad (3)$$

P = data, R c P, p ∈ P, r*(p)* = medoid

K-medoids algorithm have the same purpose as K-means, this means, it groups a set of *n* objects into *k* partitions, where $k < n$ and tries to minimize total intra-cluster variance. The only difference is that K-medoids' centers are based on medoids. A medoid is the element of a *K* cluster that has the minimum distance to all elements in the same cluster. A medoid is represented as:

$$X = \{x_1, x_2, x_3, ..., x_n\}, n > 0$$
$$M_X = min(x_i)_{x_i \in X} \left\{ \frac{1}{|X|} \sum_{j=1, j \neq i}^{|X|} \{\psi(x_i, x_j)\} \right\} \quad (4)$$

3.2 **validation algorithms**

When a data has been classified, it needs to be verified. Validation algorithms are those that tend to verify how good the data classification was. These algorithms are important, especially when clustering is used, because they can tell how accurate a classification was and can provide information about the best partition number.

3.2.1 *average silhouette*

The Silhouette validation technique calculates the silhouette width for each element in the data set, average silhouette width for each subset and overall average silhouette width for a total data set. It is based on the comparison of its tightness and separation between clusters. The average silhouette width could be applied for evaluation of clustering validity and also could be used to decide how good the number of selected clusters is. In this validation method the best number of cluster is given by the greater Silhouette value. It is defined as:

$$AS = \frac{1}{|K|} \sum_{i=1}^{|K|} \left\{ \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \right\} \quad (5)$$

a(i) = average dissimilarity of i object to all objects in same cluster

b(i) = minimum of average dissimilarity of i object to all other clusters

3.2.2 davies-bouldin

Davies-Boulding validation algorithm tries to identify the clusters that are compact and well separated. The objective is to obtain clusters with minimum intra-cluster distances. Therefore, this index is minimized when looking for the best number of clusters. It is defined as:

$$DB = \frac{1}{|C|}\sum_{j=1}^{|C|} max_{j\neq i}\left\{\frac{\Delta(x_j)+\Delta(x_i)}{\delta(x_i,x_i)}\right\} \quad (6)$$

C = clustered data, X_i = cluster, (X_n) = intra - cluster distance, $\delta(X_i, X_j)$ = inter - cluster distance

3.2.3 *calinski & harabasz*

The Calinski & Harabasz's algorithm tries to find the best partition between *k* clusters. The results in this validation algorithm will be high positive numbers. The greater result is considered as the best number to divide the data. It is defined as:

$$CH = \frac{[SSB(k-1)]}{[SSW(n-k)]} \quad (7)$$

SB = between cluster squares' sums, SSW = within cluster squares' sums, k = clusters quantity, n = data quantity

3.2.3 *dunn's index*

This validation method has the same purpose as the Davies-Bouldin algorithm. It tries to identify the compact and well separated clusters. The main goal of the measure is to maximize the inter-cluster distances and minimize the intra-cluster distances. Therefore, the number of clusters that maximizes *Dn* is taken as the optimal number of the clusters. This algorithm is defined as:

$$Dn = min_{1\leq i\leq |C|}\left\{min_{1\leq j\leq |C|, j\neq i}\left\{\frac{Dist(C_i,C_j)}{max_{1\leq k\leq |C|}\{Diam(C_k)\}}\right\}\right\} \quad (8)$$

$\text{Dist}(C_i, C_j) = \min\{\delta((x, y)\}, x \in C_i, y \in C_j$, $\text{Diam}(C_i) = \max\{\gamma((x, y)\}, x, y \in C_i$

3.3 **distances implemented**

The clustering and validation methods are based on the distance between two points (δ(x,y)) function, making them one of the most important function for this work. Different distance functions can be used (*Euclidean, Manhattan, etc.*), but for this work the distance measure used was the *Euclidean* distance. The *Euclidean* distance equation is:

$$p_n \in P, q_n \in Q, n \in N$$
$$\gamma(P,Q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (9)$$

3.3.1 *intra-cluster distance*

The intra-cluster distance is the distance between the elements that are in the cluster. For this work the intra-cluster distance implemented is the *centroid diameter* distance. This distance reflects the double average distance between all of the samples and the cluster's center. It is defined as:

$$\Delta(S) = 2\left(\frac{\sum_{x \in S} \gamma(x,v)}{|S|}\right) \tag{10}$$

v = center

3.3.2 *inter-cluster distance*

The inter-cluster distance is the distance between two or more clusters. In this work the inter-cluster distance used is the *centroid linkage*. This function reflects the distance between the centers of two clusters. It is given be the equation:

$$\delta(S,T) = \gamma(vs,vt) \tag{11}$$

vs = center of S, vt = center of T

4. **Methodology**

The methodology used is the following. The first step consists in the preprocessing of the data. Here the hyperspectral cube is transformed into the matrix mentioned before for clustering.

The next step consists in finding the number of the natural clusters where the data is distributed. This process is carried out by the following methodology. The first step is to select one of the two clustering algorithms. The transformed matrix is loaded to it and is clustered using different partition quantities, most of the time from 2 to *k*. This process is performed with both clustering algorithms.

All partition quantities are passed through all four validation methods. The algorithms will give a number. The number that is repeated more between all validation methods is selected as the best number. This process is repeated *n* times. Each time that a process is repeated is called an experiment.

A final table is constructed with the results of the experiments. This table contains a percent rate of the number that appears most of the times in each experiment between *x* clustering method with *y* validation method combination. Results with high percent rate represent that the combination is good for finding the optimal number. The number that appears most of the time in this table is selected as the best number for clustering.

5. **Implementation**

Given that a hyperspectral image is a huge collection of data, it was not possible to use Matlab and R most popular clustering algorithms. MatLab was only used for the transformation of the hyperspectral cube to the matrix. A tool was developed for the rest of the work. This tool is divided in two parts: the clustering methods and the validation methods.

The clustering methods were implemented using the C Clustering Library developed by Michiel de Hoon, Seiya Imoto, Satoru Miyano at the University of Tokyo. This library is a collection of numerical routines that implement the clustering algorithms that are most commonly used, for example: K-Means, K-Medoid, Hierarchical and Self-Organizing Maps. It was developed using C programming language, but the developers implemented these routines available for use with Python programming language. When the library is used in Python, it gives the parameters to C and works with the routines. After C finishes the clustering, it returns the results to Python.

For the integration of the validation methods, it was needed to be developed in C++ programming language. This programming language is regarded as a mid-level language; it is a combination of both high-level and low-level language features. These properties permit massive routines to be calculated in less time.

To combine this whole collection of routines and libraries, a graphical user interface (GUI) was developed. This GUI was made in Python using the wxPython libraries. To make possible the interaction between both programming languages (C++ & Python), the SWIG (Simplified Wrapper and Interface Generator) tool was used. SWIG is a tool used to connect language and makes possible the usage of libraries and programs between them. This tool made possible that the routines written in C++ can be added to the GUI without any problem.

6. **Results**

The developed tool was tested with two training images. These images represent the two extreme cases. One image shows four well separated figures with ten bands. The other image contains three figures, one over the other, with ten bands. The

parameters were: clustering from two to ten clusters and the experiments were repeated five times. These images where used for algorithm calibrations and methodology tests. The tables with the percents rates mentioned before were made using the results obtained from the tests (see Table 1, Table 2, Table 3 and Table 4).

6.1 **four figures image**

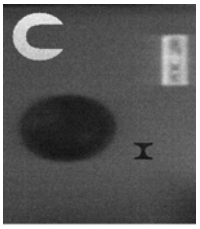

Figure 3. Four well separated objects image

6.1.1 *k-means results*

Test Number		1	2	3	4	5		Best Number	Percent (%) Rate
Silhouette		3	3	3	3	3		3	100%
DB		3	3	3	3	3		3	100%
Calinski		7	7	5	7	7		7	80%
Dunn		3	3	3	3	3		3	100%

Table 1. Four figures image's K-means results

6.1.2 *k-medoids results*

Test Number		1	2	3	4	5		Best Number	Percent (%) Rate
Silhouette		4	4	4	4	4		4	100%
DB		4	4	4	4	4		4	100%
Calinski		4	4	4	4	4		4	100%
Dunn		4	4	4	4	4		4	100%

Table 2. Four figures image's K-medoids results

6.2 **three figures image**

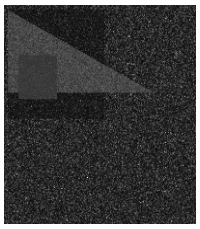

Figure 4. Three overlapped objects image

6.2.1 *k-means results*

Test Number		1	2	3	4	5		Best Number	Percent (%) Rate
Silhouette		3	3	3	3	3		3	100%
DB		3	3	3	3	3		3	100%
Calinski		8	8	8	8	8		8	100%
Dunn		3	3	3	3	3		3	100%

Table 3. Three figures image's K-means results

6.2.1 *k-medoids results*

Test Number		1	2	3	4	5		Best Number	Percent (%) Rate
Silhouette		3	3	3	3	3		3	100%
DB		3	3	3	3	3		3	100%
Calinski		7	7	8	9	7		7	60%
Dunn		3	3	3	3	3		3	100%

Table 4. Three figures image's K-medoids results

7. **Conclusion**

The clustering algorithms were tested in two extreme cases, one with well separated objects image and another with very united components image. For the well separated objects image, the results showed that K-Medoids method responded very precise in combination with all validation methods. It showed that for all cases these combinations gave the correct cluster numbers. The combination between K-Means method with all validations methods did not assert in any case.

The image with the objects that are overlapped showed that all combinations of clustering and validation methods asserted

the correct number of clusters with the exception of both clustering algorithms with Calinsky & Harabasz validation method.

These results give a recommendation that the best number for clustering a hyperspectral image can be found with the combination of K-Medoids algorithms with Dunn's Index, Average Silhouette or Davies Bouldin validation methods.

8. **Acknowledgement**

Thanks to Michiel de Hoon, Seiya Imoto and Satoru Miyano at the University of Tokyo for providing the PyCluster libraries. Also, thanks to Santiago Velasco from the University of Puerto Rico at Mayagüez for providing the Hyperspectral Image Generation Tool. This work is sponsored by URMAA (H98230-07-1-0114) and the Puerto Rico Louis Stokes Alliance for Minority Participation.

9. **References**

1. Bolshakova N. & Azuaje F., “Cluster validation techniques for genome expression data”, 2002
2. de Hoon M., et al., “The C Clustering Library”, 2005
3. Jaimes L., “Uso de técnicas de clasificación en conglomerados para describir perfiles en grandes bases de datos educativas”, 2004
4. Rousseeuw P., et al., “Clustering in an Object-Oriented Environment”, 1998
5. Velasco S. & Manian V., “Improving Hyperspectral Image Classification using Spatial Preprocessing”, 2008
6. Bolshakova N., et al., “Cluster Validity Algorithms”, http://machaon.karanagai.com/validation_algorithms.html
7. Fulton W. & Beazley D., “Simplified Wrapper and Interface Generator (SWIG)”, http://www.swig.org