Interactive models for semantic labeling of satellite images

Krzysztof Koperski, Giovanni Marchisio, Carsten Tusk, and Selim Aksoy
Insightful Corporation
1700 Westlake Ave. N, Suite 500
Seattle, WA, 98109-3044
{krisk, giovanni, saksoy, ctusk}@insightful.com

ABSTRACT

We describe a system for interactive training of models for semantic labeling of land cover. The models are build based on three levels of features: 1) pixel level, 2) region level, and 3) scene level features. We developed a Bayesian algorithm and a decision tree algorithm for interactive training. The Bayesian algorithm enables training based on pixel features. The scene level summaries of pixel features are used for fast retrieval of scenes with high/low content of features and scenes with low confidence of classification. The decision tree algorithm is based on region level features that are extracted based on 1) spectral and textural characteristics of the image, 2) shape descriptors of regions that are created through segmentation process, and 3) auxiliary information such as elevation data. The initial model can be created based on a database of ground truth and than be refined based on the feedback supplied by a data analyst who interactively trains the model using the system output and/or additional scenes. The combination of supervised and unsupervised methods provides a more complete exploration of model space. A user may detect the inadequacy of the model space and add additional features to the model. The graphical tools for the exploration of decision trees allow insight into the interaction of features used in the construction of models. The preliminary experiments show that accurate models can be build in a short time for a variety of land covers. The scalable classification techniques allow for fast searches for a specific label over a large area.

Keywords : Remote Sensing, Image Databases, Bayesian Classification, Decision Trees.

1. INTRODUCTION

Satellite technology supplies data at an enormous rate. Most database research on the analysis of remotely sensed images has concentrated on data retrieval, and on simple queries that involve spatial joins and spatial selections. For example, the Sequoia 2000 project [15] aimed at the retrieval of raster data, while the Sloan Digital Sky Survey [16] posed the need for creation of a multi-terabyte astronomy archive. Large scale system studies for the analysis of remotely sensed images have been specialized for the detection of particular features, such as volcanoes [2], or have proposed distributed and parallel data storage and query processing systems for handling geo-scientific data retrieval queries [14].

The VisiMine project provides the infrastructure and methodology required for the analysis of satellite images. In order to facilitate the analysis of large amounts of image data, we extract features of the images. Most systems for analyzing remotely sensed images allow simple queries based on the date of image capture, and location. Such systems also allow only simple analyses of single images. When we deal with large collections of remotely sensed images, however, the current systems do not scale well. Therefore, new algorithms and new indexing methods are needed to enable the analysis of data provided by satellite systems.

In order to facilitate the analysis of large amounts of image data, we extract features of the images. Large images are partitioned into a number of smaller, more manageable image tiles. Partitioning allows fetching of just the relevant tiles when retrieval of only part of the image is requested, and provides faster segmentation of image tiles. Individual image tiles are processed to extract the feature vectors. The VisiMine architecture distinguishes between pixel, region and tile levels of feature vectors.

Pixel level features describe spectral and textural information about each individual pixel. Polygon level features describe connected groups of pixels. Following the segmentation process, each polygon is described by its boundary and by a number of attributes that present information about the content of the region in terms of shape, size, etc. The spectral and texture properties are based on pixel features of points within the polygon. Tile level features present spectrum and texture information about whole image tiles. All image features, together with the original images, are

stored in a database system and indexed for fast retrieval. The auxiliary raster data such as Digital Elevation Models (DEM) can also be stored in the database and can be used for feature extraction and data analysis. The Oracle database system provides the means for fast information retrieval and network accessibility.

The data mining power of VisiMine includes similarity searches on tile and polygon levels, clustering of tiles, classification and regression analysis, label training using Bayesian and tree models, and connection to S-PLUS with over 3000 statistical functions. The data mining queries are specified in an SQL-like language. A user may specify the features that are used in the mining task and constraints used to select data for the mining process. The graphical query constructor enables fast query creation by non-technical users.

The user has high level of flexibility in choosing the features and images used for data analysis. The graphical user interface enables presentation of the models on high and general levels as well as drilling down into the details. The label training module enables interactive definition of models for land cover labels.

The tile level summaries of pixel features are used for fast retrieval of tiles with high/low content of features and scenes with low confidence of classification. The initial model can be refined based on the feedback supplied by a data analyst who interactively trains the model using the system output and/or additional scenes. The experts may also refine models created by other users. The VisiMine system enables construction of sophisticated statistical models using the S-PLUS system, which can access data directly from the database, or using the GUI.

In this paper we give an overview of the VisiMine System and present the examples of an interactive training of land cover classification models using Bayesian and decision tree classifiers. For these examples, we used the LANDSAT image of Western Washington State. This image contains about 500MB of raw pixel information in 6 bands (3 visible range and 3 near-infrared bands). The image was corrected for atmospheric and terrain distortions, and was georeferenced. The entire image then was divided into image tiles of size 512 x 512.

2. ARCHITECTURE

We decided to use a database system for storage of the images and their features in order to overcome some limits on to the maximum size of files, and to benefit from indexing, query optimization, and partitioning features of the database. The image tiles and pixel level features are stored as BLOBs, with each band or feature stored in a separate column. The region and tile level features are stored in regular database tables that can be accessed easily for further processing, using VisiMine functions or other software.

Spatial information about region levels also can be stored in ESRI's Spatial Data Engine (SDE), together with the relevant GIS information. SDE provides open data access across local and wide area networks, and the Internet, using the TCP/IP protocol. This information can be combined with region level features such as texture, spectral properties, or DEM features. Other ESRI products, such as, ArcInfo, ArcView, and MapObjects can access the data stored in the SDE, together with additional map data. The SDE format allows a fusion of GIS, optical, and DEM information for a variety of visualization methods and data analysis functions.

A mining process or a similarity search is initiated by submitting a query written in a data mining language similar to SQL. The query language allows the user to specify the type of knowledge to be discovered, the set of data relevant to the mining process, and the conditions that have to be satisfied by the data. Based on this query, an SQL statement is constructed to retrieve the relevant data. The data mining module processes the data and passes the information about the resulting tiles and regions to the GUI, which in turn directly retrieves the images from the database. The capabilities of the data mining engine are enriched through the Java connection to the S-PLUS statistical data analysis engine.

The graphical user interface enables browsing and manipulation of the satellite images and associated features, creation of data mining queries, and visualization of the results of the data analysis. The functionality of VisiMine includes a number of image processing tools, such as histogram equalization, spectral balancing, false colors, polarity switching, masking, overlays, multiband spectral mixing, etc.

The user can create new image types, and can extract image features using the VisiMine Data Manager, which is a separate program. The data manager utility also enables loading of image and DEM data from a file system to the database, browsing of meta-data, and editing of the feature descriptions. Intuitive dialogs guide the user through the specification of parameters necessary for feature extraction, the creation of data types, and the loading of data.

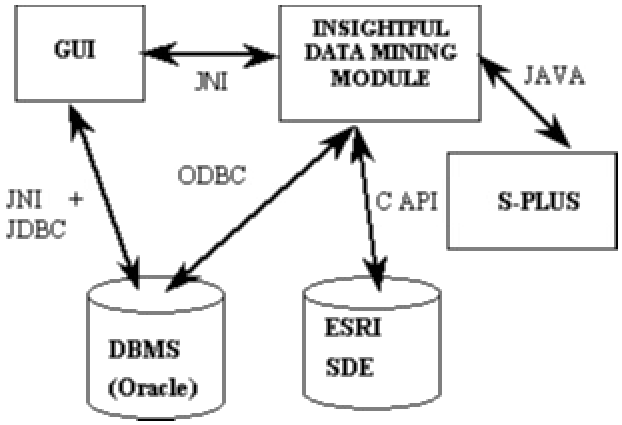

Fig. 1. The Architecture of the VisiMine System.

3. FEATURE EXTRACTION

The VisiMine architecture supports three levels of features: pixel, region, and tile level features. The feature extraction process starts with the analysis of spectral and textural properties at the pixel level. The numerical pixel data can be clustered in order to find a small number of classes. At the same time, tile level features may be extracted, thereby creating histograms of the pixel classes for each tile.

The extraction of region level features starts with a segmentation algorithm. The geometrical properties of regions, such as image moments, are extracted. Based of the pixel features, the system computes statistical properties of regions such as histograms, max, min, mean, and standard deviation features for each region. Additional features are extracted using raster information, such as digital elevation maps. These features can also be created at all three levels. The VisiMine system supports extraction of the following features:

- Texture features:
  - Gabor wavelets [6],
  - Haralick's co-occurrence [5], and
  - Laws texture [10].
- Clustering (spectral, textural, and others) using:
  - CLARA [8] (medoid algorithm),
  - RHSEG [17] (hierarchical algorithm),
  - k-means [8], and
  - model based [8].
- Spectral Mixture Analysis features.
- Segmentation and shape descriptors of the regions.
- Spatial relationships between regions.
- Histograms, max, min, mean, and standard deviation of pixel features for each region and tile.

4. DATA MINING FUNCTIONALITY

VisiMine uses an SQL-like query language that enables specification of the data mining task, features to be used in the mining process, and any additional constraints [4]. The queries are constructed using any text editor, or using Query Constructor, i.e., a GUI tool which guides a user through selection of data and conditions for the query presenting choices for the data that can be used in the query. In the example below we present a query for region similarity

searches based on the content of six endmembers within a region. Only regions with an area larger than 1000 pixels will be retrieved and used in the search.

```
FIND SIMILAR TO SELECTED REGION
IN RELEVANCE TO A.IMAGETILEID,A.REGIONID,
A.CONCRETE,A.CONIFER,A.SOIL,A.GRASS,A.DECIDUOUS,A.WATER
FROM GEOBROWSE.LANDSAT_SMA_R A,GEOBROWSE.LANDSAT_SEGM_R B
WHERE A.IMAGETILEID=B.IMAGETILEID AND
A.REGIONID=B.REGIONID AND
B.AREA > 1000
```

The system is capable of performing similarity searches based on any combination of features. A user can look for the most similar image tiles, or for the most similar regions based on a pattern tile or a pattern region. VisiMine allows weighting of the features. The values of the features can be adjusted to have the range [0, 1], they can be multiplied by a specific value, or they can remain unchanged.

Fig. 2 presents the result of a search for regions similar to McCord Air Force Base. Among the top 4 most similar regions are Olympia Municipal Airport, Whidbey Island Naval Air Station, Arlington Airport, and Abbotsford Airport. We compared the results of the tile similarity search with the region similarity search for the case when the tile containing the pattern region is treated as the pattern tile. In this case, the returned tiles contain only a small fraction of the most similar regions that were returned by region based similarity function. The features of the smaller regions tend to be overwhelmed by the overall features of the tile.

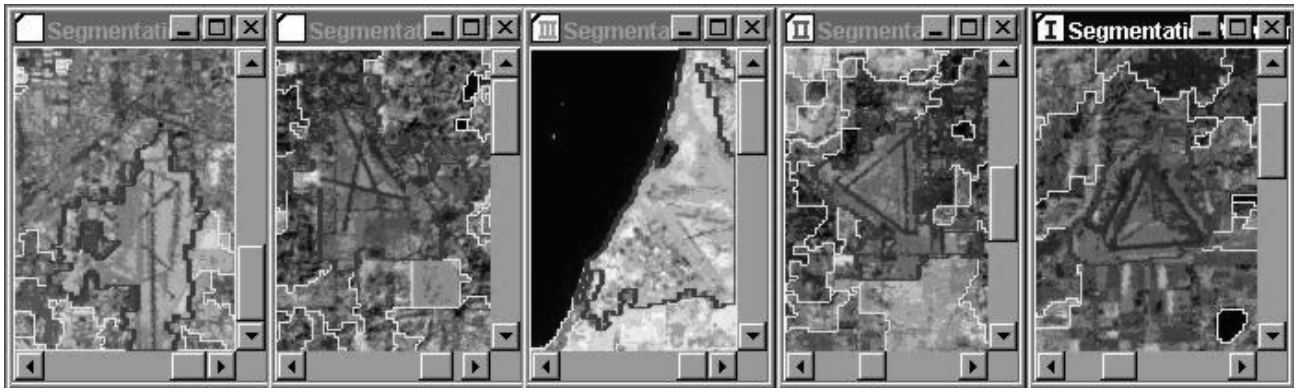

Fig. 2. Region Similarity Search for Airports.

In addition to the similarity search, the VisiMine system provides functionality for other types of analyses of remotely sensed data. This functionality includes the data clustering, building regression and classification models, prediction of land cover types, summarizing data, etc. The system is capable of clustering images into groups of image tiles that have similar features using three different clustering methods. Regions can also be clustered using a model based clustering in order to construct region prototypes for probabilistic retrieval with visual grammar. The medoids of clusters are displayed and a user can examine images that belong to each cluster. Regression trees provide an alternative to clustering by grouping images or regions that have similar values of one of the variables. Regression trees also may be applied on a region level to predict values of numerical attributes based on values of other attributes.

Classification is a data mining technique in which data in a database is analyzed to identify rules that describe the partition of the database into a given set of classes. Each object in the database (in relational databases, a tuple is treated as an object) is assumed to belong to a predefined class, as determined by one of the attributes called the *class label attribute*. The most common classification method uses *decision trees*. Such a method employs a top-down, divide-and-conquer strategy that partitions the set of given objects into smaller subsets where the *leaf* nodes are associated primarily with a single class. Decision tree methods can deal with both numerical and symbolic data. A user can interactively explore parts of the tree, and can prune the tree.

To reduce the gap between low-level features and high-level user semantics, and to support complex query scenarios that consist of many regions with different feature characteristics, VisiMine uses a probabilistic visual grammar that includes

automatic identification of region prototypes and modeling of their spatial relationships [1]. Regions are clustered using a model based clustering into a number of prototype classes, such as cities, rivers, lakes, residential areas, forests, etc. Users can compose queries for complex scenarios by giving a set of example regions that satisfy spatial relationships, such as, bordering, invaded by, disjoint, etc. The system retrieves the image tiles that contain regions with relationships similar to the query pattern.

5. TRAINING USING INTERACTIVE MODELS

In many cases it is very difficult to describe analytically the features of the objects that a user is interested in. Therefore, improvement of the quality of the description may play an important role in image analysis. A method for interactive training of land cover labels using naïve Bayesian classifiers is described in [13]. In that approach, a user can interactively train a Bayesian model to define a number of land cover classes, which can be based on textural, spectral, or DEM properties of images and terrain. The training is done based on categorical pixel level features extracted from numerical pixel features using a clustering algorithm. First, beginning with a single image tile, a user provides positive examples by selecting regions with pixels that belong to the same class, and the system then builds a model for use in identification of additional regions belonging to that class. He/she also has to provide negative examples, by selecting regions with pixels that belong to other classes. Based on this information, a model that estimates the posteriori probability of a pixel's class membership is built. The probability of pixels of the selected tile is shown on the screen. When a user judges the model to be good enough for pixels in the currently selected tile, he/she poses a query that finds either images with a high probability of belonging to the defined class, or images with a low probability (high separability from the class). Using these images, the user may choose to continue training based on other image tiles until the model is judged to be sufficiently good.

While the training is based on pixel level features, the retrieval is based on tile level features. Due to the nature of the naïve Bayesian classifier, which assumes conditional independence of the attributes, it is possible to find the average probabilities of the pixel class assignments in a tile based on the aggregated information about all pixels in the image tile. Despite the fact that the assumption of conditional independence is not always true, naïve Bayesian classifiers perform well in practice. The parameters for each feature can be estimated separately and this simplifies learning.

Formally we want to train a classifier for content label *C* so that we can estimate the probability that a pixel belongs to class *C* given the class labels for that pixel's attributes (e.g. spectral, SMA, textural). The attribute class labels are assigned to each pixel in an offline clustering (unsupervised training) stage separately done for pixel-level spectral, SMA and textural features. The description of the algorithm is presented below.

Algorithm Bayes_Label_Train

Input:

Content label: $C \in \{T, F\}$.

Feature class labels: $A_1, \ldots, A_N$ where N is the number of different pixel features computed for each pixel and A_i is the i'th features (e.g. spectral, SMA, textural) with value $a_i \in \{0, \ldots, K_i\text{-}1\}$ with K_i being the number of possible classes for that attribute (e.g. there are 15 classes ($K_i = 15$) for Laws' texture feature (A_i)).

Procedure

1. Load a tile and its features

2. While training on a single image tile {

If the user labels a pixel with attribute values $a_1, \ldots, a_N$ as a positive example
Increment *count*$\{C = T\}$.
Increment *count*$\{A_i = a_i, C = T\}$ for all $i \in \{1, \ldots, N\}$.
If the user labels a pixel with attribute values $a_1, \ldots, a_N$ as a negative example,
Increment *count*$\{C = F\}$.
Increment *count*$\{A_i = a_i, C = F\}$ for all $i \in \{1, \ldots, N\}$.

Present the probability map for an image shows the probability of each pixel belonging to the content class *C* given its attribute values $a_1,\ldots,a_N$, i.e.

$$P(C = T \mid A_1 = a_1, \ldots, A_N = a_N) = \frac{P(A_1 = a_1, \ldots, A_N = a_N \mid C = T)\, P(C = T)}{P(A_1 = a_1, \ldots, A_N = a_N)}.\}$$

3. Compute for each image tile probabilities P_{tile}:

- Average probability that a pixel within the tile belongs to class C.
- Percentage of pixels with probability that pixel belongs to class C larger than coverage_threshold.
- Percentage of pixels with probability that pixel belongs to class C larger than min_separability_threshold and smaller than max_separability_threshold.

4. Present to the user four sets of tiles:

- N tiles with the largest average probability that a pixel belongs to class C,
- N tiles with the smallest average probability that a pixel belongs to class C,
- N tiles with the highest coverage, i.e., the tiles with the largest percentage of pixels with probability that pixel belongs to class C larger than coverage_threshold.
- N tiles with the lowest separability, i.e. the tiles with the largest percentage of pixels with probability that pixel belongs to class C larger than min_separability_threshold and smaller than max_separability_threshold.

5. If the user is satisfied save the model else load another tile and continue from step 2.

In the Naïve Bayesian classifier, the attributes A_1, …, A_N are assumed to be conditionally independent given the content label C, so

$$P(A_1 = a_1, \ldots, A_N = a_N \mid C = T) = \prod_{i=1}^{N} P(A_i = a_i \mid C = T), \text{ and}$$

$$\begin{aligned} P(A_1 = a_1, \ldots, A_N = a_N) &= P(A_1 = a_1, \ldots, A_N = a_N \mid C = T)\, P(C = T) + \\ &\quad P(A_1 = a_1, \ldots, A_N = a_N \mid C = F)\, P(C = F) \\ &= \prod_{i=1}^{N} P(A_i = a_i \mid C = T)\, P(C = T) + \prod_{i=1}^{N} P(A_i = a_i \mid C = F)\, P(C = F). \end{aligned}$$

Therefore,

$$P(C = T \mid A_1 = a_1, \ldots, A_N = a_N) = \frac{P(C = T) \prod_{i=1}^{N} P(A_i = a_i \mid C = T)}{P(C = T) \prod_{i=1}^{N} P(A_i = a_i \mid C = T) + P(C = F) \prod_{i=1}^{N} P(A_i = a_i \mid C = F)}.$$

Based on the assumption of independence of features we can assume that percentage of the pixels within a tile that have classes $A_1 = a_1$, …, $A_N = a_N$ is equal to the product of the percentages of pixels that belong to each class within the image. Such features are stored in the tile level features. Having this and the probabilities $P(C = T \mid A_1 = a_1, \ldots, A_N = a_N)$ we can compute the P_{tile} probabilities.

The marginal content label probabilities are estimated using relative frequencies (maximum likelihood estimates) as

$$P(C = T) = \frac{count\{C = T\}}{count\{C = T\} + count\{C = F\}}, \text{ and}$$

$$P(C = F) = \frac{count\{C = F\}}{count\{C = T\} + count\{C = F\}}.$$

Note that P(C = T) and P(C = F) should be estimated using the other methods below if *count* {C = T} or *count* {C = F} is zero, i.e. we do not have any training example for that class.

We use five different methods to estimate the conditional attribute class label probabilities as follows. Some of these methods are also called laws of succession because they give the conditional probability that an attribute will have a particular value given its values in previous cases, i.e. training examples, for that class.

Relative frequencies:

$$P(A_i = a_i \mid C = T) = \frac{count\{A_i = a_i, C = T\}}{count\{C = T\}} \quad \text{and}$$

$$P(A_i = a_i \mid C = F) = \frac{count\{A_i = a_i, C = F\}}{count\{C = F\}}.$$

An attribute value is ignored if there are no examples for it. If there are only positive examples for one attribute value and only negative examples for another one, we set the probability to be 0.5.

The relative frequency estimate is the maximum likelihood estimate. However, if we have only a few training examples, the relative frequency estimate can give extreme values and becomes unreliable. A typical solution is the Bayes estimate

$$P(A_i = a_i \mid C = T) = \frac{count\{A_i = a_i, C = T\} + mp}{count\{C = T\} + m}$$, where p is the prior estimate for $P(A_i = a_i \mid C = T)$ and m is the weight given to the prior (i.e. the number of training examples required for the significance of their estimate to be the same as the significance of the prior) [9, 11]. It can also be explained as a linear interpolation between the maximum likelihood estimate (relative frequency) and the prior p [12]. Laplace's law of succession is a special case which is the Bayesian estimate starting from the uniform prior of $1/K_i$ on the attribute values and using $m = K_i$. Jeffreys-Perks' law of succession is the special case with $m = 0.5\ K_i$. These two estimates have been criticized for assigning either too much or too little probability to specific events [12] but were also decided to be safe choices when the distribution of the source is unknown and the number of possible classes (K_i) is fixed and known [11].

Ristad's law of succession can be explained as follows. We consider each attribute value to be a symbol in an alphabet. Then, a training session is a string that is a subset of symbols in the alphabet. In the first interpretation, all nonempty subsets of the alphabet are equally likely (uniform subsets prior). In the second interpretation, all nonzero subset cardinalities are equally likely (uniform cardinality prior). According to the uniform subsets prior, both very large and very small subsets are relatively improbable. On the other hand, the uniform cardinality prior assigns more probability to small and large subsets of the alphabet and less to subsets of moderate cardinality. Both laws reduce to Laplace's law of succession when all the symbols are attested, i.e. $q_{iT} = K_i$ in our case [12].

After initial training a user may choose another set of features for training. The sequence of positive and negative examples is stored in the original model and a new model can be build based on the same set of examples. This features enable a user to try combinations of different features and choose the most appropriate for a given task.

Fig. 3 presents the tile used for training of cloud label, the first set of tiles with the highest coverage after the initial training and the final set of tiles with the highest coverage after the training on the tile that contains snow. The first coverage set of tiles contained a tile 1504, which contains a snow-covered mountain. Because the model was not trained to distinguish between white cloud and white snow this tile is treated as containing clouds. The tile with snow was loaded and model was adjusted using snow as negative example.

Multiple labels can be trained and they can be combined using a probabilistic framework. The pixels are classified using all models and a label with the highest probability is assigned to the pixel. Optionally different pixel features may be tried. During the construction of a model the locations of positive and negative examples are stored. Using this information the Bayesian model can be build using any combination of pixel features. A user can choose the best combination of features for the final model.

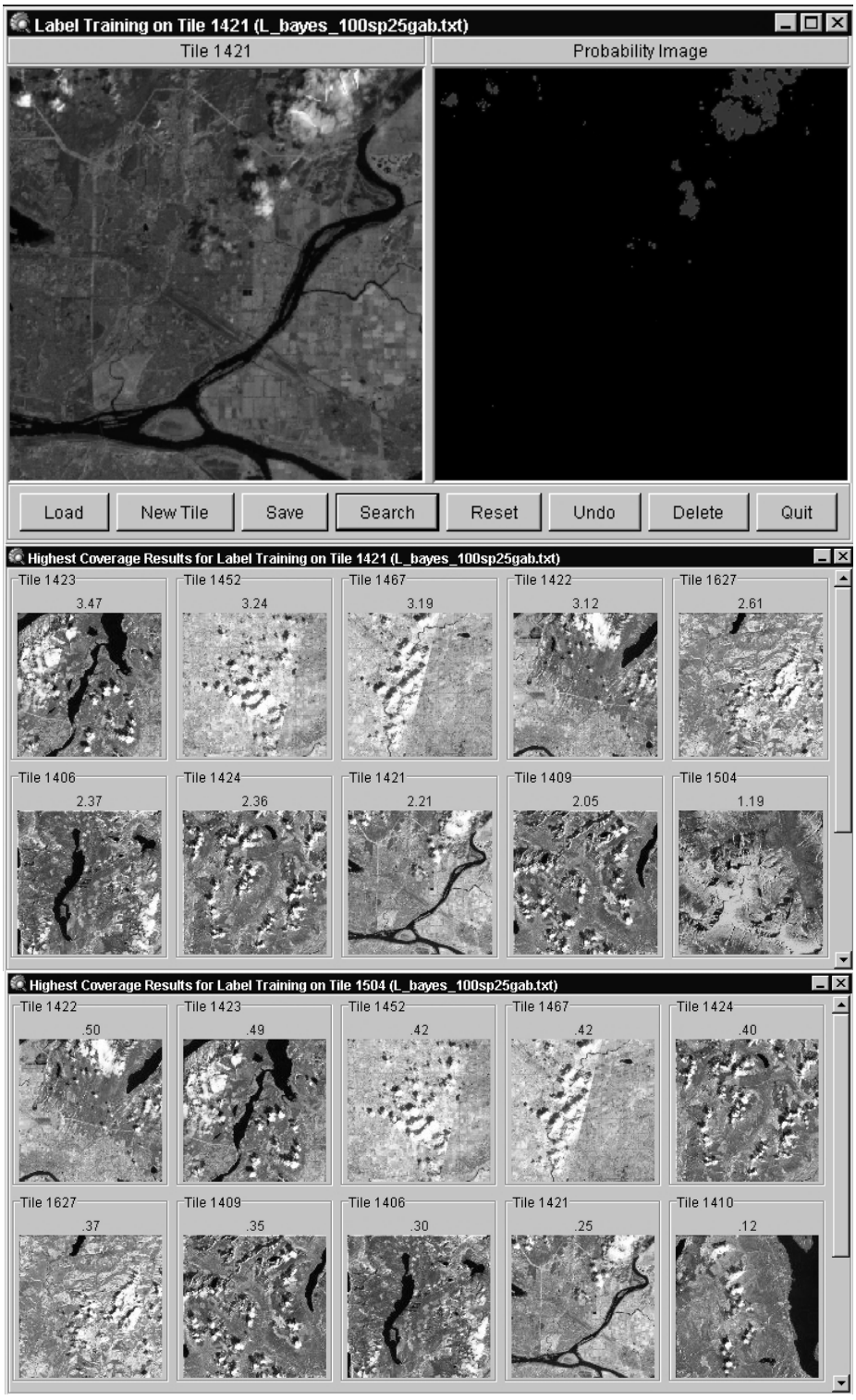

Fig. 3 Training using Bayesian models

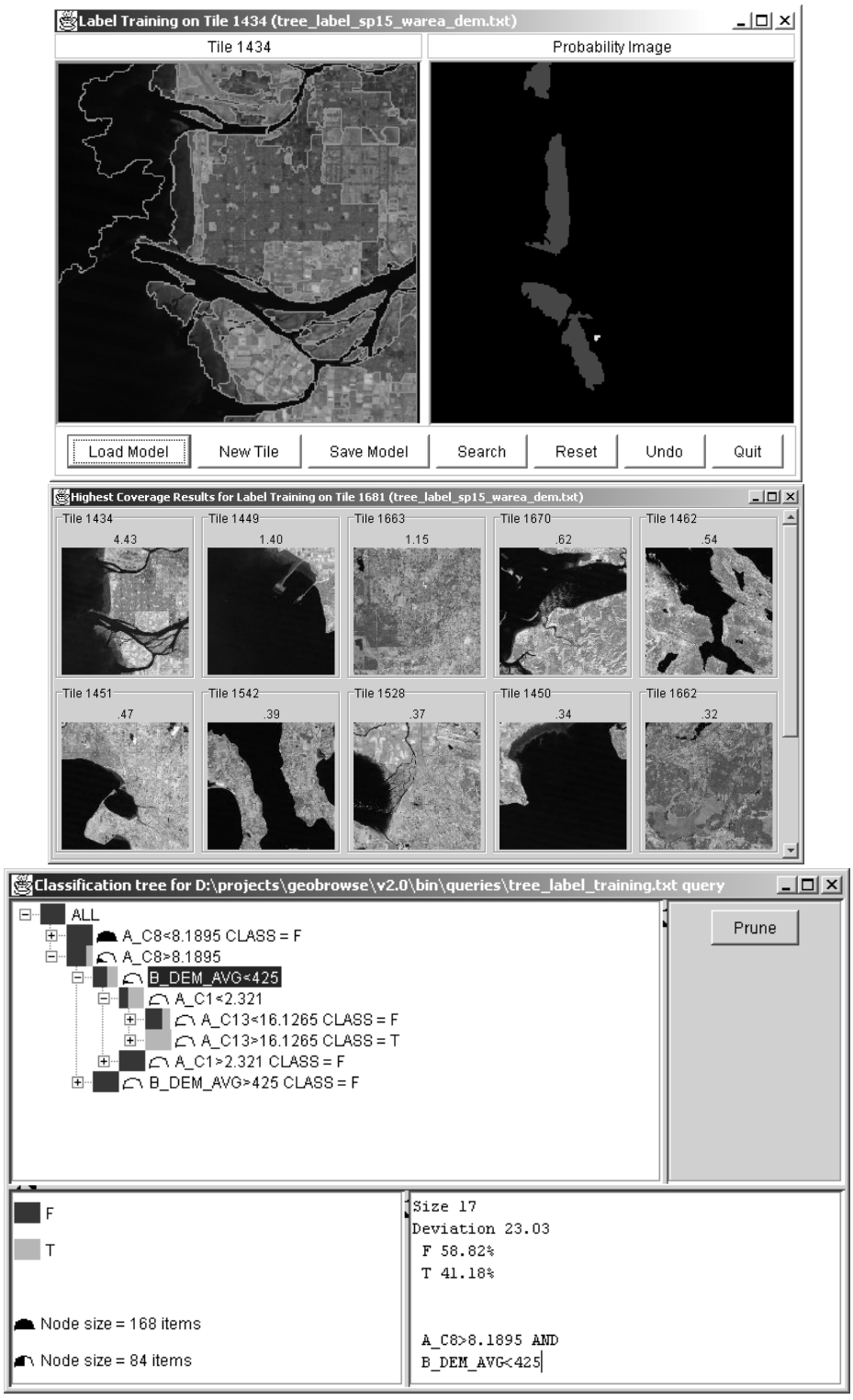

Fig. 4. Label training using region level spectral and elevation features.

Another method for label training is based on decision tree models [11]. Because the classification process on the pixel level would be extremely expensive to compute, the decision tree models are based on the region level features. In

addition to spectral properties of the regions we can perform classification based on shape properties and areas of regions, as well as on auxiliary GIS information. For example, the spectral reflectance of concrete is very similar to spectral reflectance of different types of rocks. Additional information, such as Digital Elevation Models can be used to distinguish between these two types of land cover. In a way similar to Bayesian label training, a user provides the system with positive examples by pointing to regions that belong to the trained class, and negative examples by pointing to other regions. For the first tile used for training a user has to provide examples for all regions within the tile. Default value is false (region does not belong to land cover class). For all other tiles used for training only regions pointed by the user are added to the model. Fig. 4 shows the results of decision tree label training that utilizes region spectral class histograms and DEM information. The examples of tidal flats are presented to the system and the regions with the highest coverage of the regions with high probability to be a tidal flat according to the model are shown. The decision tree is presented the highlighted node presents the split based on the average elevation of the region.

6. S-PLUS CONNECTIVITY

Insightful's S-PLUS is an interactive computing environment for graphics, data analysis, statistics, and mathematical computing. It contains a superset of the S object-oriented language and system originally developed at AT&T Bell Laboratories, and it provides an environment for high-interaction graphical analysis of multivariate data, modern statistical methods (e.g., robust and non-parametric methods), data clustering and classification, and mathematical computing. In total, S-PLUS contains over 3000 functions for scientific data analysis. VisiMine data can be accessed from within S-PLUS by using Java connectivity for images and ODBC connectivity for image and region data. In addition, VisiMine has the S-PLUS command tool, which provides for easy transfer of images to S-PLUS, and for data processing using the S-PLUS language. The S-PLUS images can be returned to VisiMine and displayed there.

The VisiMine can also display S-PLUS graphics, which are created using a command line interface and shown within S-PLUS plot window. Fig. 5 shows computations of NDVI index for a residential area of Burnaby British Columbia [7].

The combination of S-PLUS and VisiMine features creates a unique environment for interactive exploration and analysis of remotely sensed data. The rich statistical functionality of S-PLUS, together with the VisiMine user interface and the scalability of its data mining engine, allows for easy and powerful customization of the data analysis process.

7. CONCLUSIONS

We have designed, and implemented, the VisiMine system for data mining of remotely sensed images. Three levels of feature are extracted from image tiles and used in the data mining process. In addition to simple queries based on simple properties, such as geographic location or acquisition date, a user can submit queries based on feature vectors describing the images. Interactive training of the classification models that describe new types of objects can be easily performed. Scalability to the large databases is addressed through indexing of the feature vectors and by using scalable data mining algorithms in the query processing. Our region level indexing strategy enhances the data analysis and similarity search processes by allowing for the more refined classification of information derived from images. VisiMine can convert large quantities of multispectral satellite imagery into GIS format. The results of image segmentation are stored in the raster format in any database and in vector format in ESRI's Spatial Data Engine (SDE). Other ESRI products, such as, ArcInfo, ArcView, and MapObjects can access the data stored in SDE together with additional map data. The GIS, optical and DEM information can be combined together in order to be used in a variety of visualization methods and data analysis functions. The functionality of S-PLUS software can be interactively accessed from VisiMine and data can be transferred to and from S-PLUS system for further processing and visualization.

8. ACKNOWLEDGEMENTS

Funding for the prototype comes from NASA contract NAS5-01123 and NIH SBIR Phase II grant 2 R44 LM06520-02.

9. REFERENCES

1. S. Aksoy, G Marchisio, K. Koperski, and C. Tusk. Probabilistic Retrieval with a Visual Grammar. *Proc. IEEE International Geoscience and Remote Sensing Symposium*, Toronto, Canada, 2002.
2. U. M. Fayyad, and P. Smyth. Image Database Exploration: Progress and Challenges. *Proc. 1993 Knowledge Discovery in Databases Workshop*, Washington, DC, p. 14 - 27, 1993.

3. A. R. Gillespie, M. O. Smith, J. B. Adams, and S. C. Willis, Spectral Mixture Analysis of Multispectral Thermal Infrared Images, *Proceedings of the 2nd Thermal IR Multispectral Scanner Workshop*, JPL Publication 90-55:57 - 74, 1990.
4. J.Han, Y. Fu, W. Wang, K. Koperski, and O. R. Zaïane. DMQL: A Data Mining Query Language for Relational Databases. *Proc. of the Workshop on Research Issues on Data Mining and Knowledge Discovery*, Montreal, QB, pp. 27 - 34, 1996.
5. R. M. Haralick, K. Shanmugam, and T. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610-621, November 1973.
6. G. M. Hayley, and B. M. Manjunath, Rotation Invariant Texture Classification using Modified Gabor Filters, *Proc. of IEEE ICIP95*, pp. 262 -265, 1994.
7. J. R. Jensen. *Introductory Digital Image Processing: a Remote Sensing Perspective*. Prentice-Hall, 1996.
8. L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, 1990.
9. R. F. Krichevskiy. Laplace's Law of Succession and Universal Encoding. *IEEE Transactions on Information Theory*, vol. 44, no. 1, January 1998.
10. K. I. Laws. Rapid Texture Classification. *SPIE Image Processing for Missile Guidance*, vol. 238, pp. 376-380, 1980
11. T. M. Mitchell. *Machine Learning*, McGraw-Hill, 1997.
12. E. S. Ristad, A Natural Law of Succession. Technical Report, CS-TR-495-95, Department of Computer Science, Princeton University, July 1995.
13. M. Schröder, H. Rehrauer, K. Seidel, and M. Datcu. Interactive Learning and Probabilistic Retrieval in Remote Sensing Image Archives. *IEEE Trans. on Geoscience and Remote Sensing*, Sep. 2000, Vol. 38, No. 5 pp. 2288 - 2298.
14. E. C. Shek, R. R. Muntz, E. Mesrobian, and K. Ng. Scalable Exploratory Data Mining of Distributed GeoScientific Data. *Proc. of The Second International Conference on Knowledge Discovery & Data Mining*, Aug. 2-4, Portland OR, pp. 32 - 37, 1996.
15. M. Stonebraker, J. Frew, K. Gardels, and J. Meredith. The Sequoia 2000 Storage Benchmark. *Proc. ACM-SIGMOD International Conference on Management of Data*, Washington, D.C., pp. 2 - 11, 1993.
16. A. Szalay, P. Kunszt, A. Thakar, J.Gray, D. Slutz, and R. J. Brunner. Designing and Mining Multi-terabyte Astronomy Archives: The Sloan Digital Sky Survey. *Proc. ACM-SIGMOD International Conference on Management of Data*, Dallas TX, pp. 451 - 462, 2000.
17. J. C. Tilton. A recursive PVM implementation of an image segmentation algorithm with performance comparisons between the HIVE and the Cray T3E. *Proc. of the 7th Symposium on the Frontiers of Massively Parallel Computation*, Annapolis, MD pp. 146-153, Feb. 21-25, 1999.

-

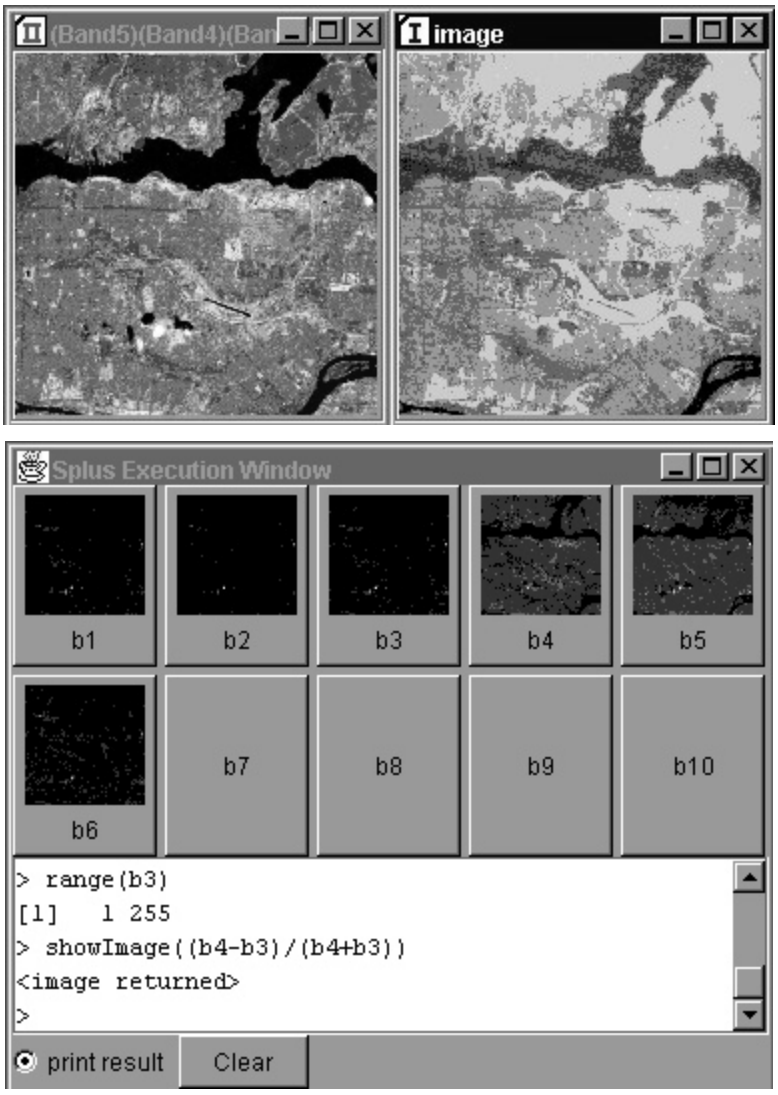

Fig. 5. S-PLUS Interface and Graphics.