Using Trained Pixel Classifiers to Select Images of Interest

D. Mazzoni,[1] K. Wagstaff,[1] and R. Castaño[1]

We present a machine-learning-based approach to ranking images based on learned priorities. Unlike previous methods for image evaluation, which typically assess the value of each image based on the presence of predetermined specific features, this method involves using two levels of machine-learning classifiers: one level is used to classify each pixel as belonging to one of a group of rather generic classes, and another level is used to rank the images based on these pixel classifications, given some example rankings from a scientist as a guide. Initial results indicate that the technique works well, producing new rankings that match the scientist's rankings significantly better than would be expected by chance. The method is demonstrated for a set of images collected by a Mars field-test rover.

I. Introduction

Mars rovers, such as the 2003 twin Mars Exploration Rovers or the 2009 Mars Science Laboratory, are designed to take images with multiple cameras, but they have very limited bandwidth to send these images back to Earth. As a result, only a small fraction of the images that the rovers are capable of capturing can ever be sent to Earth, even when utilizing various forms of image compression. In order to maximize the science return, the rover could take a number of images and onboard algorithms could analyze and rank the images. This could then be used to prioritize images for transmission back to Earth as bandwidth is available.

There are several approaches that could be used to analyze images for science content onboard a rover. One approach is to identify features in the images that explicitly represent scene properties. For example, an algorithm could identify rocks in an image, and then measure properties of the rocks using image-processing techniques such as edge detection and texture analysis. This is the approach that has been used in the development of the Onboard Autonomous Science Investigation System (OASIS) [4] to this point. While this approach can benefit from applying knowledge about a scene and properties, such as the existence and importance of rocks, soil, or other scene properties, it is heavily dependent on reliable extraction of the features and their properties. This approach suffers from some limitations in that the features need to be determined a priori and therefore may not generalize well to new scenes on the planet surface that contain new types of features. In this article, we describe a technique for characterizing and prioritizing images based on machine-learning methods that can be trained to recognize new features

[1] Exploration Systems Autonomy Section.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

and learn new prioritizations quickly without the extensive development effort usually associated with specialized feature detectors.

In Section II, we describe the machine-learning methods used to create a classifier model based on training data and the results of applying the classifier to test data. We then look at the details of both the pixel-classification layer and the image-ranking layer. We describe an experiment on image data and present experimental results in Section III. Finally, conclusions and areas for future work are discussed in Section IV.

II. Methods

The approach taken in this work is to apply pure machine-learning methods rather than image-processing techniques to identify the most important images in a set. We sequentially use two layers of classifiers. In the first layer, image pixels are individually classified as soil, rock, sky, etc. In the second layer, the overall classification of pixels in each image is used to identify which images are most interesting (to scientists).

Support vector machines (SVMs) [1] were used for classification. SVMs are similar to artificial neural networks (ANNs) in that they are supervised classifiers, i.e., they use labeled examples to build models (as opposed to unsupervised algorithms, which attempt to separate data into classes without any a priori information). The problem addressed by such classifiers is determination of the class to which a new object belongs given a set of examples and the classes to which they belong. Formally, given a set of training data containing m items

$$(x_1, y_1), \cdots, (x_m, y_m) \in R \times \{\pm 1\}$$

and a previously unseen query x_j, determine the corresponding y_j. The x's are feature vectors of dimensionality d. Each feature vector consists of values that capture key properties of an item that is to be classified. The y's are the labels, where we have defined a binary classification problem, i.e., two classes ($+1$ and -1). Supervised classification involves two phases: a training phase and a testing phase. During the training phase, a classifier uses the training data to either implicitly or explicitly learn a function mapping feature vectors to labels, $f(x) \rightarrow y$. Equivalently, the classifier learns the regions of the feature space that correspond to each class. The surface separating the regions is called the decision boundary. In the case of a support vector machine, learning involves identifying the planar boundary separating the feature vectors belonging to the two classes that has the maximum margin between the two classes (assuming separability of the classes). In the testing phase, the learned model is applied to new unlabeled feature vectors to determine the predicted class of the new items. For an SVM, this amounts to identifying on which side of the hyperplane a query point lies.

There are two parameters to an SVM: a kernel function and a regularization parameter. A kernel is a function that takes two feature vectors and returns a real number characterizing the similarity of the two vectors. Different kernels induce different similarity metrics. A similarity metric in which features in the same class are near to each other, but the distance between features in different classes is large, will have a simple decision boundary in the induced space. Often the feature vectors from different classes cannot be linearly separated in the input space; however, by applying a nonlinear mapping of the points to a higher dimension, it is possible to separate the points with a hyperplane in the higher dimension. SVMs use kernel functions to implicitly map feature vectors into a higher-dimensional space, where a linear separating hyperplane can be found. When projected back into the original feature space, the resulting decision boundary is nonlinear; thus, kernels enable SVMs to have nonlinear decision boundaries in the input feature space. There are many possible kernels, including different varieties of polynomial, sigmoid, and exponential [5]. No single kernel works best for all problems.

The second parameter to the SVM, the regularization parameter, essentially controls how rigidly the optimization procedure used in training should adhere to correctly classifying all of the training data. Training data may contain noisy or outlier data that are not representative of the underlying distribution. Fitting exactly to the training data may lead to a learned model that reflects characteristics of the training data set that are not representative of the underlying distribution from which data are sampled, i.e., overfitting, and can result in an increased error rate for new queries, thereby affecting generalization. On the other hand, dismissing true properties of the data as sampling bias in the training data will result in low accuracy. The regularization parameter is used to balance the trade-off between these two competing considerations. Setting the regularization parameter too low can result in poor accuracy, while setting it too high can lead to overfitting.

In this task, we used an automated procedure to select the SVM kernel and regularization parameter using cross-validation on two disjoint subsets of the training data for each SVM employed. By allowing a different choice of the SVM parameters for each of the different classification problems, we were able to increase the overall accuracy, since the parameters were tuned to each problem independently.

One advantage of SVMs is that the SVM training algorithm always converges to the mathematically optimal solution whereas ANN training algorithms require randomness, can get trapped in local minima, and may never converge to a solution. Recently, SVMs have been used to achieve lower error rates than any other classifier on benchmark data sets such as the Modified National Institute of Standards and Technology (MNIST) digit recognition task [2]. While highly accurate SVMs have been applied to a variety of learning tasks, the run-time classification typically takes longer than a neural network. This has led to the development of methods that dramatically speed up SVM evaluation [3].

We next describe how SVMs were used to classify image pixels. We then will discuss the use of the results of this stage for ranking a set of images with a separate prioritizing SVM.

A. Pixel Classification

The goal of the first stage was to classify each pixel in the image as one of a number of different classes of objects or materials that might be present in the scene. The classes we used for Mars Rover images were *sky, sediment, rock, layers*, and *shadow*. *Layers* refers to pixels on the horizontal boundary between two different layers of rock, a feature of particular interest to geologists. It is generally not possible to determine the class of a pixel given only its intensity, so the feature vector must include some context. We used feature vectors of size 128, consisting of the intensities of an 8×8 region surrounding each pixel, along with the discrete cosine transform (DCT) of the 8×8 region (Fig. 1). The output of the DCT is an 8×8 matrix of coefficients that represent texture information about the pixels, enabling the recognition of features such as horizontal or vertical stripes, gradients, etc.

To train the pixel classifiers, we selected six representative images from our main data set (described in detail below) and identified by hand several thousand pixels belonging to each class. Although multi-class SVMs exist, we chose to train a separate binary decision SVM for each class because of concerns that some pixels could plausibly belong to more than one class. The accuracy of each classifier on the pixels used as training data ranged from 81 percent (for *sediment*) to 98 percent (for *shadow*). We also evaluated the classifiers on two test sets (Table 1). Test set A was composed of pixels not used for training but taken from the same images as the training pixels. Test set B was composed of pixels from a novel image; accuracy values for the layer and shadow classifiers are not included, as the test image did not have any pixels of those types. Performance for all classes except sediment was relatively high. Sediment was difficult to distinguish from rock, as can be seen in Fig. 1, where local regions of the rock appear very similar to local soil regions.

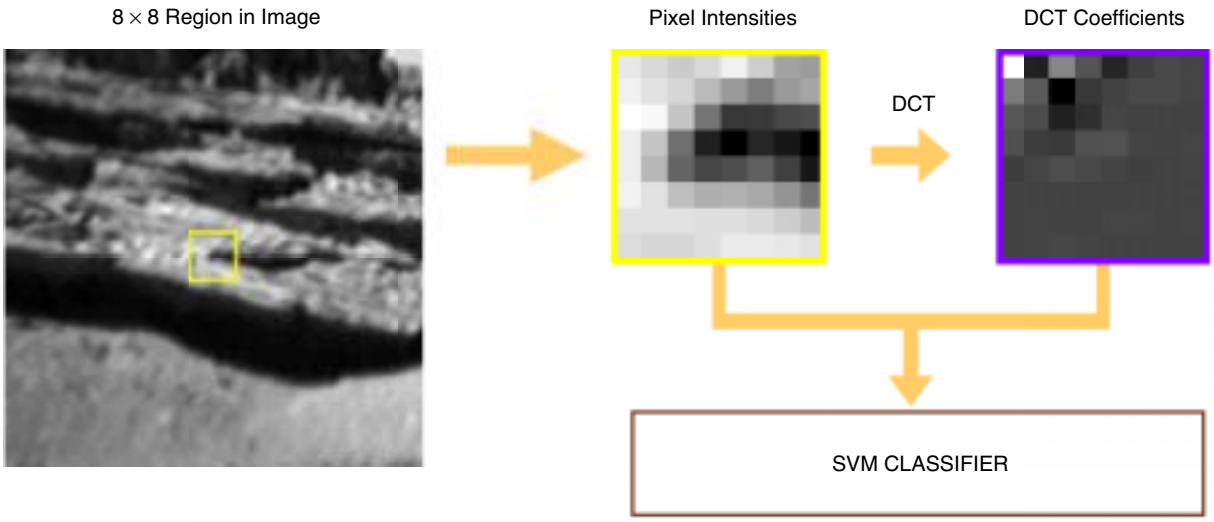

Fig. 1. Creating feature vectors for pixel classification.

Table 1. Accuracy of pixel-level classifiers.

Test set	Sky, percent	Sediment, percent	Rock, percent	Layers, percent	Shadow, percent
Training (3000 pixels)	91	81	92	94	98
Test set A (1500 pixels)	86	79	91	95	99
Test set B (302,713 pixels)	91	71	86	N/A	N/A

After using the individual SVMs to classify pixels, we combined the results from each of the classifiers to create a single multi-class classifier. In isolation, each SVM could indicate only whether the given pixel was or was not a member of a single class. One method to classify a pixel into one of multiple classes would be to assign it to the class whose SVM produced the highest confidence value. The confidence in a classification is estimated as the distance from the decision boundary. However, the distance is a function of a number of factors, including the kernel and the training points. Therefore, confidences between SVMs are not comparable.

Instead, we prioritized the SVMs based on the observed frequency of each class and the empirical performance of its associated classifier, resulting in the following order: *sky, sediment, rocks, shadow, layers*. In other words, the *sky* SVM output was checked first, and if it was positive, the pixel was classified as *sky*. Next, the *sediment* SVM was checked, and so on. If no other SVM matched, then the class of the pixel was assumed to be *soil*. The results of the classification on a representative image are shown in Fig. 2.

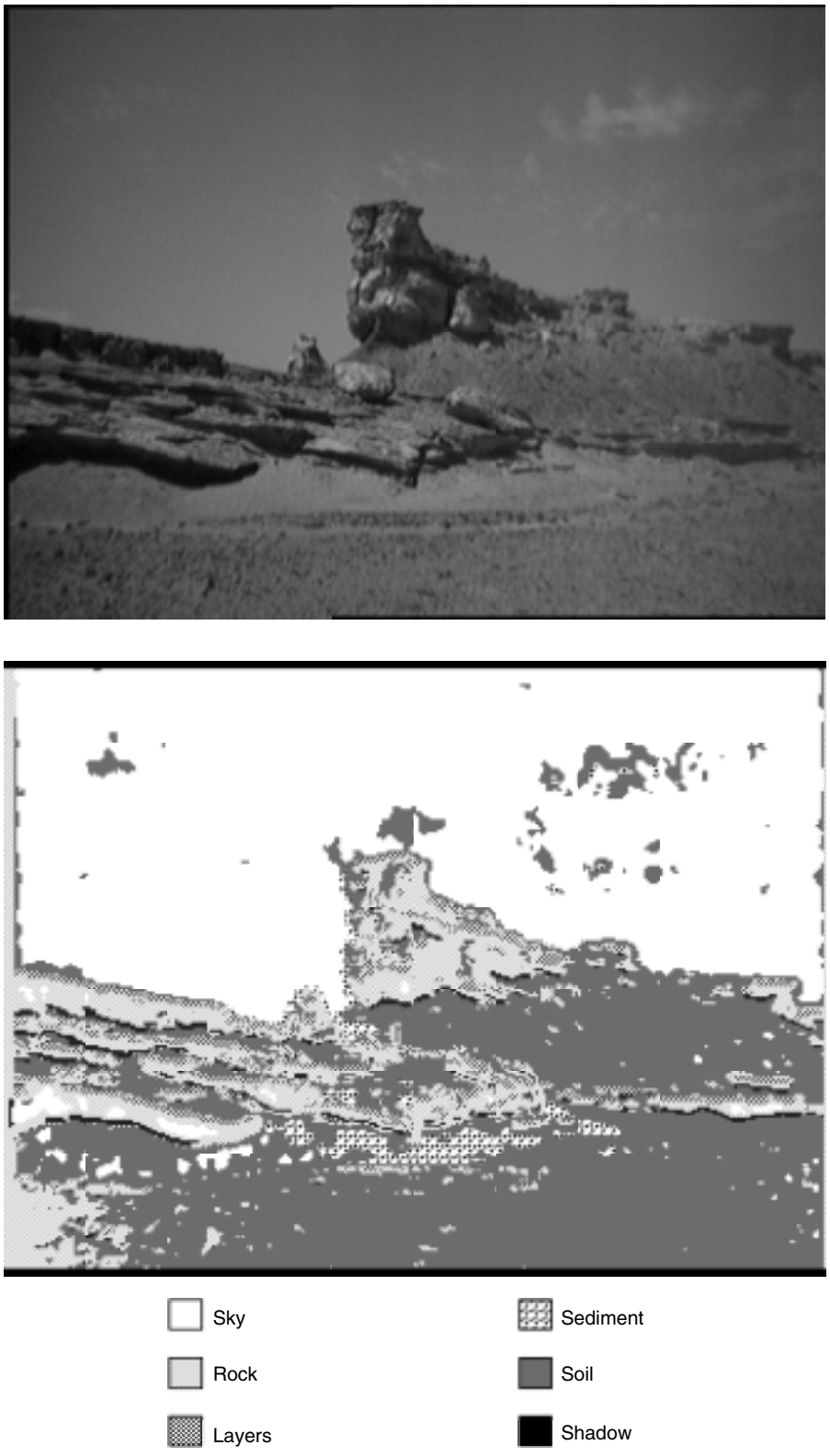

Fig. 2. Results of combined pixel classification on an image taken by the FIDO rover in Arizona.

B. Image Comparison

The second stage of evaluation is to rank the images using the pixel-classification results. A feature vector was derived for each image based on the pixels in the image. This feature vector indicated the percentage of pixels assigned to each of the six classes (including *soil*). For example, the image in Fig. 2 was classified as 45 percent sky, 2 percent sediment, 13 percent rocks, 1 percent shadow, 3 percent layers, and 33 percent soil. These summary features were used, as described in the next section, to automatically prioritize the images. We assessed the accuracy of these features in a qualitative way by comparing the relative percentages identified between two images. For example, test set image B has more sky in it than another image, C, that is a close-up view of sand and rocks. The automatically generated image-level features for B and C correctly capture this relationship. Indeed, the relative percentages were correct for all features but sediment, which, as shown in Table 1, is the least accurate of the pixel-level classifiers. Therefore, we conclude that the features are providing reasonable image summary information.

III. Experimental Results

To quantify the performance of our overall system, we conducted an experiment in which we compared the top images selected by our system to the top images chosen from the same data set by a geologist. The data set consisted of 100 images taken by a navigation camera on the Field Integrated Design and Operations (FIDO) rover in Arizona in August 2002. These images included both horizon shots and close-ups of the ground. We had an expert field geologist examine the images and choose the 20 that would have the highest priority to downlink based on science value, i.e., the 20 images with the most important science content. Our goal was to see how closely we could match those selections with an automated process.

Our test procedure consisted of randomly splitting the 100 images into two groups of 50, each with 10 of the most important images and 40 others. The first set of images was used as the training set for an image-level SVM classifier. Each training image was marked as either high priority or low priority. We tested the SVM by having it identify the best images out of the 50 in the hold-out set not used for training. Since there is no way to guarantee that the SVM would classify exactly 10 images as high priority out of the 50 images in the test set, we used the confidences of the SVM output to rank the 50 images and selected the top 10.

We performed the experiment 200 times, using different random splits of 50 training and 50 test images for each experiment. The SVM on average correctly selected five of the ten high-priority images, with a standard deviation of 1.2. An agreement on two images out of ten would be expected by random chance. The SVM's performance is significantly above this level. In addition, we asked another human expert to independently select the top 20 images from the full set of 100. The two human experts agreed on 9 of the 20 they each selected, which is very close to the agreement between the first expert and the SVM-based system.

IV. Conclusions and Future Work

While the SVM-based system and the second human expert were performing somewhat different tasks (the human expert was evaluating the images based on his own perception of the science content, while the SVM system was attempting to learn the first expert's perception of science content and evaluate the images based on this measure), we are able to conclude that the machine learning approach was capable of achieving an acceptable level of accuracy in ranking images based on scientific value, in that the system agreed with the experts as much as the experts were consistent amongst themselves.

This work has demonstrated that pixel classifiers can have relatively high error rates while still being very useful. Half of the pixels in an image could be misclassified, but if the misclassifications tend to

be more random then systematic, very little information is lost when examining the overall statistics of an image. The concept of using a low-accuracy classifier contrasts with most classifier work, where researchers and practitioners place a strong emphasis on achieving the highest levels of accuracy.

By using low-level image properties rather then extracting higher-level image features, our system is able to answer only very basic questions about the images. For example: Does the image have more rock or more soil? Does it show lots of evidence of layers? Is there anything on the horizon or is it mostly just an image of sky? However, knowing the answers to these questions was sufficient to select images with a level of agreement comparable to that of human experts.

While this experiment was quite successful, it was limited in scope. More work is required before these techniques can be used as part of a robust onboard image-classification system. For example, since all of the training and test images were of the same site, it is likely that the pixel classifiers are not sufficiently robust to perform well on images taken at different sites and under different conditions. Further research will investigate how well this method compares to previous methods that use predetermined features, and how well this method works on a more diverse data set.

There are several possible directions for future work. Alternative approaches that we are considering include (1) ranking images more precisely and (2) expanding the information contained in the feature vectors. In the present work, although the confidence in the classification of an image was used for ranking, the classifier was trained on binary data. Each training image was labeled as either high priority or low priority. An area of future work is to have the expert fully rank the images, via either a complete sorting or a partial sorting, and train a classifier to learn the ranking function.

We are also currently investigating adding more information to the image feature vectors. In addition to the percentage of the image that was assigned to each class, other information about the spatial distribution of the pixels in that class might be useful. For example, two images might both be 20 percent rock, but one image might contain one big rock, while the other might contain dozens of small rocks, and it would be useful to distinguish between the two. For example, a single number that would represent the degree of proximity of pixels belonging to each class could be derived.

Acknowledgments

The support vector machine training algorithms were written by Dennis DeCoste. Thanks to Bob Anderson and Albert Haldemann for providing the ground truth, and to Michele Judd and Ben Bornstein for helpful comments and discussion.

References

[1] B. E. Boser, I. M. Guyon, and V. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory (COLT)*, D. Haussler, Ed., Pittsburgh, Pennsylvania, pp. 144–152, July 1992.

[2] D. DeCoste and B. Schölkopf, "Training Invariant Support Vector Machines," *Machine Learning*, vol. 46, pp. 161–190, 2002.

[3] D. DeCoste and D. Mazzoni, "Fast Query-Optimized Kernel Machine Classification Via Incremental Approximate Nearest Support Vectors," *Proceedings of the 2003 International Conference on Machine Learning (ICML)*, Washington, D.C., pp. 115–122, August 2003.

[4] R. Castaño, R. C. Anderson, T. Estlin, D. DeCoste, F. Fisher, D. Gaines, D. Mazzoni, and M. Judd, "Rover Traverse Science for Increased Mission Science Return," *Proceedings of the 2003 IEEE Aerospace Conference*, Big Sky, Montana, pp. 3629–3636, March 2003.

[5] B. Schölkopf, C. Burges, and A. Smola, Eds., *Advances in Kernel Methods—Support Vector Learning*, Cambridge, Massachusetts, MIT Press, 1999.