Integrated Information Mining and Image Retrieval in Remote Sensing

Jiang Li[*] and Ram M. Narayanan[+]

[*]Department of Computer Science and Information Technology
Austin Peay State University
Clarksville, TN 37044
Email: lij@apsu.edu

[+]Department of Electrical Engineering
The Pennsylvania State University
University Park, PA 16802
Email: ram@ee.psu.edu

Most existing remote sensing image retrieval systems allow only simple queries based on sensor, location, and date of image capture. This approach does not permit the efficient retrieval of useful information from large image databases. This chapter presents an integrated approach to retrieving spectral and spatial patterns from remotely sensed multi- and hyperspectral images using state-of-the-art data mining and advanced database technologies. Land cover information corresponding to spectral characteristics is identified by supervised classification based on support vector machines (SVM) with automatic model selection, while textural features characterizing spatial information are extracted using Gabor wavelet coefficients. Within identified land cover categories, textural features are clustered to acquire search efficient space in an object-oriented database (OODB) with associated images stored in an image database. Interesting patterns are then retrieved using a query-by-example (QBE) approach. The evaluation of the study results using coverage and novelty measures validates the effectiveness of the information mining and image retrieval framework, which is potentially useful for applications such as agricultural and environmental monitoring.

1. Introduction

The volume of remotely sensed imagery continues to grow at an enormous rate due to advances in sensor technology for both high spatial and temporal resolution systems. As an example, NASA's Earth Observing System (EOS) is projected to receive one terabyte of image data per day when fully operational. However, most existing information systems for managing remote sensing imagery allow only simple queries based on sensor type, geographical location, and date of capture. Building an information mining system to efficiently retrieve useful hidden patterns from large remotely sensed image databases becomes a challenge.

Although significant research progress has been made in the field of data mining, which aims to extract essential information implicitly stored in large data archives [1], research in image information mining (IIM) is still in its infancy. Mining useful information from image archives is very much an interdisciplinary endeavor that draws upon expertise in image processing, database organization, pattern recognition, information retrieval, and data mining [2]. Some important research issues and system frameworks for IIM have been proposed [3], [4]. First, IIM is different from low-level image processing techniques as it deals with the extraction of useful patterns that are previously unknown from a large collection of images, referred to as an image database, whereas image processing generally focuses on extracting and understanding features within a single image. Second, some overlap exists between the concept of IIM and content-based image retrieval (CBIR) [5], [6] regarding retrieval of images relevant to user requests from image databases. CBIR is characterized by the ability of the system to retrieve relevant images based on their semantic and visual contents rather than by using atomic attributes or keywords assigned to them. IIM emphasizes the process of discovering significant and potentially useful hidden

patterns from large image databases, whereas the set of relevant images is dynamic, subjective, and even completely unknown. Therefore, an IIM system should be adaptive and process queries from the viewpoint of the user's interpretation of the image content and domain semantics.

Cromp and Campbell's work [7], which examined several image classification algorithms and argued the need for augmenting the meta-database with information on image content, stands as the original effort to extend data mining techniques into remote sensing. Alber et al [8] demonstrated the capability of image search algorithms to search large databases for multi- and hyperspectral image cubes most closely matching a particular query cube. An interactive search and analysis tool was presented and tested based on a relevance feedback approach to enhance a content-based image retrieval process. Chang [9] and Ren [10] studied target detection and image classification algorithms in hyperspectral imagery. An automatic target generation process generates a set of targets from image data in an unsupervised manner which is classified by the target classification process. Schweizer and Moura [11] exploits both the spatial and spectral correlations in hyperspectral imagery and proposed an approach based on a Gauss–Markov random field (GMRF) modeling of the clutter, which has the advantage of providing a direct parameterization of the inverse of the clutter covariance.

Recently, several remote sensing IIM research prototype systems have being developed such as Algorithm Development and Mining (Adam) System at ITSC [12], Diamond Eye System at JPL/NASA [13], Intelligent Satellite Information Mining System at DLR [14], and VisiMine System at Insightful Corporation [15]. Unlike Adam and Diamond Eye which are general data mining systems for algorithms development and distributed processing in a wide range of applications for earth science data, the integrated framework presented here is for experts or non-specialists to retrieve spectral and spatial information in remotely sensed image databases. The key component in DLR system is a hierarchical naive Bayes learning model which lacks a convenient query interface. We aim to build an adaptive system processing queries including the remote sensing domain semantics from the user's view point. VisiMine is the latest search engine for analyzing image databases designed for satellite imagery and aerial photos. Its infrastructure addresses the key scientific need for organizing and discovering information in large databases of remotely sensed images. However, it stores features such as color, texture and shapes together with raw images within the same relational database model. We discuss a different data modeling approach, which stores features in an object-oriented database to facilitate the clustering and retrieving, and stores the images in a separate image database to facilitate browsing and query.

The spectral information, characterized by land cover and land use (LCLU) classes in a classified thematic map, is regarded as one of the most important information for remote sensing image interpretation. The *a priori* knowledge about the study region, e.g. a state-wide LCLU map, allows us to identify appropriate training sites and use a supervised classification approach based on support vector machines (SVMs) rather than an unsupervised classification technique like clustering or mathematical morphology [16] – [18]. The SVM is a novel type of learning machine based on statistical learning theory introduced by Cortes, Vapnik and Burges [17], [18]. Study has been conducted addressing the SVM-based classification scheme for land cover using polarimetric synthetic aperture radar (SAR) images [19]. In [20], it has been concluded using hyperspectal AVIRIS images that the SVM outperforms the other traditional classification rules. A SVM-type classifier has been developed for automatic classification of cloud data from GOES imagery in [21] and other kernel methods for unsupervised discovery of snow, ice, clouds have been discussed in [22].

On the other hand, the spatial information, characterized by the texture features in this study, has been regarded as an important visual primitive to search through large collections of synthetic or natural visually similar patterns. However, no universally accepted mathematical definition of texture exists, and texture analysis is even more difficult in remote sensing due to the lack of a set

of descriptive terms and texture patterns within remotely sensed imagery [23]. Reed and Buf [24] present a detailed survey of various texture methods for image analysis. Although they have reported that several texture features such as co-occurrence features and transform features show roughly the same performance, recent studies using wavelet transforms for texture analysis have generally reported better accuracy [25]. Among the wavelet transform approaches, Gabor wavelets have been proved superior to represent the texture features in the natural scene and aerial photographs [26].

To obtain an efficient searching space for specific applications, the system allows user to define categories of interest (COI) consisting of mixed LCLU information. Within each COI, the extracted texture features are clustered using an optimized *k*-means clustering approach, which automatically identifies the number of clusters and optimal initial points. The clusters are stored in an object-oriented database with associated images stored in an image database. A *k*-nearest neighbor search is performed via a Query-by-Example (QBE) interface. Finally, we evaluate the system performance using the user-oriented measures which reflects the coverage and novelty of the discovered patterns.

The rest of the chapter is organized as follows: After introducing the system architecture in Section 2, we discuss land cover mapping using SVM with automatic model selection in Section 3. Section 4 describes the texture feature extraction by Gabor wavelets, and Section 5 presents texture feature clustering with optimized *k*-means algorithm. Section 6 presents and discusses the experimental results, and Section 7 contains conclusions and proposals for future work.

2. System Architecture

The functionality of an information mining system, in general, is based upon a chain of operations consisting of image collection, processing, indexing and query supported by a database management system (DBMS), and the use of the discovered information for some decision-making process. Ref. [4] distinguishes between two kinds of frameworks characterizing image information mining systems: function-driven framework and information-driven framework. We follow the function-driven framework since it is application-oriented and relatively easy to implement according to the module functionality. Fig. 1 shows the schematic diagram of the mining prototype system, which consists of three major components: image processing module, databases module, and graphical user interface (GUI).

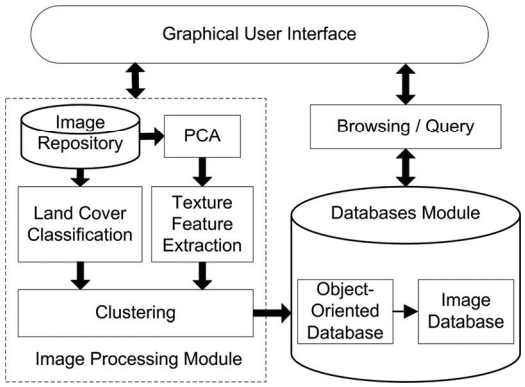


Fig. 1. Overall system architecture.

A key issue in the system architecture design is that users of remotely sensed imagery are still thinking in terms of file systems rather than DBMS, which supports indexing and query in an efficient and transparent manner with the ability to integrate data from different representations. However, traditional relational databases do not support complicated data structures storage and

query composition very well. Recently, DBMS supporting multidimensional discrete data (MDD) such as RasDaMan [27] has been developed to provide the same quality of service on raster data as is available on conventional alphanumeric data. Unfortunately, its SQL-like query language RasQL with a simple C++ application programming interface (API) is not adequate for a wide range of image information mining applications in remote sensing. In this study, we explore a different scheme consisting of two sub-database systems: an object-oriented database (OODB) and an image database (IDB). The images are stored in an IDB while the feature vectors and the pointers to the corresponding images are maintained in an OODB. OODBs provide significant advantages over traditional relational and object-relational databases. An OODB uses the language's native structures for its storage format and therefore supports an API that offers transparent storage and retrieval. This frees us to focus on designing appropriate classes and methods, rather than worrying about mappings between the program's representation and the database's representation. Another advantage is that database queries are efficiently and naturally handled, even when the data structure is complex.

3. Land Cover Classification

A set of radiometric and geometric rectified Landsat Thematic Mapper (TM) images, covering the scenes of the eastern Nebraska are utilized. All the images have been pre-calibrated and registered to UTM-13 map. We select central 4096×4096 pixels in each full TM scene, which is further divided into 16 images of 1024×1024 pixels each.

Automated classification provides an efficient and accurate way to map land cover information classes especially highly dispersed covers such as vegetation. Researchers in remote sensing keep improving classifiers for land cover and land use classification. Many parametric schemes [28], where decision boundaries are found after distribution functions are estimated from given training sets, have been presented. However, the classification results are often not satisfactory since the estimated distribution function, which is usually Gaussian, does not represent the actual distribution of the data. A natural alternative is SVM, a non-parametric scheme based on the state-of-the-art statistical learning theory, to improve the classification accuracy. SVM can also be applied to multispectral and hyperspectral images without suffering from the "curse of high dimension", or the so-called Hughes Effect. Remote sensing image classification using SVM has been reported to be computationally simple and can result in better accuracy compared to other more computationally intensive classifiers [19] – [21].

3.1. *Training and Test Data Sampling*

A region growing algorithm has been developed to sample the training and test data. The motivation for using region growing for sampling is to get a data set covering a relatively homogeneous area. The spectral angle distance (SAD) [29] is used as the similarity criterion to incorporate pixels to regions. Suppose **X** and **Y** are N dimensional spectral vectors for two pixels, then the cosine value of the spectral angle θ between the two vectors is defined as

$$\cos\theta = \frac{\mathbf{X}\cdot\mathbf{Y}}{\|\mathbf{X}\|\|\mathbf{Y}\|}, \quad 0 \le \theta \le 90 \tag{1}$$

where $\bullet$ represents inner product. SAD is invariant to unknown multiplicative scaling of spectra that may arise due to differences in illumination and angular orientation [29]. Meanwhile, we set two spatial constraints: 1) maximum area A of region R, and 2) maximum distance D from a pixel within R to the seed. The procedure is outlined in Table 1, where (r_s, c_s) is the coordinate of the seed, $\mathrm{I}(r_n, c_n)$ refers to the spectral vector at a pixel (r_n, c_n) in region R, and $\boldsymbol{\mu}_R$ is the mean vector of all pixels in R.

Table 1. Region Growing Algorithm

repeat until $A_R > A$
for each pixel p at the border of R **do**
  for all neighbors p_n at (r_n, c_n) of p **do**
    if $\|(r_n, c_n) - (r_s, c_s)\| \le D$ **and** $\arccos \frac{\mathbf{I}(r_n, c_n) \cdot \boldsymbol{\mu}_R}{|\mathbf{I}(r_n, c_n)\| \boldsymbol{\mu}_R|} \le \theta$ **then**
      add p_n to R **and** update $\boldsymbol{\mu}_R$
    end if
  end for
end for

The elements of the spectral vector, composed of the gray level in each of the six bands $[I_1, I_2, I_3, I_4, I_5, I_7]$ of TM data, are normalized to $[0, \ 1]$, i.e., $(I_j - I_{j,\min})/(I_{j,\max} - I_{j,\min})$ in each band where $I_{j,\max}$ and $I_{j,\min}$ are the maximum and minimum values in jth band respectively. As a general rule [28], more than $10N$ pixels of training data should be sampled for each class where N is the number of bands. At least 200 pixels for each class were actually sampled. After interactively selecting seeds via a GUI by referring to the Nebraska LCLU map shown in Fig. 2, we fix A, e.g. $A \ge 200$, and tune D and θ until enough data have been sampled. The region is more sensitive to the parameter θ. A large value of θ incorporates pixels of other classes while a small value of θ causes the algorithm to stop quickly with few data sampled. Therefore, we need to decide whether the regions should be merged if multiple detections are present for a single region due to a strict similarity criterion. Current implementation of the GUI-based Signature Editor allows us to easily merge or remove a region from the sampled data sets. Five data sets for each class were sampled at different sites, and we used 20% of the data for training and the remaining 80% for testing.

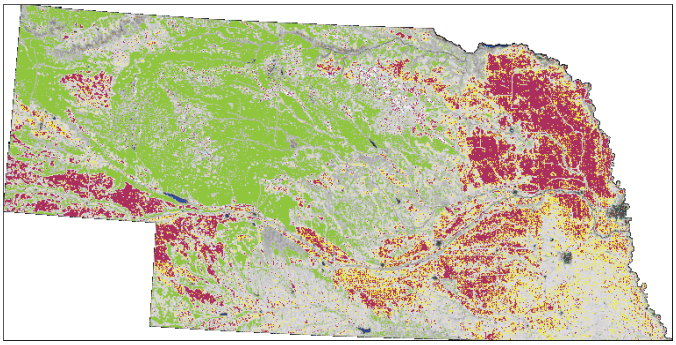


Fig. 2. Nebraska land cover and land use map.

3.2. *Theoretical Background of SVM*

A comprehensive introduction to SVM is given in [30] and briefly reviewed as follows. Given a set of examples consisting of pairs of class labels and n-dimensional feature vectors as $(y_i, \mathbf{x}_i), i = 1,...,l$, $y_i \in \{1, -1\}$, $\mathbf{x}_i \in \mathbf{R}^n$, the SVM approach places the hyperplane $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$, $\mathbf{x} \in \mathbf{R}^n$, $b \in \mathbf{R}$, so that the margin, which is defined as the distance of the closest vectors in both classes to the hyperplane, is maximized. It can be shown that the geometric margin is computed as $1/\|\mathbf{w}\|_2$ and the corresponding hyperplane is obtained by the optimization problem:

$$\min_{\mathbf{w},b} \quad < \mathbf{w} \cdot \mathbf{w} >, \quad y_i (< \mathbf{w}_i \cdot \mathbf{x}_i > + b) \ge 1, \quad i = 1,...,l \,. \tag{2}$$

This optimization problem can be translated into the following form by introducing the Lagrange multipliers $\alpha_i \geq 0$,

$$\textbf{max} \quad W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j < \mathbf{x}_i \cdot \mathbf{x}_j >, \tag{3}$$

subject to $\sum_{i=1}^{l} \alpha_i y_i = 0, \ \alpha_i \geq 0, \ i = 1, \ldots, l$.

Only a small number of multipliers α_i have nonzero values and they are associated with the so-called support vectors, which form the boundaries of the classes. The maximal margin classifier can be generalized to nonlinearly separable data via two approaches. One is to introduce a soft margin parameter C to relax the constraint that all the training vectors of a certain class lie on the same side of the optimal hyperplane. This approach is effective in case of noisy data. The other is to transform input vectors into a higher dimensional feature space by a map function φ, followed by a linear separation there. The expensive computation of inner products can be reduced significantly by using a suitable kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$. We implemented the SVM classifier using the LIBSVM library [31], and adopted Radial Basis Function (RBF) defined as the kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2) \tag{4}$$

3.3. *Automatic Model Selection*

A practical difficulty of using SVM is the selection of parameters, i.e., the soft margin parameter C and kernel parameter γ in our case. Although SVM is not sensitive to different choices of parameter γ for RBF kernel [32], it is desirable to obtain optimal values for both of these parameters given a special data set. Automatic model selection for SVM has been studied extensively in the field of machine learning. Several upper bounds of the generalization errors were defined [33], [34], and efficient algorithms [35], [36] were designed to search the best values for these parameters. Accuracy in this study is satisfactory using $C = 100$ and $\gamma = 0.5$, the optimal values computed by LOOMS (leave-one-out model selection) algorithm [36] as shown in Fig. 3.

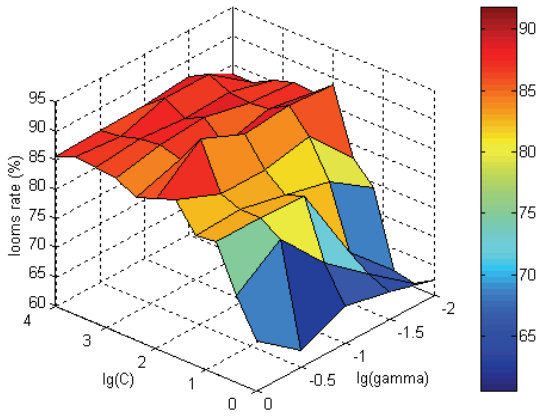


Fig. 3. Automatic model selection for the SVM classifier.

According to USGS Land Use/Land Cover map (Fig. 2), eight major land cover classes are identified in Table 2. Figs. 4 (a) and 4 (b) show an original TM image and the corresponding classified image respectively.

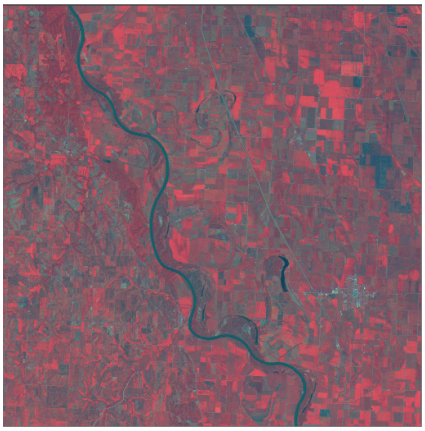


Fig. 4 (a). Original Landsat TM image.

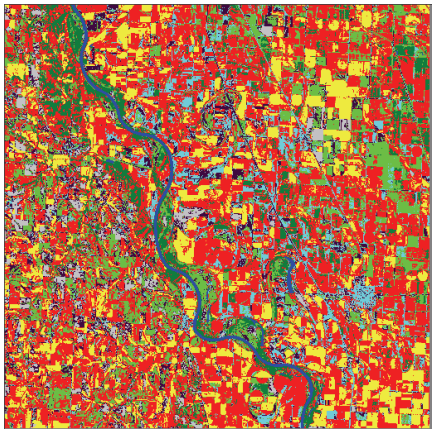


Fig. 4 (b). Classified Landsat TM image.

Table 2. Land Cover Categories

Open Water		Deciduous/Evergreen Forest	
Woody/Emergent Wetlands		Row Crops/Small Grains/Fallow	
Grasslands/Herbaceous		Bare Rock/Sand/Clay	
Pasture/Hay		Residential/Industrial	

3.4. *Accuracy Assessment*

The classification was repeated five times using a different 20% subset of the sampled data for training and using the remaining 80% for testing. We used error (confusion) matrix, a common approach, to assess the classification error and computed producer's accuracy and user's accuracy. Producer's accuracy is the probability of a reference (training) pixel being correctly classified, i.e., a measure of omission error. It is the number of pixels correctly classified as a land cover class divided by the total number of reference pixels for that land cover class. User's accuracy indicates reliability, or the probability that a pixel classified in the image is really that land cover class on the ground. It is the number of pixels correctly classified as a land cover class divided by the total number of pixels that were classified in that land cover class. Table 3 gives average classification accuracy for different classes for a TM image. Note that both the producer's and user's accuracy are high and water is almost classified perfectly.

Table 3. Classification Accuracy for Different Classes

Class Name	Producer's Accuracy	User's Accuracy
Open Water	98.7%	99.1%
Woody/Emergent Wetlands	94.6%	92.9%
Grasslands/Herbaceous	89.4%	91.2%
Pasture/Hay	93.2%	95.3%
Deciduous/Evergreen Forest	91.4%	90.5%
Row Crops/Small Grains/Fallow	88.2%	87.4%
Bare Rock/Sand/Clay	92.5%	89.7%
Residential/Industrial	86.4%	84.2%

4. Texture Feature Extraction

To extract the texture features, we first perform the principal component analysis (PCA) on each 1024 × 1024 pixels image. Then, we decomposed the first component image into 64 regions (tiles) of 128 × 128 pixels each, and computed a texture feature vector represented by Gabor wavelets for each region. Smaller region size does not cover sufficient spatial/texture information to characterize land cover types, while a large region size will involve too much information from other land-use types. Current implementation of the texture feature extraction toolbox allows user to dynamically select the size of region, e.g., 64 × 64, 128 × 128, etc.

4.1. *Principal Component Analysis*

PCA has been used in remote sensing for different purposes. A comprehensive summary of different applications of PCA, including correlation analysis of TM images for effective feature recognition and change detection with multi-temporal images, is presented in [28]. PCA is a coordinate transformation typically used to remove the correlation contained within the multi-band imagery by creating a new set of components, which are often more interpretable than the original images. PCA images thus generated are uncorrelated and ordered by decreasing variance. The covariance matrix of the transformed data is a diagonal matrix of which the elements are composed of the eigenvalues.

The first component image has the maximum signal-to-noise ratio and the largest percentage of the total variance as shown in Fig. 5. Each subsequent component contains the maximum variance for any axes orthogonal to the previous components.

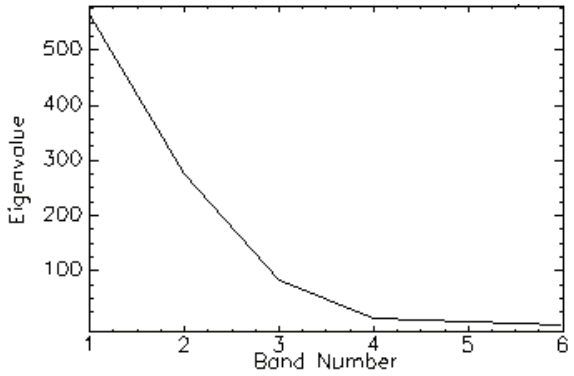


Fig. 5. Eigenvalues of principal components.

4.2. *Texture Feature Extraction*

Gabor texture feature extraction scheme has been proposed in [26] and Gabor wavelets provide the best overall performance compared with other multiresolution texture features using the Brodatz texture database. The experimental results on large aerial photographs also indicate that Gabor wavelets give good pattern retrieval accuracy [26]. A two-dimensional Gabor function $g(x, y)$ is a Gaussian function modulated by a complex sinusoid [38]:

$$g(x,y) = \left(\frac{1}{2\pi\sigma_x\sigma_y}\right)\exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)+2\pi j\omega\right]. \tag{5}$$

A class of self-similar functions, referred to as Gabor wavelets (filters), can be obtained by appropriate dilations and rotations of the mother wavelet $g(x, y)$ through the generation function:

$$g_{mn}(x,y) = a^{-m}g(x',y'),\ a>1,\ m,n = \text{integer}$$
$$x' = a^{-m}(x\cos\theta + y\sin\theta),\ y' = a^{-m}(-x\sin\theta + y\cos\theta), \tag{6}$$

where $\theta = n\pi / K$, $n = 0,1,\ldots,S-1$. S is the number of scales in the multiresolution decomposition and K is the number of orientations. The scale factor a^{-m} normalizes the filter responses, i.e. ensures that the energy is independent of m. Gabor filters are considered as orientation and scale tunable edge and line detectors. The statistics of the filtered outputs can be used to characterize the underlying texture information.

Given an image $I(x,y)$, its Gabor wavelet transform coefficients, i.e., the outputs of Gabor filters for each pixel, is defined as

$$w_{mn}(x,y) = \iint I(x_1,y_1) g^*_{mn}(x-x_1, y-y_1)\,dx_1 dy_1 \,, \tag{7}$$

where $*$ indicates the complex conjugate. We use the mean and the standard deviation of the magnitude of the transform coefficients to represent the texture information of the image for clustering and retrieval purposes:

$$\mu_{mn}(x,y) = \iint |w_{mn}(x,y)|\,dxdy \,, \tag{8}$$

$$\sigma_{mn}(x,y) = \sqrt{\iint \left(|w_{mn}(x,y)| - \mu_{mn}\right)^2 dxdy} \,. \tag{9}$$

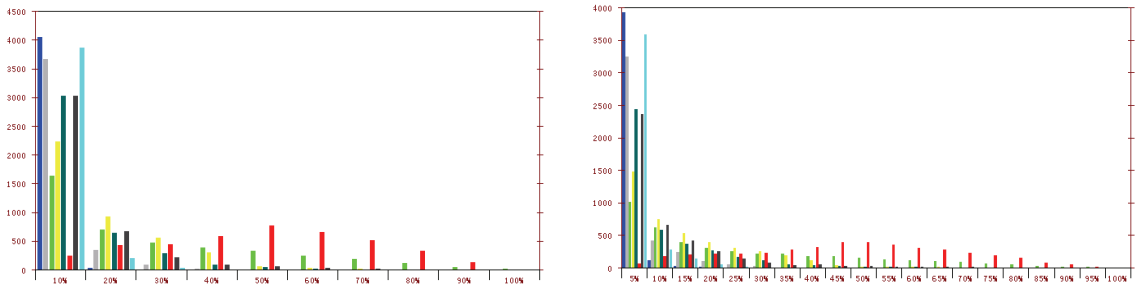
A feature vector is constructed using μ_{mn} and σ_{mn} as feature components. Based on the experimental results reported in [26], we use four scales $S = 4$ and six orientations $K = 6$, resulting in a feature vector

$$\mathbf{F} = \left[\mu_{00}\sigma_{00}\mu_{01}\sigma_{01}\cdots\mu_{35}\sigma_{35}\right]. \tag{10}$$

5. Category-Based Clustering

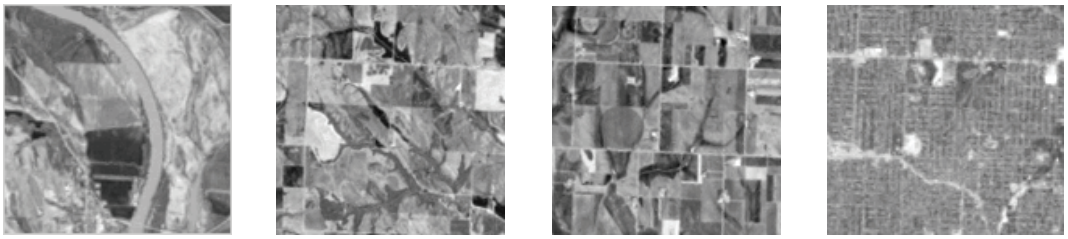
The major difficulty for texture analysis in remote sensing is that no standard texture categories exist since the visual effects of texture features depend on the spatial resolution of the image data as well as the land cover characteristics of interest in specific applications [39]. In this study, to reduce the searching space of similar texture patterns, each region is first assigned to one or more categories in terms of the significant LCLU classes. Conceptually, suppose K is the number of LCLU classes, $\binom{1}{K} + \binom{2}{K} + \cdots + \binom{K}{K} = 2^K - 1$ categories exist if we consider all combinations of land cover characteristics. This number increases exponentially and becomes prohibitively high when the number of LCLU classes becomes large. For example, 20 classes are defined in LCLU map of Nebraska, which will result in millions of categories. Fortunately, not every combination is interesting in practice, and actually not all of the LCLU classes are present in every region. A simple but efficient approach is used to help user quickly build the categories of interest (COI).

First, the distribution (i.e., the percentage) of land cover classes in each region is computed. Then we plot the histogram (Fig. 6) that represents the number of regions containing different percentage range, e.g., [0, 10%), [10%, 20%) …, of each land cover class. Thus the user can define COI by specifying the (combined) percentage of land cover classes within a region. For example, based on the eight identified primary land cover classes, we defined four experimental categories: 1) water/wetlands/pasture ($\geq 5\,\%$); 2) grasslands/forest ($\geq 10\,\%$); 3) crops/pasture ($\geq 25\,\%$); 4) residential/rock/grasslands ($\geq 30\,\%$). Fig. 7 shows some representative regions in each category. An important criterion for building categories this way is to avoid significant overlap between different categories while encompassing most of the regions in the union of all the categories.



(a) 10% bin (b) 5% bin

Fig. 6 Histogram of the land cover and land use classes in the database.



water/wetlands/pasture grasslands/forest crops/pasture residential/rock/grasslands

Fig. 7. Examples in the texture category based on land cover classes

5.1. *Optimized k-means clustering*

Within each subspace divided by COI, an optimized *k*-means clustering algorithm is developed to organize the features into clusters as illustrated in Fig. 8. The major reason for clustering is to facilitate the query discussed in the next section. *k*-means, which has been studied extensively, is one of the most popular clustering algorithms applied in pattern recognition due to its properties of local-minimum convergence and implementation simplicity. However, *k*-means suffers some intrinsic deficiencies that impede its application to data mining. First, it is very sensitive to initial starting conditions, i.e., it is fully deterministic given the randomly or arbitrarily chosen initial centers. Second, the number of clusters has to be provided as a parameter, which assumes *a priori* knowledge about the data is available. Third, computation is expensive as it requires multiple data scans to achieve convergence. Although a universal solution does not exist, various approaches have been proposed as partial remedies. Speed is greatly improved by embedding the data set in a multiresolution *kd*-tree and storing sufficient statistics at its nodes [40]. X-mean [41] can quickly estimate the number of clusters and scales better than *k*-means. The number of clusters can be automatically determined via a cluster validity measure [42] based on intra-cluster and inter-cluster distance. Bradley and Fayyad [43] discuss an approach to refine the selection of starting centers through repeated sub-sampling and smoothing. An empirical comparison of several initialization methods for the *k*-means algorithm can be found in [44].

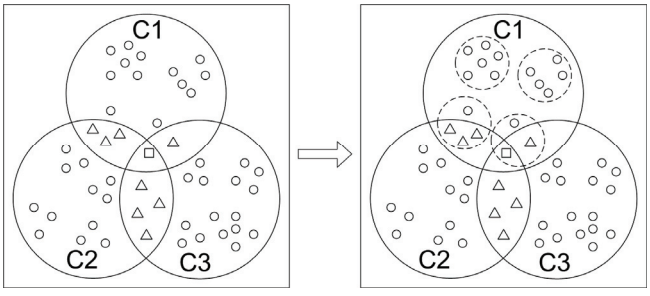


Fig. 8. Subspace partition and category-based clustering.

We combine the cluster validation measure with the initial centers refinement algorithm to form an integrated *k*-means clustering optimization procedure. The detailed discussion of the cluster validity measure is given in [42] and briefly summarized as follows. Suppose N is the total number of samples, K is the number of clusters, and $\mathbf{m}_i$ is the center of cluster C_i, we want to minimize the intra-cluster distance, i.e., the average of the sum of distance between each point and its cluster center

$$M_{intra} = \frac{1}{N}\sum_{i=1}^{K}\sum_{\mathbf{x}\in C_i}\left\|\mathbf{x}-\mathbf{m}_i\right\|^2 \tag{11}$$

Meanwhile, we want to maximize the inter-cluster distance, i.e., the minimum distance between any two cluster centers

$$M_{inter} = \mathbf{min}\left(\left\|\mathbf{m}_i-\mathbf{m}_j\right\|^2\right),\quad i=1,2,\ldots,K-1,\quad j=i+1,\ldots K \tag{12}$$

The validity measure is then defined as the ratio of these two distances [40]

$$Validity = \frac{M_{intra}}{M_{inter}} \tag{13}$$

The initial centers refinement algorithm is given in [43], which is summarized as follows:

(1) Randomly choose J small sub-samples of the data, S_i, $i=1,2,\ldots J$.

(2) Each set of sub-samples is clustered via a modified *k*-means clustering producing solutions CM_i, which forms the set CM, i.e., estimates of the true cluster centers. The modified *k*-means checks the solution at termination for empty clusters and sets the initial estimates of the empty cluster centroids to data elements that are farthest from their assigned cluster center. Then a standard *k*-means is called to re-cluster these sub-samples.

(3) CM is initialized by each of the solution CM_i and clustered via a standard *k*-means producing corresponding solution FM_i.

(4) The refined initial points are chosen as FM_i having minimal distortion over the set CM. The distortion is computed as the sum of square distances of each data point to its nearest mean.

The combined optimization procedure is described as follows:

(1) Assume K_{max} is the upper limit on the number of clusters, an iterative process tries different value of K that satisfies $2 \le K \le K_{max}$, and the clustering with a minimum value for the validity measure gives the optimal K_{opt}.

(2) Run the cluster validation algorithm on each sub-sample set used by the refinement routine, and take the average of all the candidates' $K_{opt,i}$ s as the optimal K_{opt}.

(3) The optimal value of K_{opt} is then used as one of the input parameters of the refinement algorithm to find the optimized the starting centers.

5.2. *Database Organization*

Each cluster is stored as an object with the properties of cluster identifier, cluster center, category identifier, and identifiers of all the regions within the cluster in an OODB. The corresponding raster data of the regions are stored in an IDB. The image database was constructed by incorporating the BLOB (object) into a relational model implemented using Microsoft Access as shown in Fig. 9 (a). The specification of each table is as follows:

Image Table: Storing the compressed images in JPEG format with metadata.
PCA1Image Table: Referring to Image Table and storing PCA component 1 images.
ClassMap Table: Referring to Image Table and storing classified images (LCLU maps).
Region Table: Referring to Image Table and storing the regions.
PCA1ImgRgn Table: Referring to PCA1Image Table and storing the PCA regions.

ClsMapRgn Table: Referring to ClassMap Table and storing the regions of the LCLU maps.
Sensor Table: Storing the information about the sensors, e.g., Landsat TM.
LCLUClass Table: Storing the definitions of LCLU classes.

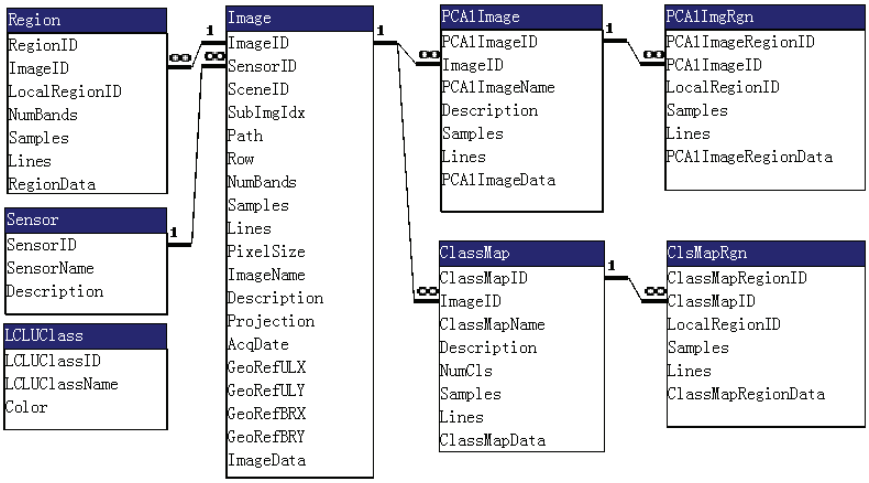


Fig. 9 (a). Image database structure.

The object-oriented programming languages in use today (most notably C++ and Java) are a direct outgrowth of the object-oriented paradigm. The object-oriented data model is an extension of object-oriented programming. Objects have entered the database world in two ways: (1) a pure object-oriented DBMS is based solely on the object-oriented data model, or (2) a hybrid (post-relational) DBMS is primarily relational but stores objects in relations.

There are a number of industrial-strength pure object-oriented DBMSs available today, some of which are being used in mission-critical applications. We adopted PSE Pro for C++ to implement the OODB. PSE Pro is a single-user, multithreaded version of ObjectStore that is perfect for transaction-oriented embedded applications. PSE Pro for C++ provides easily local data management capabilities to C++ applications by using a high-performance, transaction-oriented, persistent object storage engine.

The entity objects used by object-oriented programs are directly analogous to database entities used by pure object-oriented databases, with one major difference: Program objects disappear once a program stops running; database objects must exist. The idea that an object continues to exist once the program that created it has finished running is know as persistence. Most pure object-oriented DBMSs today are based on the concept of persistent objects and use class declarations very similar to those used by object-oriented programming languages.

Those class declarations indicate that objects created from the classes are persistent and in some way indicate relationships between objects. An object-oriented database design can be modeled using an entity-relationship (ER) diagram, just like a relationship database design. However, the modeling techniques must include the ability to represent classes and the added class relationships that object orientation provides.

Several data modeling techniques have been designed specifically for object-oriented scenarios such as OMT (Rombaugh) Notation, Booch Notation, and Unified Modeling Language (UML). We adopted UML since it approaches a standard for diagramming data models and other elements of a system design in an object-oriented environment. UML combines many of the elements of other diagramming methods and includes provisions that the others lack. Basically, UML is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct, and document the artifacts of a software-intensive system.

Fig. 9 (b) shows the OODB structure represented by a simplified UML model that shows only the attributes without operations. The specification of each class is as follows:

CImage: Base Class for general multi-spectral image modeling, including the attributes such as Image Id, Path, Row, Acquisition Date, Num of Bands, etc.

CPCA1Image: Inherited from CImage for first PCA component modeling, including attributes of CImage with private attributes: PCA1 Image Id and PCA1 Image Name.

CClassMap: Inherited from CImage for LCLU image (map) modeling, including every attribute of CImage with private attributes: Map Id and Map Name.

CRegion: Inherited from CImage for region modeling, including every attribute of CImage with private attributes: Region Id, Statistical Features, etc.

CPCA1ImageRegion: Inherited from both CPCA1Image and CRegion for first PCA component region modeling, including all the attributes of CPCA1Image and CRegion with private attributes: PCA1 Region Id, Texture Features, and a collection of Cluster Ids.

CClassMapRegion: Multi-inherited from CClassMap and CRegion for first PCA component region modeling, including attributes of CClassMap and CRegion with private attributes: Map Region Id and Statistical Features (histogram of LCLU classes).

CStatFeature: Modeling the statistical features of the regions.

CTextFeature: Modeling the texture, including Scale, Orientation, and Gabor Feature Vector.

CTextureCategory: Modeling the categories based on the combination of LCLU classes, including each LCLU class and corresponding histogram.

CSpectralClass: Modeling the LCLU classes such as Water, Crops, Forest, etc.

CTextFeatureCluster: Modeling the clusters of the texture features, including Category Id, Number of Clusters, Cluster Centroid, and a collection of (PCA1) Region Ids with corresponding Similarity Values of each pair.

CSensor: Modeling the Sensor information such as Landsat TM, Landsat MSS, etc.

stcGaborFeat: Modeling the Mean and Variance of Gabor Wavelet coefficients.

stcDate: Structure for modeling the Date information.

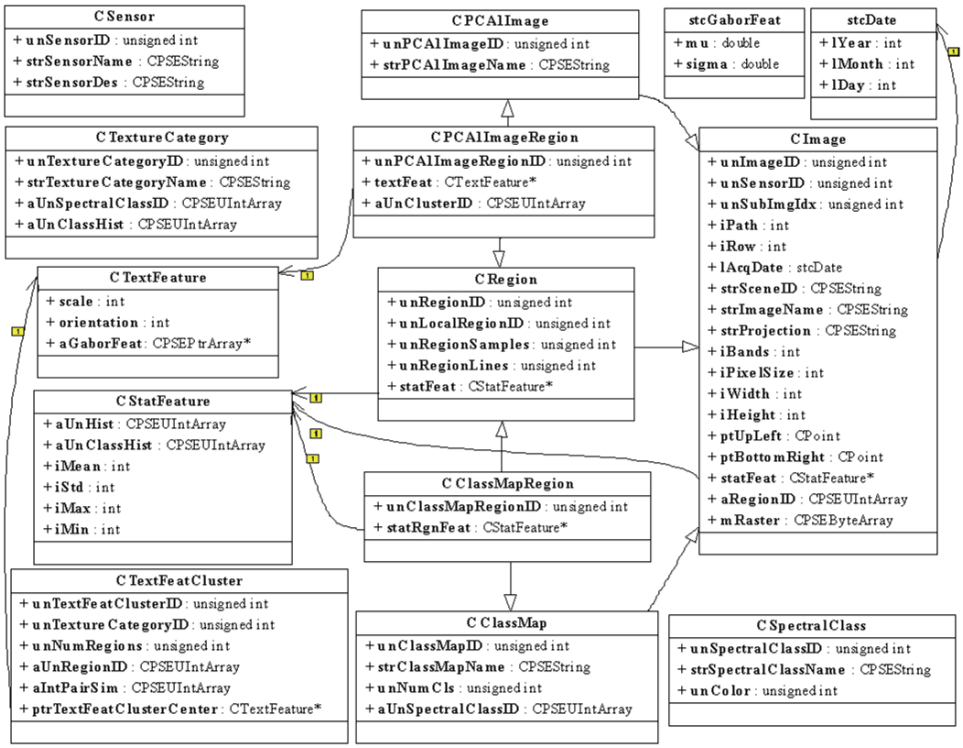


Fig. 9 (b). Object-oriented database structure.

6. Experimental Results

The prototype system has been implemented on Windows platform. The image processing module, indexing and querying module, and GUI module were developed using Microsoft Visual C++. A state-of-the-art Tree-List style interface was designed to help user quickly navigate both the OODB and the IDB as shown in Fig. 10.

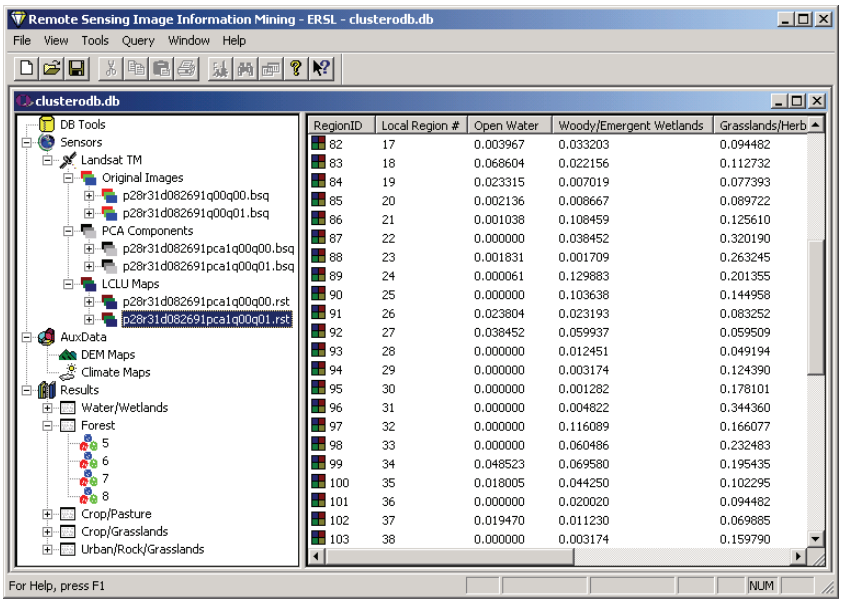


Fig. 10. The tree-list style user interface for database browsing.

6.1. *Query Processing*

The texture patterns constrained within each land cover category can be efficiently retrieved using a Query-by-Example (QBE) approach. QBE is a method of query creation that allows the user to search for similar patterns based on a selected example. Fig. 11 shows a typical query generation snapshot. A user can select any region of interest while examining the distribution of the spectral classes within the region by referring to the corresponding land cover map.

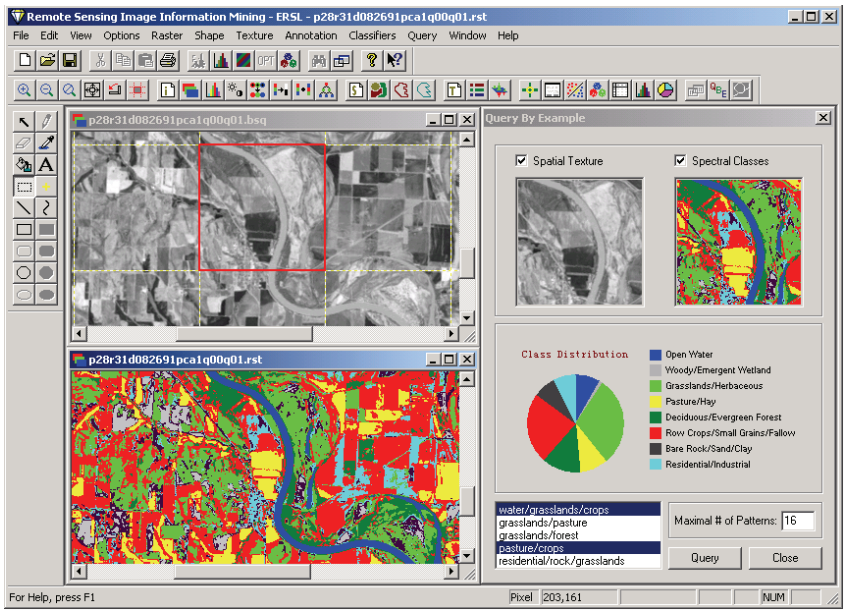


Fig. 11. Query generation by Query-by-Example.

For example, the user looks for similar patterns containing mixed crops/pasture and water classes, and sets the maximum number of patterns returned equals to sixteen as shown in Fig. 12 (a). The retrieved patterns are shown Fig. 12 (b), in which the patterns are ranked according to their distance measures. The content-based mining process therefore answers the question like "Find all regions similar to this query region that has agricultural lands around a river." Fig. 13 (a) and (b) show query pattern of crops/grasslands and corresponding results.

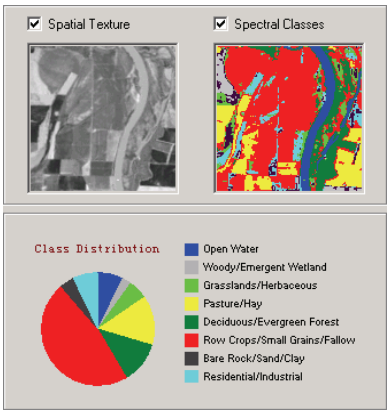


Fig. 12 (a). Query example generation of a river scene.

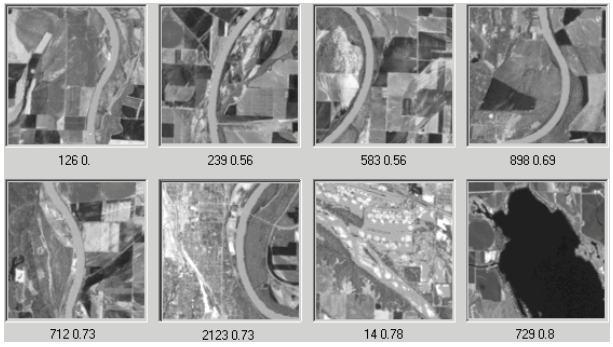


Fig 12 (b). Similar patterns shown in a ranked order.

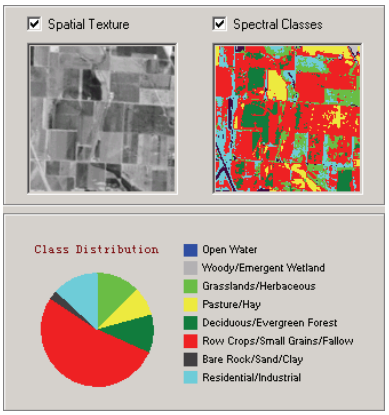


Fig. 13 (a). Query example generation of a crop scene.

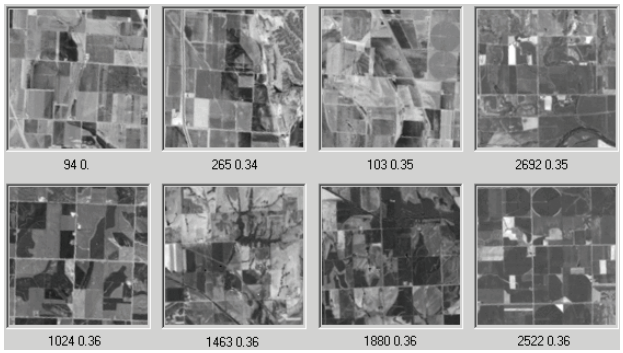


Fig 13 (b). Similar patterns shown in a ranked order.

Generally, there exist two modes in the searching process: batch (browse) mode and on-line (query) mode. In the batch mode as illustrated in Fig. 14 (a), the query example has already been indexed in the database, i.e., it is one of the patterns that exist in the clusters. Therefore, the system simply returns the *k*-nearest neighbors around the query example within a specific cluster ordered by a similarity measure based on a predefined distance, e.g. the Euclidean distance in this study. Consider the query example q and a pattern p in the database, and let $\mathbf{F}_q = \left[\mu_{00}^q \sigma_{00}^q \mu_{01}^q \sigma_{01}^q \cdots \mu_{35}^q \sigma_{35}^q\right]$ and $\mathbf{F}_p = \left[\mu_{00}^p \sigma_{00}^p \mu_{01}^p \sigma_{01}^p \cdots \mu_{35}^p \sigma_{35}^p\right]$ represent the corresponding feature vectors. Then, the distance between the two patterns in the feature space is defined as

$$d(q,p) = \left\|\mathbf{F}_q - \mathbf{F}_p\right\|^2 = \sqrt{\sum_{i,j}\left[(\mu_{ij}^q - \mu_{ij}^p)^2 + (\sigma_{ij}^q - \sigma_{ij}^p)^2\right]}. \qquad (14)$$

In the on-line mode, a new image is processed to generate texture features and land cover map. Then the user locates a new pattern that has not been clustered in the database, and the *k*-nearest neighbors of the nearest cluster center will be returned as shown in Fig. 14 (b). The new image can also be ingested into the system by the following procedure. First, each region is categorized according to the distribution of the land cover classes. Then texture features of the region are compared with each cluster center and the region is assigned to the best matched

cluster. The cluster validity measure is computed and a re-clustering procedure shall be called if the measure is larger than a predefined threshold.

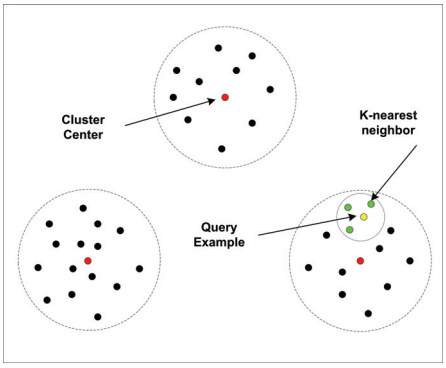


Fig. 14 (a). The query example exists in the database.

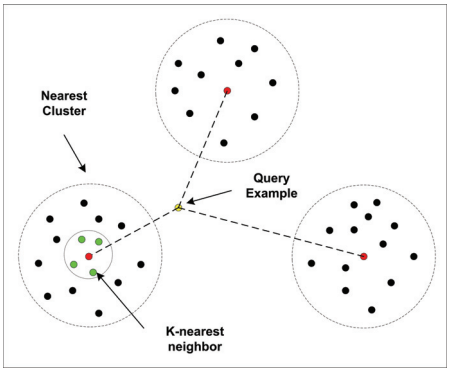


Fig. 14 (b). The query example is new to the database.

6.2. *Performance Evaluation*

Currently, we have processed 4 full-scene TM images, which are decomposed into 64 images containing 4096 regions of 128×128 pixels. On average, searching for similar patterns takes about 150 milliseconds on a PC-Pentium 750MHz running Windows 2000. This query response time is in the same order of the fastest time reported in similar CBIR systems [45].

The classical performance measures, precision and recall, and their variations such as quality and efficiency, the Harmonic Mean, and the E-Measure [46], have been used widely to evaluate the retrieval performance of content-based image retrieval systems. However, these measures assume the set of relevant image for a query is the same, neglecting the fact that different users might have a different understanding of the relevant images. Meanwhile, the complete relevant image set within a very large database is not always available. To deal with these problems, we introduce coverage and novelty, the user-oriented measures [47] originally proposed for document retrieval.

Suppose R is the set of relevant images, A is the discovered image set, U is the subset of R that is known to the user, the intersection of the sets A and U is R_k that represents discovered images known to the user to be relevant, and R_u contains discovered relevant images previously unknown to the user. The relationship between these sets is shown in Fig. 15. Let $|R|$, $|A|$, $|U|$, $|R_k|$, and $|R_u|$ represent the number of images in these sets respectively.

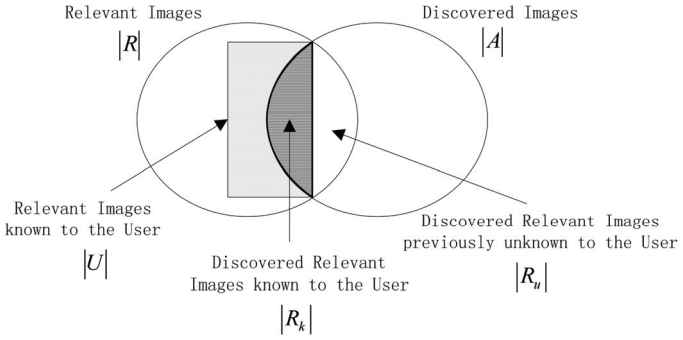


Fig. 15. Coverage and Novelty (adapted from [42]).

The coverage is defined as the ratio of the images known to be relevant that have been discovered,

$$coverage = \frac{|R_k|}{|U|}. \tag{15}$$

The novelty is defined as the percent of the discovered relevant images unknown to the user,

$$novelty = \frac{|R_u|}{|R_u| + |R_k|}. \quad (16)$$

A high coverage value shows that system can retrieve most of the relevant images the user expects to see, while a high novelty value indicates that the system can discover many new relevant images previously unknown to the user. These user-oriented measures are more appropriate for evaluating quality and efficiency of an information mining system, which aims to reveal the unknown information to the user.

The sets of experimental relevant images for each land cover category are identified with the help of the experts in remote sensing. For example, Fig. 16 shows the patterns (agricultural lands around the Missouri River) that are relevant to the query pattern of Fig. 12 (a). Fig. 17 shows the coverage and novelty ratios under different categories. Note that the performance depends much on the specific categories used in this study. Although on average, both the coverage and the novelty ratios are less than 50%, we believe that an appropriate category division can help to improve the overall performance of the system.

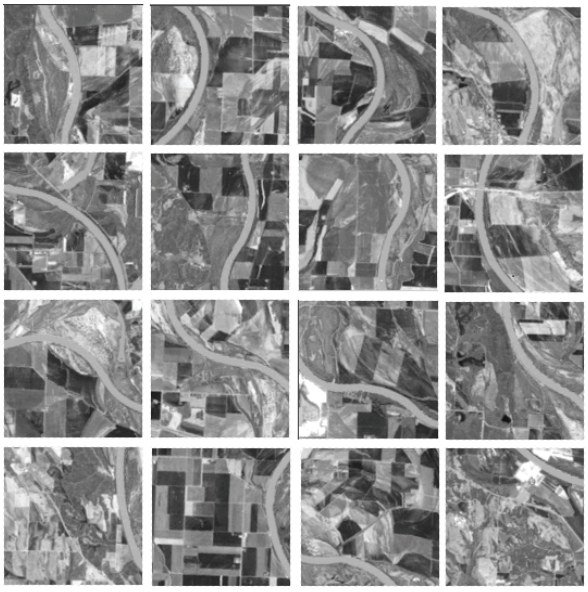


Fig. 16. Relevant patterns (agricultural lands around the Missouri River)

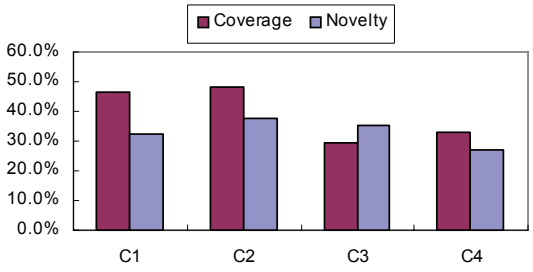


Fig. 17. Coverage and novelty under different categories.

7. Conclusions and Future Work

In this chapter, we have presented a remote sensing image information mining framework, which explores state-of-the-art data mining and databases technologies to retrieve integrated spectral and spatial information from remote sensing imagery. We extracted texture features using statistics of Gabor wavelet coefficients to characterize spatial information, and identified land

cover and land use information corresponding to spectral reflection using SVM-based classification. Feature vectors were clustered and indexed in object-oriented databases with associated raster data stored in image databases. The effectiveness of the system was measured by the coverage and novelty.

Work in progress includes using more multi- and hyperspectral images to build the database systems and identify practical land cover categories for texture analysis. A scalable data warehouse containing a huge amount of images may be a better database architecture for fundamentally distributed data management and mining system such as NASA Earth Observing System (EOS). Meanwhile, performance analysis for clustering on and retrieving from large volumes of images is critical for the system to succeed in practical applications.

In addition, current implementation provides only tile-based search. A segmentation process can be used to segment an image into non-overlapping regions on which we can further apply the texture feature extraction. We will also exploit data warehouses technique and adopt a uniform object-oriented data model for hierarchy multi-resolution storage and retrieval.

The need to include auxiliary data in the process of interpreting remotely sensed data has long been acknowledged by the remote sensing community. However, dealing with different types of data and incorporating them into an information mining process remain a very difficult problem. The system extensibility will consider geographical information system (GIS) connectivity that can help efficiently access auxiliary data in vector format such as soil map, hydrologic map, etc, which will support in-depth data fusion. Consequently, the information derived from image information mining system can be used to correct, update, and maintain cartographic databases within GIS.

Acknowledgments

The authors thank B. S. Manjunath (University of California- Santa Barbara) for providing the source code for texture feature extraction. The authors also thank D. C. Rundquist (University of Nebraska-Lincoln) for providing the image data sets.

References

[1] U. M. Fayyad, G. P. Shapiro, P. Smyth, and R. Uthurusamy (Eds.), *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, 1996.

[2] M.C. Burl, C. Fowlkes, and J. Roden, "Mining for image content," in *Systemics, Cybernetics, and Informatics / Information Systems: Analysis and Synthesis*, Orlando, FL, July 1999.

[3] L.-K. Soh and C. Tsatsoulis, "Data mining in remotely sensed images: a general model and an application," in *Proceedings of IEEE IGARSS 1998*, vol. 2, Seattle, Washington, USA, Jul 1998, pp. 798-800.

[4] J. Zhang, H. Wynne, M. L. Lee, "Image mining: issues, frameworks, and techniques," in *Proceedings of 2nd International Workshop on Multimedia Data Mining*, San Francisco, USA, Aug 2001, pp. 13 – 20.

[5] G.B. Marchisio and J. Cornelison, "Content-based search and clustering of remote sensing imagery," in *Proceedings of IEEE IGARSS 1999*, vol. 1, Hamburg, Germany, Jun 1999, pp. 290 ─ 292.

[6] A. Vellaikal, C.-C. Kuo, and S. Dao, "Content-based retrieval of remote sensed images using vector quantization," in *Proc. of SPIE Visual Info. Processing IV*, vol. 2488, Orlando, USA, Apr 1995, pp. 178 – 189.

[7] R. F. Cromp and W. J. Campbell, "Data mining of multidimensional remotely sensed images," in *Proc. 2nd International Conference of Information and Knowledge Management,* Arlington, VA, Nov, 1993, pp. 471-480.

[8] I E. Alber, Z. Xiong, N. Yeager, M. Farber and W. M. Pottenger, "Fast retrieval of multi- and hyperspectral images using relevance feedback," in *Proceedings of the IGARSS 2001*, vol.3, Sydney, Australia, July 2001, pp. 1149 - 1151.

[9] C.-I Chang and H. Ren, "An experiment-based quantitative and comparative analysis of hyperspectral target detection and image classification algorithms," *IEEE Trans on Geoscience and Remote Sensing*, vol. 38, no. 2, pp. 1044-1063, March 2000.

[10] H. Ren and C.-I Chang, "Automatic spectral target recognition in hyperspectral imagery," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1232-1249, October 2003.

[11] S. M. Schweizer and J. M. F. Moura, "Efficient detection in hyperspectral imagery," *IEEE Trans. on Image Processing*, vol. 10, no. 4, pp. 584 – 597, April 2001.

[12] R. Ramachandran, H. T. Conover, S. J. Graves, and K. Keiser, "Challenges and solutions to mining earth science data," in *Proceedings of SPIE AeroSense Conference on Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, vol. 4057, Orlando, FL, USA, Apr 2000, pp. 259 – 264.

[13] M.C. Burl, C. Fowlkes, J. Roden, A. Stechert, and S. Mukhtar, "Diamond Eye: a distributed architecture for image data mining," in *Proceedings of SPIE AeroSense Conference on Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, vol. 3695, Orlando, FL, USA, Apr 1999, pp. 197 – 206.

[14] M. Datcu, K. Seidel, "Image information mining: exploration of image content in large archives," in *IEEE Aerospace Conference Proceedings*, vol. 3, Big Sky, Montana, Mar 2000, pp. 253 – 264.

[15] K. Koperski, G. Marchisio, S. Aksoy, and C. Tusk, "VisiMine: interactive mining in image databases," in *Proceedings of IEEE IGARSS 2002*, vol. 3, Toronto, Canada, Jun 2002, pp. 1810 ─ 1812.

[16] C.Y. Ji, "Land-use classification of remotely sensed data using Kohonen self-organizing feature map neural networks," *Photogramm. Eng. Remote Sens.*, vol. 66, No. 12, December 2000, pp. 1451-1460.

[17] C. Cortes and V.N. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 1 ─ 25, 1995.

[18] C. J.C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121-167, 1998.

[19] S. Fukuda and H. Hirosawa, "Support vector machine classification of land cover: application to polarimetric SAR data," in *Proceedings of the IGARSS 2001*, vol. 1, Sydney, Australia, July 2001, 187 – 189.

[20] J. A. Gualtieri, S. R. Chettri, R. F. Cromp, and L. F.Johnson, "Support vector machine classifiers as applied to AVIRIS data", in R. Green, Ed*., Summaries of the Eighth JPL Airborne Earth Science Workshop: JPL Publication 99-17*, NASA/JPL, Feb 1999, pp. 217–227.

[21] M. R. Azimi-Sadjadi and S. A. Zekavat, "Cloud classification using support vector machines", in *Proceedings of the IEEE IGARSS 2000*, vol.2, Honolulu, Hawaii, USA, July 2000, pp. 669 – 671.

[22] A. Srivastava and J. Stroeve, "Onboard detection of snow, ice, clouds and other geophysical processes using kernel methods," *Machine Learning Technologies for Autonomous Space Applications Workshop*, 12th Int. Conf. on Machine Learning (ICML-2003), Washington, DC, August 21– 24, 2003.

[23] S. Shekhar, P.R. Schrater, R.R. Vatsavai, W. Wu, and S. Chawla, "Spatial contextual classification and prediction models for mining geospatial data," *IEEE Tran. on Multimedia*, vol. 4, no. 2, pp. 174 -188, June 2002.

[24] T. R. Reed and J.M.H. Buf, "A review of recent texture segmentation and feature extraction techniques", *Computer Vision, Image Processing and Graphics*, vol. 57, no. 3, pp. 359 – 372, 1993.

[25] O. Pichler, A. Teuner, and B.J. Hosticka, "A comparison of texture feature extraction using adaptive Gabor filter, pyramidal and tree structured wavelet transforms", *Pattern Recognition*, vol. 29, no. 5, pp. 733-742, 1996.

[26] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837-842, August 1996.

[27] P. Baumann, A. Dehmel, P. Furtado, R. Ritsch, and N. Widmann, "Spatio-temporal retrieval with RasDaMan," in *Proceeding of the 25th VLDB Conference*, Edinburgh, Scotland, 1999.

[28] J. R. Jensen, *Introductory Digital* Image *Processing: A Remote Sensing Perspective*, Prentice-Hall, NJ, 1996.

[29] Y. Sohn and N.S. Rebello, "Supervised and unsupervised spectral angle classifiers," *Photogramm. Eng. Remote Sens.*, vol. 68, no. 12, pp. 1271 – 1280, December 2002.

[30] N. Cristianini, and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, The Cambridge University Press, Cambridge, UK, 2000.

[31] C. Chang and C. Lin, "LIBSVM: a library for support vector machines," 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[32] B. Scholkopf *et al*., "Comparing support vector machines with Gaussian kernels to Radial Basis Function classifiers," *Technical Report A. I.Memo No. 1599*, Massachusetts Institute of Technology, December, 1996.

[33] T. Joachims, "Making large-scale SVM learning practical", in *Advances in Kernel Methods – Support Vector Learning*, MIT Press, 1999, pp. 169 – 184.

[34] V. Vapnik and O. Chapelle, "Bounds on error expectation for support vector machines," *Neural Computation*, vol. 12 no. 9, pp. 2013 – 2036, 2000.

[35] O. Chapelle and V. Vapnik, "Model selection for support vector machines," in *Advances in Neural Information Processing Systems, vol. 12.* Cambridge, MA: MIT Press, 2000, pp. 230-237.

[36] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 131– 159, 2002.

[37] R. Congalton, "A review of assessing the accuracy of classifications of remotely sensed data," *Remote Sensing of Environment*, vol. 37 pp. 35 – 46, 1991.

[38] J. Daugman, "Complete discrete 2-d Gabor transforms by neural network for image analysis and compression," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 7, pp. 1169 – 1179, 1988.

[39] M. Schaale, I. Keller, and J. Fischer, "Land cover texture information extraction from remote sensing image data," in *Proceeding of the ASPRS-RTI 2000 Annual Conference*, Washington, DC, USA, May 2000.

[40] T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, "An efficient *k*-means clustering algorithm: analysis and implementation," *IEEE Trans. PAMI*, vol. 24, pp. 881– 892, 2002.

[41] D. Pelleg and A. Moore, "X-means: extending k-means with efficient estimation of the number of clusters," in *Proc. 17th Inter. Conf. on Machine Learning*, San Francisco, CA, 2000, pp. 727 – 734.

[42] S Ray and R H Turi, "Determination of number of clusters in *k*-means clustering and application in color image segmentation," in *Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques*, Calcutta, India, December 1999, pp. 137 – 143.

[43] P. S. Bradley and U. M. Fayyad "Refining initial points for *k*-means clustering," in J. Shavlik, Ed., *Proceedings of the 15th International Conference on Machine Learning*, San Francisco, CA, 1998, pp. 91 – 99.

[44] J. M. Pena, J. A. Lozano, and P. Larranaga, "An empirical comparison of four initialization methods for the k-means algorithm," *Pattern Recognition Letters*, vol. 20, pp. 1027 – 1040, Oct 1999.

[45] E. Albuz, E. Kocalar, and A. A. Khokhar, "Scalable color image indexing and retrieval using vector wavelets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 5, pp. 851 – 861, Sep/Oct 2001.

[46] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison Wesley, NY, 1999.

[47] R. Korfhage, *Information Storage and Retrieval*, John Wiley & Sons, Inc., 1997.