Available online at www.sciencedirect.com
ScienceDirect
Remote Sensing of Environment 107 (2007) 149 – 158
Remote Sensing of Environment
www.elsevier.com/locate/rse

An operational MISR pixel classifier using support vector machines

Dominic Mazzoni *, Michael J. Garay, Roger Davies, David Nelson

Jet Propulsion Laboratory, California Institute of Technology, United States

Received 2 March 2006; received in revised form 22 June 2006; accepted 23 June 2006

Abstract

The Multi-angle Imaging SpectroRadiometer (MISR) data products now include a scene classification for each 1.1-km pixel that was developed using Support Vector Machines (SVMs), a cutting-edge machine learning technique for supervised classification. Using a combination of spectral, angular, and texture features, each pixel is classified as *land*, *water*, *cloud*, *aerosol*, or *snow/ice*, with the aerosol class further divided into *smoke*, *dust*, and *other aerosols*. The classifier was trained by MISR scientists who labeled hundreds of scenes using a custom interactive tool that showed them the results of the training in real time, making the process significantly faster. Preliminary validation shows that the accuracy of the classifier is approximately 81% globally at the 1.1-km pixel level. Applications of this classifier include global studies of cloud and aerosol distribution, as well as data mining applications such as searching for smoke plumes. This is one of the largest and most ambitious operational uses of machine learning techniques for a remote-sensing instrument, and the success of this system will hopefully lead to further use of this approach.
© 2006 Elsevier Inc. All rights reserved.

Keywords: MISR; Multi-angle; Support vector machines; Supervised classification; Neural networks

1. Introduction

The Multi-angle Imaging SpectroRadiometer (MISR) (Diner et al., 1998) provides an unique view of the Earth by capturing images from nine cameras at fixed angles ranging from nadir to 70.5° in both the forward and aft directions relative to the direction of motion of the satellite platform. MISR exploits the multiple angular views of each scene to retrieve a number of physical parameters, including the height of clouds using stereoscopic pattern matching (Moroney et al., 2002) and aerosol microphysical properties (Martonchik et al., 2002). While MISR data products include multiple cloud masks based on several different algorithms, previously there was no data product that attempted to make distinctions between clouds and aerosols, or between clouds and highly reflective snow or ice. Several years ago we began a project to investigate the use of machine learning techniques to add new classifiers and to improve the accuracy of existing MISR cloud classifiers (as described in the Algorithm Theoretical Basis Documents on "MISR Level 1 Cloud Detection" and "MISR Level 2 Cloud Detection and Classification" (Diner et al., 1999)). This has resulted in the operational system we describe here.

2. Background

2.1. Supervised classification

In the paradigm of supervised classification, a user wishes to classify objects into k discrete categories. In the training phase, the user presents to a learning algorithm a collection of n objects $x_1,\ldots, x_n$ along with corresponding labels $y_i \in \{1,\ldots,k\}$, indicating the category or class each object belongs to. In this case, each x_i is a vector of numerical features derived from one MISR 1.1-km image pixel to be classified, and the labels y_i correspond to the different scene classifications, such as "cloud" or "smoke." The learning algorithm then develops a model of the data, which it can then use to classify new examples into the correct category. (In contrast, unsupervised classification is where an algorithm looks for patterns in the data to separate it into categories without being given training labels by a user.) The accuracy of a particular model can be assessed by applying it to new examples and comparing the classifications to the "correct" classifications given by a human expert. Note that the same learning algorithm can often produce quite different

* Corresponding author. Google, Inc., 604 Arizona Ave., Santa Monica, CA, 90401.
E-mail address: dominic@minorninth.com (D. Mazzoni).

0034-4257/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.rse.2006.06.021

models given the same training data, depending on various user parameters. No straightforward learning algorithm is known that always produces a good model on any training data without any other user parameters. In fact, so-called "perfect" learning can be proven to be computationally infeasible. Nevertheless, recent advances have resulted in practical techniques for machine learning that work quite well under many circumstances.

Machine learning systems are either online or offline. Online learners continually update their model based on new observations or new labels. Offline learners go through a training process to produce a single fixed model, which is then used to classify all future examples. While we were collecting training labels from scientists, we used the online learning paradigm: the model was constantly being updated as each new label was added. Once a comprehensive set of training labels was collected, we switched to an offline learning mode, using a slower, more robust technique to train a single best model. Our operational system simply applies this model to all new examples, without any adaptation. This keeps the operational system simple and predictable. Of course at any time we have the option of swapping in a better model that was trained offline.

Methods for supervised classification include maximum likelihood, k-nearest-neighbor, decision trees, artificial neural networks, and Support Vector Machines (SVMs). For this project, we focused on SVMs, one of the newest and most promising learning techniques. One of the shortcomings of many learning techniques is the tendency to overfit the training examples, resulting in a model that correctly classifies the training data but fails to generalize well to new examples. SVMs are specifically constructed to minimize a statistical bound on the generalization error, resulting in models that extrapolate to new examples quite well. SVM training is also a deterministic quadratic optimization procedure, making SVM training faster than neural network training, which is usually done using a randomized procedure. One downside of SVMs is that the models they produce can be large, leading to slow evaluation of new examples. To circumvent this problem, we explored various optimization techniques to make SVMs fast enough for operational use.

2.2. Related work

Dozens of papers exist on automatic satellite pixel classification using learning algorithms. Some pioneering work was done by Welch et al. (1992), comparing the use of discriminant analysis and two types of neural networks to classify pixels in Advanced Very High Resolution Radiometer (AVHRR) images as one of a number of classes. Bankert (1994b,a) and Bankert and Aha (1995) compared neural networks to decision trees and a 1-nearest-neighbor classifier in AVHRR data. Tian et al. (2000) used probabilistic neural networks to classify 10 cloud types in Geostationary Operational Environmental Satellite (GOES) images using both spatial and temporal features. Saitwal et al. (2003) extended Tian et al.'s work to nighttime classification. Baum et al. (1997) used fuzzy logic to detect multilayer systems in AVHRR scenes based on examples getting classified into more than one of eight trained cloud classes. Azimi-Sadjadi and Zekavat (2000) used a hierarchical arrangement of support vector machines to classify six different cloud types in infrared GOES-8 imagery, while Lee et al. (2004) investigated using a multi-class support vector machine to distinguish between ice and water clouds in MODIS images. Li et al. (2003) used the maximum likelihood technique to improve on the basic cloud classification provided in the MODIS standard product. This is only a tiny fraction of the work that has been done in the area, and is not intended to represent a comprehensive survey.

While many research projects have investigated the use of machine learning techniques to classify pixels from remote sensing instruments, very few have resulted in an application in an operational environment. One successful example is Bankert's real-time classifier for GOES images available on the web (Bankert, 2005), which uses a nearest-neighbor classification algorithm and a large collection of training data. More details about his approach are given in Tag et al. (2000), which describes the approach as applied to AVHRR imagery. An SVM-based classifier is currently running onboard NASA's EO-1 spacecraft, as part of the Autonomous Sciencecraft Experiment (Mazzoni et al., 2005a,b). When images of certain targets are acquired by the Hyperion instrument, the classifier applies a trained SVM to 12 of its 242 spectral bands to classify each pixel as land, water, cloud, ice, or snow. Simple heuristics are then used to determine whether or not the image should be saved for downlink or discarded to make room for more interesting images.

2.3. MISR data products

The Multi-angle Imaging SpectroRadiometer (MISR) instrument (Diner et al., 1998) flies on NASA's Terra (AM-1) spacecraft, in sun-synchronous polar orbit with an equator crossing time of 10:30 am. MISR images an approximately 400 km wide swath of the daylit portion of the earth in four spectral bands (blue, green, red, and near-infrared) from nine different viewing directions (±70.5°, ±60.0°, ±45.6°, ±26.1°, and 0°). Each camera has a maximum resolution of 275 m, but in normal operation only the nadir camera and the red channel of the other 8 cameras capture data at full resolution; the other channels are subsampled to 1.1 km to reduce the data transmission volume.

MISR's level 1B (L1B) products, which contain the calibrated radiances from all nine cameras, are coregistered based on their ground location, onto a predetermined grid described by the World Geodetic System 1984 (WGS94) ellipsoid using the Space-Oblique Mercator (SOM) projection (Diner et al., 1998). Georectification of the MISR image pixels has been shown to be accurate to approximately 50 m (Jovanovic et al., 2002). Although the nine MISR cameras view the earth at slightly different times, with approximately 1 min between each camera's view, the accurate coregistration of the images onto the same grid makes it easy to compare each camera's view of the same pixel.

MISR's level 2 (L2) products contain data sets derived from the L1B data, such as the cloud masks, the estimated height of each pixel, estimated wind speed and direction, aerosol optical depth, and aerosol microphysical properties. L2 products range from 1.1-km to 17.6-km resolution. We developed our SVM cloud classifier to work as a new L2 product, taking L1B data as input and outputting a classification decision for each 1.1-km pixel.

2.4. Support vector machines

What follows is an extremely brief introduction to the theory of Support Vector Machines (SVMs). Modern SVMs were introduced by Cortes and Vapnik (1995); interested readers are referred to tutorials such as Burges (1998) or books such as Schölkopf and Smola (2002) for a thorough treatment of the subject. As in most machine learning problems, we represent each training example by a *feature vector* $\vec{x}_i$, where all of the relevant data used to classify each example has been captured in a vector of numbers. Consider, first, the binary case, where there are only two classes of items to separate. Given a set of training examples $\vec{x}_1, \ldots, \vec{x}_N$ and corresponding training labels $y_1, \ldots, y_N \in \{+1, -1\}$, a linear binary SVM attempts to find the hyperplane that best separates the positive from the negative examples; new points are then classified by determining on which side of the hyperplane they lie. Of all possible hyperplanes that separate the training examples, the one is chosen which maximizes the *margin*, the sum of the distances between the hyperplane and the nearest positive and negative example. Parameterizing the hyperplane as $\vec{w} \cdot \vec{x} + b = 0$, the following inequality expresses the constraint that each example must fall on the proper side of the hyperplane:

$$y_i(\vec{x}_i \cdot \vec{w} + b) - 1 \geq 0 \qquad (1)$$

If all constraints are minimally satisfied, it can be shown that the margin must equal $2/||\vec{w}||$. Thus finding the margin-maximizing satisfying solution corresponds to minimizing $||\vec{w}||$ while satisfying Eq. (1) for each $i \in \{1,\ldots, N\}$. Using Lagrange multipliers, the problem is transformed into its Wolfe dual, making it possible to express the optimization purely in terms of the Lagrange multipliers α_i, where finding the solution is equivalent to maximizing

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \vec{x}_i \vec{x}_j \, . \qquad (2)$$

subject only to the constraints that each $\alpha_i > 0$. This can be solved using conventional quadratic programming (or specialized algorithms for SVMs such as SMO (Platt, 1998, 1999)), and $\vec{w}$ and b can be computed directly once the solution has been found.

To allow some of the examples to fall on the wrong side of the hyperplane when no perfect solution can be found, positive slack variables ξ_i can be introduced such that:

$$y_i(\vec{x}_i \cdot \vec{w}_i + b) - 1 + \xi_i \geq 0. \qquad (3)$$

To maximize the margin while minimizing the use of slack variables, our new objective function is to minimize the sum:

$$||w|| + C \sum_{i=1}^{N} \xi_i \qquad (4)$$

where C is a user defined parameter that controls the balance between margin maximization and error minimization. Surprisingly, when converting to the dual form, the slack variables disappear and the solution is identical to Eq. (2) except that each Lagrange multiplier α_i must now be bounded above by C.

To handle nonlinear situations, note that the original example vectors $\vec{x}_i$ only appear in Eq. (2) inside of an inner product. Replacing $\vec{x}_i \cdot \vec{x}_j$ with $(\vec{x}_i \cdot \vec{x}_j)^2$ turns out to be equivalent to mapping each $\vec{x}_i$ to a higher-dimensional feature space and performing the dot-product there. In general, we replace each $\vec{x}_i \cdot \vec{x}_j$ with $K(\vec{x}_i, \vec{x}_j)$, where $K(\vec{u}, \vec{v})$ is a so-called *kernel function*, which operates only on the inner product of $\vec{u}$ and $\vec{v}$, but corresponds to mapping $\vec{u}$ and $\vec{v}$ to a (possibly infinite-dimensional) space and computing the dot product there. It turns out that the functions permissible as K are those which are *positive definite*. Common kernels include:

Linear	$K(\vec{u}, \vec{v}) = \vec{u} \cdot \vec{v}$
Polynomial	$K(\vec{u}, \vec{v}) = (\vec{u} \cdot \vec{v} + a)^p$
Gaussian	$K(\vec{u}, \vec{v}) = \exp(-\gamma \cdot \|\| \vec{u} - \vec{v} \|\|^2)$
Sigmoid	$K(\vec{u}, \vec{v}) = \tanh(\gamma \cdot \vec{u} \cdot \vec{v} + a)$
Normalized	$K'(\vec{u}, \vec{v}) = K(\vec{u}, \vec{v}) K(\vec{u}, \vec{u})^{-\frac{1}{2}} K(\vec{v}, \vec{v})^{-\frac{1}{2}}$

Substituting a kernel function into the optimization problem is a convenient way to add nonlinearity because the optimization problem has not increased in complexity, but the hyperplane found belongs to the implicit higher-dimensional space. When backprojected into the original space, the separating surface can now be curved. Before, it was possible to compute the equation for the hyperplane directly, but now $\vec{w}$ and b exist in the implicit higher-dimensional space. In order to determine what side of the hyperplane a new example $\vec{q}$ is on, the training examples and Lagrange multipliers must be used:

$$f(\vec{q}) = \sum_{i=1}^{N} \alpha_i y_i K(\vec{q}, \vec{x}_i) + b, \qquad (5)$$

which returns a positive number if the example is on the positive side of the hyperplane and a negative number otherwise. For large numbers of training examples, this can lead to a lengthy computation just to evaluate a single new example. Luckily, α_i will usually go to zero for the majority of the examples; the remaining example vectors with positive α_i are called *support vectors* and are the only ones that need to be used to evaluate a new example. Typically the number of support vectors scales roughly with the complexity or difficulty of the classification problem. Thus, for difficult problems, the number of support vectors can still be large.

To reduce the computational challenge of evaluating new examples when there are hundreds of support vectors, various techniques have been proposed to reduce the number of support vectors needed. Burges (1996) proposed a method to incrementally construct a smaller set of vectors that approximate the same hyperplane as the SVM. The more vectors that are used, the better the approximation, but often the results are indistinguishable from the original hyperplane after only a fraction of vectors have been constructed. This method is powerful but difficult to implement in practice because it requires a nonlinear nonconvex optimization procedure.

Finally, SVMs are extended to classify examples into more than two classes by breaking the problem down into several binary problems. For example, if there are k classes, $k(k-1)/2$ classifiers can be trained, one to distinguish between each pair of classes. To evaluate a new example, all classifiers are run and each one casts a “vote” as to the correct class. A simpler method is to train k “one-vs-all” classifiers, each one of which distinguishes between one class and all $k-1$ others. To evaluate a new example, the one with the largest (most positive) raw SVM output from Eq. (5) is chosen as the class decision. The one-vs-all approach is slower to train but faster to classify, which was an important consideration since our final classifier needed to be able to run quickly. Neither of these two approaches has been shown to be superior to the other in terms of accuracy; they perform similarly, as do several other proposed methods for solving multi-class problems using binary classifiers.

3. Methodology

Here we describe how feature vectors were constructed from the raw MISR pixel data, and then how we went about collecting training data and developing the operational SVM classifier. One of our goals in this project was to maximize the effectiveness of time donated by expert scientists to this project. To realize this goal we used a custom tool for interactive training that expert scientists used when providing training labels. This interactive approach was an important part of the success of the project.

3.1. Constructing feature vectors

An SVM, like any other machine learning algorithm, considers each example it sees as simply a mathematical vector of numbers; the technique is the same whether the examples to be classified are pixels, people, or potatoes. An important step in training a supervised classifier in the real world is to determine what features should be used for each example. Previous pixel classifiers used for remote-sensing instruments have used spectral, texture, and contextual features. Because our classifier applies to MISR data, we could also incorporate angular features, making use of data from multiple cameras. We therefore had four types of features to consider incorporating. To determine whether a certain type of feature was worth including, experiments were done comparing the accuracy of classification of the same examples with and without each type of feature, using a JPL-developed feature extraction library called LibFeature (Mazzoni, 2005). Some of these experiments were performed early on, before any expert training examples were available, in order to create an initial system to demonstrate the concept and to allow further experimentation. However, these experiments were repeated several times over the course of the training process, as we refined our plan over time.

- *Spectral* features were a clear win. Using radiances from all four of MISR’s spectral bands resulted in significantly higher accuracy than using just one or two. We converted each radiance value to a Bidirectional Reflectance Factor so that pixels acquired at different times and of different locations could be compared directly.
- *Angular* features were also clearly useful, but with a caveat. Because the MISR imagery is registered to the earth ellipsoid, objects that are above sea level appear to move from south to north through the sequence MISR cameras depending on their height above the surface, due to the parallax effect. For this reason, including radiance information from MISR’s 70° cameras in the same vector as data from the nadir (0°) camera tended to reduce classification accuracy, because the pixels were often not from the same object. After much experimentation, we settled on extracting features from the center five (+45.6°,+26.1°, 0°, −26.1°, −45.6°) of MISR’s nine cameras.
- *Contextual* features could include the latitude and longitude of the pixel being classified, the view angle geometry, the time of year, the expected type of surface (land, water, etc.) or even things such as local weather conditions. We investigated several such features, but decided against using most of them out of fear of overfitting. For example, including the latitude as a feature would likely result in a classifier that was more likely to classify a pixel as snow/ice near the poles and less likely near the equator. We felt this was undesirable as it could be less likely to detect clouds over the poles or snow on mountains in mid-latitudes. Including the time of year as a feature was problematic because, in order to avoid undesired behavior, examples would need to be found of every class during every time of year, significantly increasing the effort during the training process. As a result, we settled on only two such features: the cosine of the solar zenith angle and the expected surface type (land vs. water). The cosine of the solar zenith angle accounts for the change in the amount of available illumination per unit area depending on the relative location of the sun in the sky and helps distinguish between surfaces that are truly reflective and locations, such as oceanic sunglint, that are bright due to geometric effects. The surface type was already available and easily obtained from MISR ancillary data fields; its resolution was the same as our classifier, 1.1 km square.
- *Texture* features include a broad range of possibilities. The use of texture features usually implies numerically capturing

the patterns that might appear in a local region, defined as anywhere from a few neighboring pixels to a radius of several dozen pixels. Popular texture features include Haralick features, wavelet features, and Gabor filter features. Unfortunately, computing these features can be quite slow, requiring tens of thousands of computations per pixel, and we found this to be unacceptable for either our interactive training application or for our intended operational classifier. Instead, we were inspired by the work of DeCoste and Schölkopf (2002) who showed that SVMs can been used to successfully distinguish between different handwritten digits, where the feature vectors consist of simply the grayscale values of a bitmapped representation of the scanned digit image. Other methods failed to achieve high accuracy without incorporating some knowledge of the structure of the data, but SVMs were found to be quite successful at using the raw pixel values in feature vectors. Based on this, texture information was obtained by including the actual radiance values from a 5 × 5 neighborhood of 1.1-km pixels centered at each target pixel into the feature vector. Including a 5 × 5 neighborhood from at least one camera and spectral band appeared to improve the accuracy significantly over a 3 × 3 neighborhood. A 7 × 7 neighborhood did not improve the accuracy; we suspect that perhaps far more training examples would have been needed to deal with feature vectors that large.

In the end, the feature vector that we chose to use operationally consisted of different neighborhoods of radiance values from each of MISR's cameras and spectral bands, plus the two contextual features. It included 154 features: the solar zenith angle, the surface class, and the 152 features represented by Table 1.

All of our features were continuous-valued except for the surface type, which was binary (land or water). Though including the surface type could lead to discontinuities at coastlines, in practice these were rarely observed, and empirically including this flag in the feature vector improved the classifier's accuracy.

We chose to gather most of our feature vectors from the green channel not for any particular scientific reason, but because empirically when we considered the channels independently, the classification had the highest accuracy with the green channel. A feature vector of size 154 is relatively large for this type of classification. An interesting avenue for future research would be to explore feature selection algorithms. However, we discovered that SVMs work quite well with large feature vectors, making it less important to minimize the size of the feature vectors.

Table 1

	45.6°	26.1°	Nadir	26.1°	45.6°
Near-IR			3 × 3		
Red			3 × 3		
Green	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Blue			3 × 3		

3.2. Interactive training

It is very common for the process of collecting training data to be totally complete before any machine learning algorithms are run. However, we were very concerned that, at best, this is an inefficient use of time, and, at worst, it can lead to poor classification accuracy. As an alternative, a custom tool was developed, called *PixelLearn*, for interactively labeling multi-spectral, multi-angle datasets. PixelLearn includes a built-in fast SVM training algorithm, and it updates the classification on one half of the screen while the user adds more labels on the other half. Fig. 1 shows a screenshot of PixelLearn in action, where a user has labeled land, water, and cloud classes on top of a MISR image on the left, and the SVM has classified the rest of the pixels on the right.

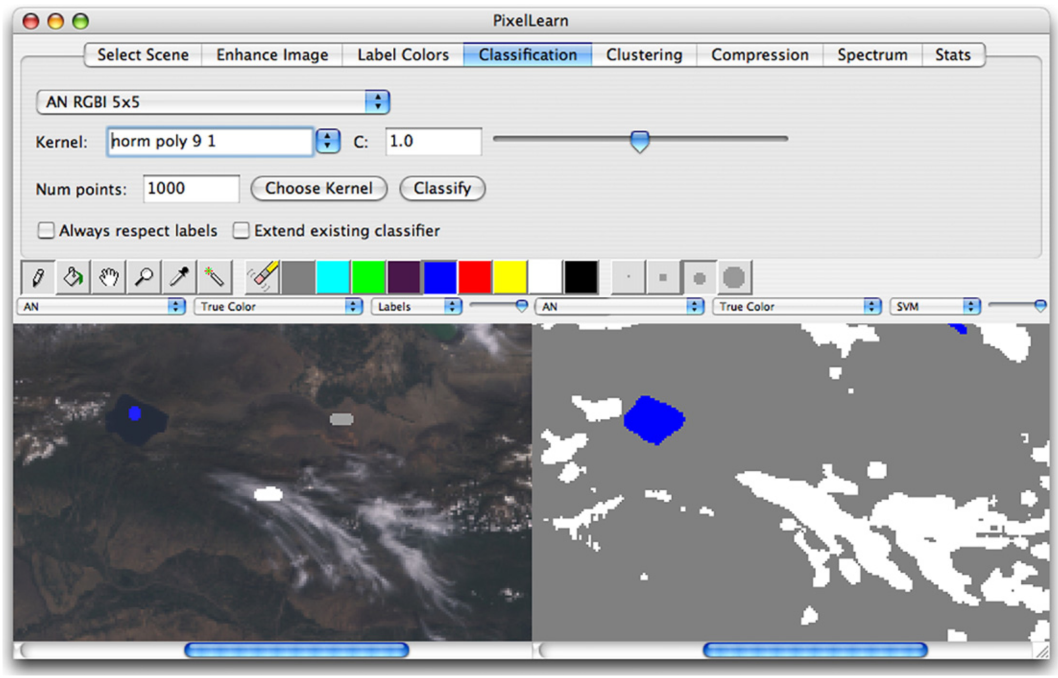
One advantage of using an interactive training process is that the classes did not need to be chosen ahead of time. Initially, our primary goal was attempting to distinguish among different cloud types (cirrus, cumulus, stratus, etc.). This work is described in Garay et al. (2005) and Mazzoni et al. (2005a,b). Reasonable success was achieved, but the overall effectiveness of our approach was limited because cloud formations often consist of more than one cloud type and the differences among the cloud types as seen from the satellite can be rather subjective. The task of separating clouds from aerosols, such as smoke and dust, and snow/ice proved to be a more well-defined problem. While there was much interest in distinguishing between dust and smoke, we did not feel that we had collected enough training examples of each. Therefore, we grouped all aerosols into a single category, then used a separate SVM to distinguish between dust, smoke (including volcanic ash), and other aerosols within this category.

For speed, the SVM training algorithm in PixelLearn typically only trains on a few thousand examples at a time. When the user has labeled more pixels, it automatically subsamples proportionally from each of the labeled classes. Various SVM parameters such as the choice of kernel function and the value of the regularization parameter C can be modified by the user, or chosen automatically to fit the current labeled data using a cross-validation algorithm.

3.3. Offline training

At the end of the interactive phase, we had obtained hundreds of thousands of labeled pixels spanning hundreds of scenes. The training labels cover all four seasons and a full range of latitudes, however we noted that due to the interactive training process, we ended up with relatively few training scenes of easy-to-classify scenes, such as clouds over ocean, and many training scenes involving relatively rare classes that are more difficult to classify, such as smoke plumes, snow on mountains, and aerosols.

Training a single support vector machine on the entire data set was, unfortunately, infeasible because SVM training time is slightly worse than quadratic in the number of training examples. Instead, inspired by approaches to this problem such as Collobert et al. (2002) and Graf et al. (2005), we trained



AN RGBI 5x5

Kernel: horm poly 9 1 C: 1.0

Num points: 1000

Fig. 1. The PixelLearn tool we developed allowed scientists to explore MISR data and interactively label the pixels. In the example above, the user has labeled three regions containing water, clouds, and land in the lower-left pane, and a support vector machine classification of the image appears in the lower-right pane. For advanced users, the controls in the top pane control all aspects of SVM training.

an SVM on a small subset of the training examples, then applied this to all of the remaining training examples and added only those examples that were classified incorrectly to the new training subset. The result was approximately 10,000 training examples, which took on the order of hours to converge. The SVM used a radial basis function kernel with $\gamma = 0.1$; the kernel parameters were chosen using cross-validation on the training set.

Our resulting SVM had 1941 support vectors. With 154 elements in each feature vector, evaluating a single pixel would require about 300,000 multiply-add instructions, about an order of magnitude larger than we could afford given the computational constraints that needed to be met in order to run our algorithm as part of the operational MISR data products system. As a result, it was necessary to find a way to reduce the number of support vectors without significantly affecting the accuracy of the classifier.

We found that Burges' constructive method (Burges, 1996) worked quite well at finding an approximate solution to binary SVMs. We made several improvements by adapting the algorithm to reduce multiclass SVMs, as well as binary, and using a faster, more robust optimization procedure based on Differential Evolution (Storn and Price, 1997) instead of Burges' gradient descent; this new technique is described in a paper by Tang and Mazzoni (2006). As the number of reduced vectors increased, the accuracy of our reduced SVM improved, approaching the accuracy of the model containing all 1941 support vectors. We decided that the SVM containing 98 reduced vectors had approximately the right combination of accuracy and speed. (On our validation data set, described below, the non-reduced SVM had an overall accuracy 0.9% higher than the reduced SVM, but was over 10 times slower.) Our dust vs. smoke SVM was trained in a similar manner. Because it was less important, we settled on only 20 reduced vectors to use for this classifier. Both SVMs used the same feature vectors.

4. Operational implementation

We integrated the SVM classifier into the operational data processing system at the Langley Atmospheric Science Data Center, as part of release 4.0 of the MISR processing software, which started processing new data on December 1, 2005. The code runs as part of the level 2 processing and generates several new fields as part of the top-of-atmosphere/cloud classifiers data file. Examples of the classifications produced by the operational classifier can be seen in Figs. 2 and 3. For information on how to access and use these data products, see the MISR Data Products Specifications (http://eosweb.larc.nasa.gov/PRODOCS/misr/DPS/).

While the code to build feature vectors from MISR pixels and evaluate an SVM was written specifically for this application, the SVM model is stored in a separate ancillary file to enable future enhancements. The ancillary file also contains a table indicating the number of pixels to include from each MISR camera/band in the feature vector, allowing for the possibility of a future enhancement to the classifier using a different feature vector (within the same framework).

For efficiency, data is processed one row at a time. Because of our 5 × 5 neighborhoods of pixels used for texture, the radiance data from at least two previous and two following rows must be kept in memory as well. Feature vectors are generated for one

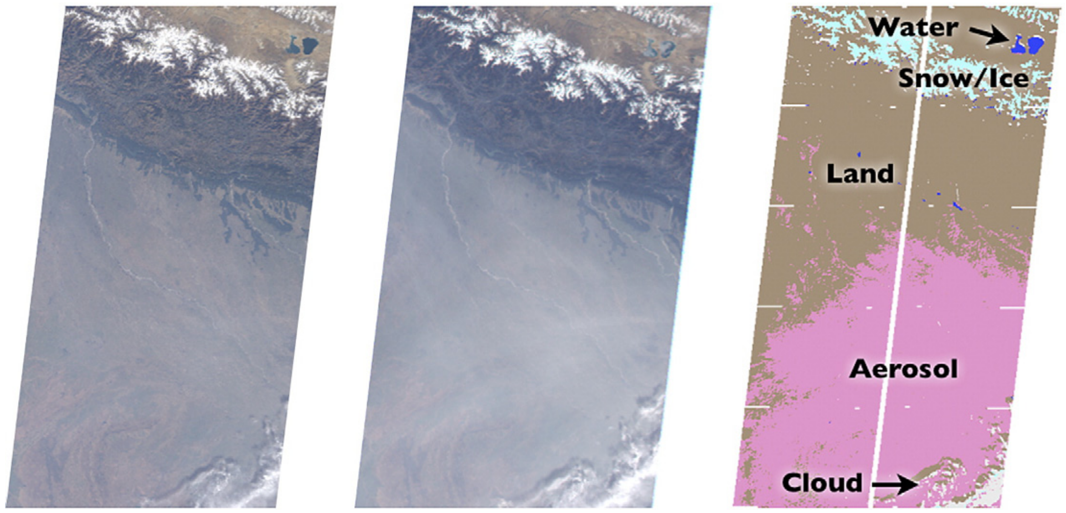


Fig. 2. An example of the operational MISR pixel classifier in action. The scene focuses on north-eastern India, with the Himalayas at the top. The left swath is from MISR's nadir-pointing camera, and the middle swath is from MISR's 45° forward-pointing camera. The image on the right shows a representation of the SVM classifier's output. The SVM classifier has correctly identified the snow on the mountains, a lake, pollution aerosols over the low-lying plains, and clouds in the lower part of the scene. The vertical stripe down the center of the MISR image is due to poor-quality data from one of the pixels in the center of one of MISR's cameras. This data was captured on December 2, 2005, Terra orbit 31686, path 145, MISR blocks 66–70.

row of data and stored in a single contiguous array. If any radiance values from any pixel are missing, the entire pixel is marked as bad and the SVM computation is skipped; SVMs are not robust to missing values in feature vectors. The most time consuming operation is to take a row of m feature vectors of length d, and compute the output of the kernel function $K(\vec{u},\vec{v})$ between each feature vector and each of the n support vectors. Dot products were implemented using a cache-efficient matrix multiply subroutine, taking $O(m \cdot d \cdot n)$ operations.

Recall that our five-class SVM is implemented using five separate binary SVMs, each trained to distinguish one class from the four others. The classification decision for each pixel is chosen by the class with the largest SVM output value. However, this adds little information about the possible uncertainty of this result. Five additional fields were therefore included that indicate the likelihood that the pixel belongs to each of the five classes, where the confidence is determined by the real-valued output of each binary SVM. To keep the file size small, only one of four possible values for each class is reported:

1. With high confidence, pixel is in this class.
2. With low confidence, pixel is in this class.
3. With low confidence, pixel is not in this class.
4. With high confidence, pixel is not in this class.

These values were chosen to be analogous to the four possible values for a pixel in a MISR cloud mask. As a result, a user can not only determine that one pixel is most likely aerosol,

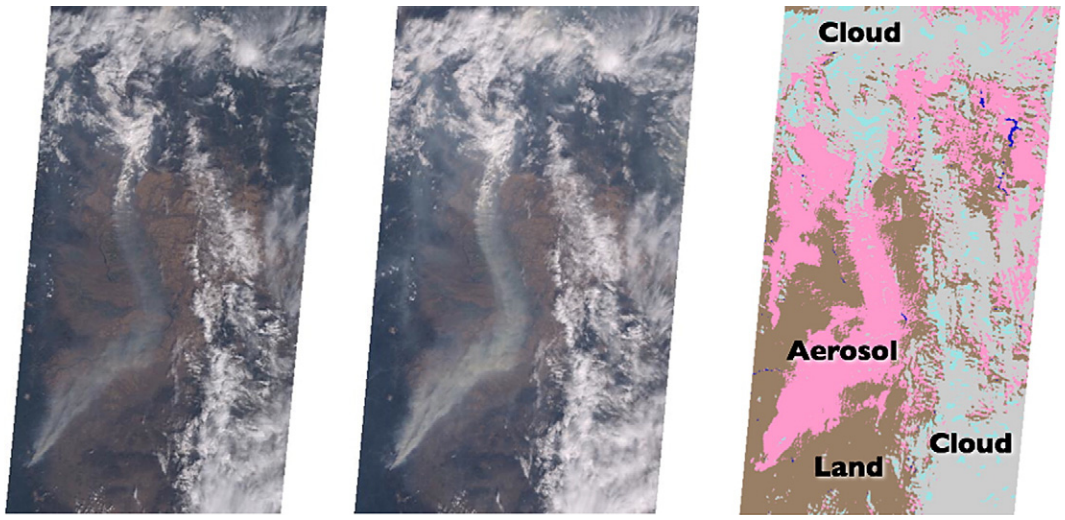


Fig. 3. Another example of the operational MISR pixel classifier. This scene shows the B&B Complex Fires in central Oregon that were observed on September 4, 2003. The images show MISR's nadir-pointing camera, MISR's 45° forward-pointing camera, and the SVM classifier's output. The SVM classifier has correctly separated the smoke from the clouds in this scene, and nicely traces out the shape of the smoke plumes. Terra orbit 19753, path 44, MISR blocks 51–55.

for example, they can also distinguish between the confident case where the aerosol classifier is highly positive, and the other classifiers are highly negative, and the ambiguous case where two or more classifiers are highly positive. We used thresholds of ±1.0 on the SVM output, which has good theoretical justification, i.e. if the SVM output was 1.1 it would be high confidence positive, if 0.9 it would be low confidence positive.

Although the dust vs. smoke SVM was trained entirely separately, it is packed into the same matrix of support vectors for simplicity. Thus, a total of 118 support vectors are evaluated for each pixel. The first 98 of these are used for the single five-class classification (aerosol, cloud, land, water, ice/snow) plus the five confidence fields for these classes, and the last 20 support vectors are used to distinguish between dust and smoke, but only over areas where the aerosol is chosen as the most likely classification.

5. Validation

The final classifier was validated by having two human experts independently label millions of pixels. These labels were then compared with the results of the SVM classifier. In addition, two human experts each labeled some of the scenes independently, making it possible to compare the error rates of the SVM with the degree of disagreement between the human labels. Unfortunately at this time we have not completed validation of the aerosol subclassification (dust vs. smoke vs. other aerosols); we hope to discuss this in a future paper.

Validation was performed using four complete MISR orbits, chosen so that they were spread out reasonably well in both space and time. Table 2 expresses the overall accuracy, expressed as the percentage of pixels for which there was agreement in the classification.

Overall, the SVM's five-class classification decision agreed with the human expert 80.9% of the time, at the 1.1-km pixel level. When each block of 16×16 pixels was replaced by the majority class, the agreement improved to 84.9%, indicating that a small but significant fraction of the errors were isolated blunders and not gross misclassifications. The level of agreement between two humans, at 93.0% and 96.3% for the 1.1-km and 17.6-km resolutions, respectively, quantify the degree of subjectivity present in these experiments. These serve as an upper bounds for the maximum performance one could expect from any classifier. It should be noted that Hutchison et al. (1997) gave an accuracy of 98–99% for a *single*, trained analyst repeatedly classifying the *same* scene using an interactive labeling tool. Discrepancies are obviously increased when two different scientists attempt to classify the same scene. Moreover, inspection of the differences revealed that they were most common along the boundaries of classes — the edges of clouds, for example. These situations are fundamentally ambiguous, but their influence on the result decreases when the 17.6-km resolution is used, as indicated by the improvement in both the SVM and human classification.

Table 2

	1.1-km resolution	17.6-km resolution
SVM	80.9%	84.9%
Human expert	93.0%	96.3%

Table 3

%	Cloud	Water	Land	Ice	Aerosol
Cloud	80.1	7.0	2.3	8.0	1.9
Water	10.0	88.5	0.5	0.3	0.6
Land	11.6	1.5	77.7	3.3	5.8
Ice	28.2	0.2	1.5	70.0	0.1
Aerosol	6.6	8.4	9.6	0.1	75.3

The *confusion matrix* (Kohavi and Provost, 1998) in Table 3 breaks down the performance of the SVM by class. The human-labeled classification is shown along the left-hand column, and the SVM classification is shown along the top row.

The percentage of pixels correctly classified is shown along the diagonal of the confusion matrix. Off-diagonal elements indicate percentages of misclassification. For example, the SVM correctly classified 80.1% of the pixels labeled as clouds. It misclassified 7.0% of cloudy pixels as water, 2.3% of cloudy pixels as land, 8.0% of cloudy pixels as ice, and 1.9% of cloudy pixels as aerosol. Note that each row within the confusion matrix must add to 100% because each pixel must be classified as one category by the SVM. Each column within the matrix will not, in general, add to 100%. As an example, consider the case of a classifier that classified all pixels as cloudy. Then the "Cloud" column would have 100% in each category, summing to 500%, while each row would still equal 100%.

For the MISR SVM classifier, the confusion matrix shows that the classification for the cloud and water classes is the most accurate. However, in this form the confusion matrix does not capture the fact that some classes are far more prevalent than others. The confusion matrix in Table 4 shows the same data, but expressed in thousands of pixels, instead of percentages.

Expressed in this way, the confusion matrix allows some overall performance characteristics to be determined. The overall accuracy of the classifier is the ratio of the number of correct classifications to the total number of pixels. For the MISR SVM, the overall accuracy is nearly 81% across all classes. However, as shown in the table, clouds were approximately 50 times more prevalent than aerosols within our validation data set. In order to correctly classify aerosols, it is necessary to have a relatively high false-positive rate. Looking at the last column, one can compute that if a pixel is classified as an aerosol, there is only a 33% chance that this is the correct classification. However, looking at the last row, if a pixel really is an aerosol, there is a 75% chance it will be classified as such.

Table 4

	Cloud	Water	Land	Ice	Aerosol
Cloud	10939	945	311	1086	252
Water	417	3695	22	13	27
Land	248	33	1666	71	125
Ice	563	4	29	1395	2
Aerosol	18	23	26	0	204

This bias in favor of false positives is a natural consequence of the fact that the aerosol class is relatively rare.

It should also be noted that, looking at the classification output, many of the misclassifications are easily spotted. For example, a pixel in the middle of a cloud in the central Pacific can be classified as snow/ice. Given that the SVM is using only local radiance information to make its classification decision, this is to be expected. An application making use of the SVM classifier could simply incorporate additional information to result in an improved classification. Moreover, making use of the local context, which has not been done for this global classifier, the results could be improved significantly.

6. Conclusions

The project described in this paper is one of the more ambitious and large-scale applications of machine learning technology to an operational remote-sensing application. We demonstrated that using an interactive application is an effective way to make more efficient use of expert scientists' time, and that it allows the exact definitions of the classes to be flexible until it is determined what classifications can be made reliably. We also demonstrated that, using reduced-set techniques, it is possible to create an SVM with limited computational requirements.

A variety of applications exist for this type of classification product and the MISR project is already making use of the SVM classifier described in this paper for a number of them. In one case, the classifier is being used in conjunction with pattern recognition software to automatically detect smoke plumes from forest fires over North America. In addition, the automatic classifier significantly reduces the time and effort it takes to perform global studies of smoke and aerosol distributions, which are important for understanding human impact on global climate change.

The advantage of using a supervised classifier that can be easily trained interactively is that it gives scientists confidence in its performance in a variety of situations, while, at the same time, allowing rapid development and application towards studies involving large amounts of satellite data. The support vector machine approach adds the additional benefit of flexibility in feature selection that greatly reduces the time required to explore the possible important parameters of a problem. This is particularly useful in situations where the distinguishing features are not readily apparent at the outset. A possible shortcoming of the approach is that the trained SVM may not reveal the physical mechanisms within the feature set that allow the classification to be performed accurately. One future avenue of research is developing techniques that allow this important information to be extracted from the trained SVM.

Acknowledgements

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

Azimi-Sadjadi, M. R., & Zekavat, S. A. (2000). *Cloud classification using support vector machines. Proceedings of the 2000 IEEE Geoscience and Remote Sensing Symposium (IGRASS 2000), Vol. 2*. (pp. 669−671) Honolulu, HI.

Bankert, R. (2005). Naval research laboratory Monterey GOES cloud classification (website). http://www.nrlmry.navy.mil/sat-bin/clouds.cgi

Bankert, R. L. (1994). Cloud classification of AVHRR imagery in maritime regions using a probabilistic neural network. *Journal of Applied Meteorology*, *33*, 909−918.

Bankert, R. L. (1994). Cloud pattern identification as part of an automated image analysis. *Preprints, 7th American Meteorological Society Conference on Satellite Meteorology and Oceanography* (pp. 441−443). Boston, MA.

Bankert, R. L., & Aha, D. W. (1995). Automated identification of cloud patterns in satellite imagery. *Preprints, 14th Conference on Weather Analysis and Forecasting, American Meteorological Society* (pp. 313−316). Dallas, TX.

Baum, B. A., Tovinkere, V., Titlow, J., & Welch, R. (1997). Automated cloud classification of global AVHRR data using a fuzzy logic approach. *Journal of Applied Meteorology*, *36*, 1519−1540.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*(2), 121−167.

Burges, C. J. C. (1996). Simplified support vector decision rules. *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 71−77).

Collobert, R., Bengio, S., & Bengio, Y. (2002). A parallel mixture of SVMS for very large scale problems. *Advances in Neural Information Processing Systems*. MIT Press.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273−297.

DeCoste, D., & Schölkopf, V. (2002). Training invariant support vector machines. *Machine Learning*, *46*, 161−190.

Diner, D. J., Beckert, J. C., Reilly, T. H., Bruegge, C. J., Conel, J. E., Kahn, R., et al. (1998). Multiangle imaging spectroradiometer (MISR) instrument description and experiment overview. *IEEE Transactions on Geoscience and Remote Sensing*, *36*, 1072−1087.

Diner, D. J., Abdou, W., Ackerman, T., Borel, C., Bruegge, C., Chrien, N., et al. (1999). *Multiangle imaging spectroradiometer (MISR) algorithm theoretical basis documents (ATBDs).* Available http://eospso.gsfc.nasa.gov/eos_homepage/for_scientists/atbd/viewInstrument.php?instrument=9

Garay, M. J., Mazzoni, D. M., Davies, R., & Diner, D. (2005). The application of support vector machines to the analysis of global datasets from MISR. *Proceedings of the Fourth Conference on Artificial Intelligence Applications to Environmental Science* San Diego, CA.

Graf, H. P., Cosatto, E., Bottou, L., Dourdanovic, I., & Vapnik, V. (2005). In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Parallel support vector machines: The cascade svmAdvances in Neural Information Processing Systems*, *Vol. 17*. (pp. 521−528) Cambridge, MA: MIT Press.

Hutchison, K. D., Etherton, B. J., Topping, P. C., & Huang, H. L. (1997). Cloud top phase determination from the fusion of signatures in daytime AVHRR imagery and HIRS data. *International Journal of Remote Sensing*, *18*, 3245−3262.

Jovanovic, V. M., Bull, M. A., Smyth, M. M., & Zong, J. (2002). MISR in-flight cam-era geometric model calibration and georectification performance. *IEEE Transactions on Geoscience and Remote Sensing*, *40*, 1512−1519.

Kohavi, R., & Provost, F. (1998). Glossary of terms. *Machine Learning*, *30*, 271−274.

Lee, Y., Wahba, G., & Ackerman, S. (2004). Cloud classification of satellite radiance data by multicategory support vector machines. *Journal of Atmospheric and Oceanic Technology*, *21*(2), 159−169.

Li, J., Menzel, W. P., Yang, Z., Frey, R. A., & Ackerman, S. A. (2003). High-spatial-resolution surface and cloud-type classification from MODIS multispectral band measurements. *Journal of Applied Meteorology*, *42*, 204−226.

Martonchik, J. V., Diner, D. J., Crean, K. A., & Bull, M. A. (2002). Regional aerosol retrieval results from MISR. *IEEE Transactions on Geoscience and Remote Sensing*, *40*, 1520−1531.

Mazzoni, D. (2005, April). LibFeature: A software library for quickly generating feature vectors on the fly from structured data. *Proceedings of the Eighth*

Workshop on Mining Scientific and Engineering Datasets, 2005 SIAM International Conference on Data Mining.

Mazzoni, D., Tang, N., Doggett, T., Chien, S., Greeley, R., & Cichy, B. (2005). Learning classifiers for science event detection in remote sensing imagery. *Proceedings of the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2005)*.

Mazzoni, D. M., Horváth, A., Garay, M. J., Tang, B., & Davies, R. (2005). A MISR cloud-type classifier using reduced support vector machines. *Proceedings of the Eighth Workshop on Mining Scientific and Engineering Datasets, 2005 SIAM International Conference on Data Mining*.

Moroney, C., Davies, R., & Muller, J. -P. (2002). Operational retrieval of cloud-top heights using MISR data. *IEEE Transactions on Geoscience and Remote Sensing*, *40*, 1532−1540.

Platt, J. (1998). Sequential minimal optimization: A fast algorithm for training support vector machines. *Technical Report MSR-TR-98-14, Microsoft Research (MSR)*.

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, & A. J. Smola (Eds.), *Advances in Kernel Methods — Support Vector Learning* (pp. 185−208). Cambridge, MA: MIT Press.

Saitwal, K., Azimi-Sadjadi, M. R., & Reinke, D. (2003). A multichannel temporally adaptive system for continuous cloud classification from satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing*, *41*(5), 1098−1104.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels.* Cambridge, MA: The MIT Press.

Storn, R., & Price, K. (1997). Differential evolution — A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*, 341−359.

Tag, P. M., Bankert, R. L., & Brody, L. R. (2000). An AVHRR multiple cloud-type classification package. *Journal of Applied Meteorology*, *39*, 125−134.

Tang, B., & Mazzoni, D. M. (2006). Multiclass reduced-set support vector machines. *Proceedings of the 23rd International Conference on Machine Learning* Pittsburgh, PA, June 2006.

Tian, B., Azimi-Sadjadi, M. R., Vonder Haar, T. H., & Reinke, D. (2000). Temporal updating scheme for probabilistic neural network with application to satellite cloud classification. *IEEE Transactions on Neural Networks*, *11* (4), 903−920.

Welch, R., Sengupta, S., Goroch, A., Rabindra, P., Rangaraj, N., & Navar, M. (1992). Polar cloud and surface classification using AVHRR imagery: An intercomparison of methods. *Journal of Applied Meteorology*, *31*(5), 405−420.