Support Vector Machine Classifiers as Applied to AVIRIS Data

J. A. Gualtieri[a], S. R. Chettri[a], R. F. Cromp[b], L. F. Johnson[c]

[a]Global Science and Technology at Applied Information Sciences Branch
Code 935, NASA/GSFC, Greenbelt, MD 20771, U.S.

[b]Earth and Space Data Computing Division
Code 930, NASA/GSFC Greenbelt, MD 20771, U.S.
now at E-systems, Raytheon, Dallas, TX

[c]Ecosystem Science and Technology Branch
NASA/ARC, Moffett Field CA 94035-1000, U.S.

To be published in : Summaries of the Eighth JPL Airborne Earth Science Workshop, February 8-11, 1999

1. INTRODUCTION

The Support Vector Machine (SVM) is a relatively recent approach introduced by Boser, Guyon, and Vapnik (Boser *et al.*, 1992), (Vapnik, 1995) for solving *supervised* classification and regression problems, or more colloquially learning from examples. In the following we will discuss only classification and its application to hyperspectral data from AVIRIS.

Traditionally, classifiers model the underlying density of the various classes and then find a separating surface. However density estimation in high-dimensional spaces suffers from the Hughes effect (Hughes, 1968), (Landgrebe, 1999): For a fixed amount of training data the classification accuracy as a function of number of bands reaches a maximum and then declines, because there is limited amount of training data to estimate the large number of parameters needed. Thus usually, a feature selection step is first performed on the high-dimensional data to reduce its dimensionality.

As we will demonstrate, the SVM approach does not suffer this limitation and uses the full dimensionality of the hyperspectral data. Support Vector Machines directly seek a separating surface through an optimization procedure that finds the exemplars that form the *boundaries* of the classes. These exemplars are called the *support vectors*. This is significant because it is usually the case that there are a small subset of all the training data that are involved in defining the separating surface, i.e., those examples that are closest to the separating surface.

In addition, the Support Vector Machine approach uses the kernel method, discussed below, to map the data with a non-linear transformation to a *higher dimensional* space and in that space attempts to find a *linear* separating surface between the two classes. The transformation to a higher dimensional space tends to spread the data out in a way that facilitates the finding of a linear separating surface. In this way the separating surfaces that would be non-linear (not a hyperplane) in the original data space can become linear (a hyperplane) in the higher dimensional space. Instead of being penalized by the *curse of dimensionality* and its attendant Hughes effect, the Support Vector Machine can use the full dimensionality of the hyperspectral data without the feature selection preprocessing step. Why the curse of dimensionality is not a problem for the kernel method is discussed below.

A number of useful introductions are available in publications and on the world wide web (Burges, 1998), (Gunn, 1997), (Support-Vector-Machines, 1999). In what follows we will first focus on *binary* classification

Further author information: (Send correspondence to J. A. Gualtieri)
J.A. Gualtieri: E-mail: gualt@peep.gsfc.nasa.gov
S. R. Chettri: E-mail: chettri@yellowsnow.gsfc.nasa.gov
R.F. Cromp: E-mail: cromp@sauquoit.gsfc.nasa.gov
L. F. Johnson: E-mail: Ljohnson@mail.arc.nasa.gov

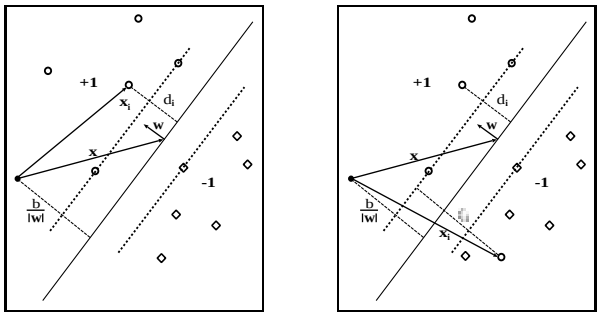

Figure 1. A. Schematic for separable data in $\mathbf{R}^2$. B. Schematic for non-separable data. in $\mathbf{R}^2$. The circles are feature vectors in class $+1$ and the diamonds are feature vectors in class -1. There is one feature vector that is not separable in the non-separable case.

– in the class or not in the class. Subsequently we will handle multiple classes by building separate classifier for each pair of classes and follow this with a voting strategy to choose the class label.

The plan for the paper is to give an overview of the mathematical formulation for binary classification. Then we introduce the optimal margin hyperplane, the transformation of its resulting optimization problem by means of Lagrange multipliers, and its solution. This is done for both the separable and non-separable cases. We then discuss the kernel method and the generalization to multiple classes. Following this, a section is devoted to describing the hyperspectral data we have used for demonstrating the classifier. Then we discuss implementation details and present the results. The conclusion summarizes the results and suggests further development of the method.

2. MATHEMATICAL FORMULATION

In the following we will highlight the mathematical formulation. More complete details can be found in a recent publication of Gualtieri and Cromp. (Gualtieri & Cromp, 1998).

2.1. Classification

For classification, a set of examples consisting of pairs of class labels and feature vectors is known, and you desire to find a classifier function that gives correct answers on these examples and has low generalization error, meaning it gives good results for the class labels when applied to feature vector inputs it has not seen before. We are given l training pairs, $(y_i, \mathbf{x}_i)$ $i = 1, \ldots l$, consisting of class labels, $y_i \in \{1, -1\}$, and n-dimensional feature vectors, $\mathbf{x}_i \in \mathbf{R}^n$. We wish to find a function $f(\ \ ; \boldsymbol{\alpha}) : \mathbf{x} \mapsto y$ that represents the classifier $y = f(\mathbf{x}; \boldsymbol{\alpha})$, where $\boldsymbol{\alpha}$ are all the parameters of the classifier.

2.2. Optimal Margin Method for Separable Data

Vapnik and Chervonenkis (Vapnik & Chervonenkis, 1974), and Vapnik (Vapnik, 1982) originated the optimal margin method for separable data. With reference to Fig. 1A. the problem is how to place a hyperplane such that: (1) All data belonging to class $+1$ lies on one side of the hyperplane and all data belonging to class -1 lies on the other side. (2) The hyperplane is placed so that the distance of the closest vectors in both classes to the hyperplane, called the margin, is the largest it can be. The hyperplane is defined by the equation $\mathbf{w} \cdot \mathbf{x} + b = 0$, where $\mathbf{x}$ is a point on the hyperplane, $\mathbf{w}$ is the n-dimensional vector perpendicular to the hyperplane, and b is the distance of the closest point on the hyperplane to the origin. The classifier is then $f(\mathbf{x}; \mathbf{w}, b) = sgn(\mathbf{w} \cdot \mathbf{x} + b)$. These requirements can be shown to lead to an optimization problem:

$$\min_{\mathbf{w}, b} \ \frac{1}{2}|\mathbf{w}|^2, \quad y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0 \quad i = 1, \ldots, l. \tag{1}$$

2.3. The Dual Optimization Problem for Separable Data

The quadratic optimization problem in Eq. (1) can be simplified so as to replace the inequalities with a simpler form by transforming the problem to a *dual* space representation using Lagrangian multipliers. We absorb the constraints into the minimization problem by defining Lagrange undetermined multipliers. $\lambda_i \geq 0 \quad i = 1, \ldots l$, which multiply the constraints and are subtracted from the original minimization function giving a new quantity to be optimized: $\mathcal{L}(\mathbf{w}, b, \lambda_1, \ldots, \lambda_l) = \frac{|\mathbf{w}|^2}{2} - \sum_{i=1}^{l} \lambda_i \ [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1]$. This gives the dual optimization problem,

$$\underset{\lambda_1 \ldots \lambda_l}{\mathbf{max}} \quad \underset{\mathbf{w}, b}{\mathbf{min}} \ \mathcal{L}(\mathbf{w}, b, \lambda_1, \ldots, \lambda_l), \quad \lambda_i \geq 0 \quad i = 1, \ldots, l, \quad y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0 \quad i = 1, \ldots, l. \tag{2}$$

Assuming that $\mathcal{L}$ is a differentiable function of $\mathbf{w}, b$, we then have after performing partial derivatives, the conditions for a minimum, $\mathbf{w} = \sum_{i=1}^{l} \lambda_i y_i \mathbf{x}_i, \ \sum_{i=1}^{l} \lambda_i y_i = 0$, which when substituted in Eq. (2) allows us to eliminate $\mathbf{w}$ and b and to obtain an equivalent quadratic optimization problem. This is called the *dual* problem optimization and it has simplified constraints:

$$\underset{\lambda_1 \ldots \lambda}{\mathbf{max}} \left[-\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i y_i (\mathbf{x}_i \cdot \mathbf{x}_j) y_j \lambda_j + \sum_{i=1}^{l} \lambda_i \right], \quad \lambda_i \geq 0 \quad i = 1, \ldots, l, \quad \sum_{i=1}^{l} \lambda_i y_i = 0. \tag{3}$$

In the process of obtaining a solution, we expect that some of the λ_i will be 0, and the remaining ones will be associated with the support vectors. From the solution for the λ_i we obtain $\mathbf{w}$, from $\mathbf{w} = \sum_{i=1}^{l} \lambda_i y_i \mathbf{x}_i$, and b from Eq. (2). What is often found is that the number of support vectors (vectors for which $\lambda_i > 0$) is not dependent on the dimensionality, n, of the vectors, but reflects an intrinsic dimension for classification in the space of the training vectors. This is at the root of why the curse of dimensionality is *not* a problem in this method. Methods of solving the optimization problem are taken up in the section on implementation.

2.4. Non-separable Data

Cortes and Vapnik generalized the optimal margin methods to non-separable data (Cortes, 1995), (Cortes & Vapnik, 1995) which we discuss next. With reference to Fig. 1B the problem is now that there is no way to place a hyperplane such that we can separate the training data into two classes.

Cortes and Vapnik (Cortes, 1995), (Cortes & Vapnik, 1995) gave the following solution and named it the soft margin classifier. They relaxed the restriction that every training vector of a given class lie on the same side of the optimal hyperplane, by introducing a penalty whose cost is controlled by a parameter C. The limit $C \to \infty$ means the optimization reduces to the formulation for the separable case, and the limit $C \to 0$ means no extra cost if a training vector lies on the opposite side of the separating hyperplane. Using a generalization of the optimization formulation given above and again transforming to the dual representation using Lagrange undetermined multipliers we obtain,

$$\underset{\lambda_1 \ldots \lambda_l}{\mathbf{max}} \left[-\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i y_i (\mathbf{x}_i \cdot \mathbf{x}_j) y_j \lambda_j + \sum_{i=1}^{l} \lambda_i \right], \quad C \geq \lambda_i \geq 0 \quad i = 1, \ldots, l, \quad \sum_{i=1}^{l} \lambda_i y_i = 0 \quad i = 1, \ldots, l \tag{4}$$

Note the only difference from the dual of the separable case, Eq. (3), is that λ_i are bounded above by C. Knowing the solutions λ_i, we can find $\mathbf{w}$ for the hyperplane as above and similarly b can be found using one or more of the equations (not given here) that come from the using the dual formulation.

2.5. Kernel Method

Up to this point we have only dealt with classification as a linear function of the training data – the decision surface is a hyperplane defined by linear equations on the training data. However, it can be the case that no hyperplane exists to separate the data. The non-separable method provides one way to deal with this. As an alternative we would like a way to build a *non-linear* decision surface. An extremely useful generalization which can give non-linear decision surfaces and improved separation of the training data is possible using the following idea, first introduced by Aizerman, Braverman and Rozoner (Aizerman *et al.*, 1964), and incorporated into machine learning as part of the Support Vector Machine by Boser, Guyon, and Vapnik (Boser *et al.*, 1992).

Note the way that the training data enters the optimization problems, Eqs. (3), (4), is as dot products. Suppose that we map the feature vectors, $\mathbf{x} \in \mathbf{R}^n$ into a *higher* dimensional Euclidean space, $\mathcal{H}$, by means of a non-linear vector function $\mathbf{\Phi} : \mathbf{R}^n \mapsto \mathcal{H}$. Then we may again pose the optimal margin problem in the space $\mathcal{H}$ by replacing $\mathbf{x_i} \cdot \mathbf{x_j}$, by $\mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j})$. Then, as before, solve the optimization problem for the λ_i. This finds the support vectors among the transformed vectors, $\mathbf{\Phi}(\mathbf{x_i})$, by association with the $\lambda_i > 0$. We then use these to build the classifier function:

$$f(\mathbf{x}, \lambda_1, \ldots, \lambda_l) = sgn\left(\sum_{i=1}^{l} \lambda_i y_i \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x}) + b\right). \tag{5}$$

Now suppose there exists a *kernel function* K such that $K(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j})$, then everywhere $\mathbf{x_i} \cdot \mathbf{x_j}$ occurred, we could replace it with $K(\mathbf{x_i}, \mathbf{x_j})$. We need not explicitly compute $\mathbf{\Phi}(\mathbf{x})$, which could be computationally expensive, but only need compute the kernel functions. In fact we need not have an explicit representation of $\mathbf{\Phi}$ at all, but only K. The restrictions on what functions can qualify as kernel functions is discussed in Burges (Burges, 1998). What is gained is that we have moved the data into a larger space where the training data may be spread further apart and a larger margin may be found for the optimal hyperplane. In the cases where we can explicitly find $\mathbf{\Phi}$, then we can use the inverse of $\mathbf{\Phi}$ to construct the non-linear separator in the original space. Clearly there is a lot of freedom in choosing the kernel function and recent work has gone into the study of this idea both for SVM and for other problems (Smola *et al.*, 1998). With respect to the curse of dimensionality, we never explicitly work in the higher dimensional space, so we are never confronted with computing the large number of vector components in that space.

For the results presented below, we have used the inhomogeneous polynomial kernel function

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d, \tag{6}$$

with $d = 7$, though we found little difference in our results for $d = 2, \ldots, 6$ (See Fig. 2B). The choice of the inhomogeneous polynomial kernel is based on other workers success using this kernel function in solving the handwritten digit problem (Boser *et al.*, 1992). In fact, there are principled ways to choose among kernel functions and to choose the parameters of the kernel function. Vapnik (Vapnik, 1995), (Vapnik, 1998) has pioneered a body of results from probability theory that provide a principled way to approach these questions in the context of the Support Vector Machine.

2.6. Multi-Class Classifiers

Two simple ways to generalize a binary classifier to a classifier for K classes are: (1) Train K binary classifiers, each one using training data from one of the K classes and training data from all the remaining $K-1$ classes. Apply all K classifier to each vector of the test data, and select the label of the classifier with the largest margin – the value of the argument of the *sgn* function in Eq. (5). (2.) Train $K(K-1)/2$ binary classifiers on all pairs of training data. Apply all $K(K-1)/2$ binary classifiers to each vector of the test data and for each outcome give one vote to the winning class. Select the label of the class with the most votes. For a tie, apply a tie breaking strategy. We chose the second approach, and though it requires building more classifiers, it keeps the size of the training data smaller and is faster for training.

3. HYPERSPECTRAL DATA

In this work, hyperspectral data was obtained from the AVIRIS imaging spectrometer for two scenes: (1) Indian Pines 92 from Northern Indiana on June 12, 1992 taken on a NASA ER2 flight at high altitude with a ground pixel size of 17m resolution; (2) Salinas 98 taken in the Salinas Valley, California on October 9, 1998 from a NOAA Twin Otter flight at low altitude with a pixel size of 3.7m.

3.1. Indian Pines 1992 Data

The Indian Pines data consists of 145×145 pixels by 220 bands of reflectance data with about two-thirds agriculture, and one-third forest or other natural perennial vegetation. There are two major dual lane highways, a rail line, as well as some low density housing, other built structures, and smaller roads. Since the scene is taken in June some of the crops present, corn, soybeans, are in early stages of growth with less than 5% coverage. The ground truth available is designated into sixteen classes and is not all mutually exclusive. The data and the ground truth are available from D. Landgrebe (Landgrebe, 1992), (Landgrebe, 1998). This scene has been recently studied by S. Tadjudin and D. Landgrebe (Tadjudin & Landgrebe, 1998a), (Tadjudin & Landgrebe, 1998b) and allows us a comparison to their classifier. Following their work we have looked at: (1) A part of the 145×145 scene, called the subset scene, consisting of pixels $[27-94] \times [31-116]$ for a size of 68×86. [Upper left in the original scene is at $(1,1)$]. There is ground truth for over 75% of this scene and it is comprised of the three row crops, Corn-notill, Soybean-notill, Soybean-mintill, and Grass-Trees. (2) The full 145×145 scene for which there is ground truth covering 49% of the scene and it is divided among 16 classes ranging in size from 20 pixels to 2468 pixels.

Following Tadjudin and Landgrebe's work, we have also reduced the number of bands to 200 by removing bands covering the region of water absorption: $[104-108], [150-163], 220$.

3.2. Salinas 1998 Data

This scene, just south of the city of Greenfield in the Salinas Valley in California, was acquired on October 9, 1998 (Johnson, 1998). This data was available only as at-sensor radiance data. It includes vegetables, bare soils, and vineyard fields with sub-categories as given in the following table. In Fig. 3, on the perimeter, are images of all the classes on the day the data was taken.

Vegetables				Soils and Ground	Vineyards
brocoli	romaine lettuce	celery	corn_senesced	fallow	vineyard_untrained
brocoli_green_weeds_1	4 wk		and green weeds	fallow_rough_plow	vineyard_vert_trellis
brocoli_green_weeds_1	5 wk			fallow_smooth	grapes_untrained
	6 wk			stubble	
	7 wk			soil_vineyard_dev.	

The sub-categories of brocoli and green weeds are distinguished with 1 having smaller and fewer weeds while 2 has taller and more weeds with both categories mostly covering the soil. The romaine lettuce is at different weeks since planting and with growth increasingly covering the soil. The fallow_rough_plow class has recently been turned with larger clumps and appears more moist while the fallow class is plowed soil with smaller clumps and the fallow_smooth class has even smaller clumps. The stubble class is bare soil with stubble. The class soil_vineyard_develop is mostly finely plowed bare soil with occasional green weeds of a few inches height and a regular array of vertical wood posts and plastic posts. The vineyard_untrained class shows untrained vines growing on wood and plastic posts with their canopy spreading out to cover most of the soil while the vineyard_vert_trellis class has much greater soil visibility and much smaller grape canopy. The two classes grapes_untrained, and vineyard_untrained have been treated as different in the analysis below, although after this work was completed, it was learned that these two classes were in fact the same.

Two sub-scenes were used: (1)Salinas 98 A, B: Sub-scenes comprising 86×83 pixels located within the same scene at [samples, lines] = [591 - 676, 158 - 240] which include the six classes: brocoli_green_weeds_1, corn_senesced_green_weeds, lettuce_romaine_4wk, lettuce_romaine_5wk, lettuce_romaine_6wk, lettuce_romaine_7wk. (2) Salinas 98 C: A subset comprising 217×512 pixels located within the same scene at [samples, lines] = [216 - 432, 1 - 512] which includes all 16 classes described above.

Figure 3. Around the perimeter are photographs of the the sixteen classes in the Salinas C scene. In the center is the classification of the whole scene, where the regions for which the ground truth was given are delineated with black polygons. The colors in the boxes surrounding the class names are the colors for the classes.

3.3. Data Preprocessing and Post-processing

For each band at each pixel in all the scenes used, the data was rescaled from the input two byte short integer by dividing by 10000 to make a floating point number in the range $[0, 1]$. Then the data was *centered*, which means that for the scene, for each band, the mean was found and this was subtracted from all the data in that band. This distributes the data around 0 and considerably speeds up the optimization routines.

To establish the performance of the support vector classifier a fixed percent of all the known ground truth was randomly chosen as the training data and used to train the classifier. The resulting classifier was then applied to the whole scene at hand, but the performance was only measured on the the testing set, which was those ground truth pixels less the training pixels. Thus the performance figures are always on pixels the classifier has not seen before. This approach was repeated over several trials using different random choices of training data: five for the subset Indian Pines scene, one for the full Indian Pines scene, five for the Salinas 98 subset scenes A and B, and two for the Salinas 98 C subset scene.

4. APPLICATION OF SVM TO HYPERSPECTRAL DATA AND RESULTS

4.1. SVM Implementation

We have implemented the Support Vector Machine by incorporating software from T. Joachims (Joachims, 1998a). This code consists of a learning module that finds the support vectors given two set of training vectors and a specification of the non-separable parameter C and the kernel function (in this work a polynomial with parameter d), and a classification module that classifies any test vector into one of the pair of classes. We have generalized these codes to handle multiple classes, creating $K(K-1)/2$ such classifiers in the learning phase and then using the voting strategy after applying all these classifiers to each test vector.

Central to the learning module is the quadratic optimization as formulated above in Eqs. (3), (4). Direct application of quadratic optimization to large numbers of training vectors can be computationally slow, but Joachim (Joachims, 1998b) has shown a way to reformulate the problem as a series of smaller optimization problems. The solution of these smaller optimizations is accomplished using a quadratic optimization code written by A. Smola (Smola, 1998).

4.2. Results for Indian Pines 92 Subset Scene

From the subset scene, a random sample of 20% of the pixels was chosen from the known ground truth of the four classes: Corn-notill, Soybean-notill, Soybean-mintill, Grass-Trees. This was used to train six binary classifiers, one for each pair of classes. The trained classifiers were then applied to the remaining 80% of the known ground pixels in the scene, with the voting strategy above. Ties were broken by a random choice. This procedure was repeated in five trials using a different random seed for the selection of the 20% of the training data. Fig. 2A shows the training data for one of the five trials and shows because of the spectral overlap of the three row crops that this is a challenging classification problem. Table 1 shows the contingency table for a typical trial. For a trial, the overall performance is the sum of the number of samples correctly labeled for each class in the test set divided by the total number of samples in the test set. Note that Grass-Trees was classified almost completely correctly as might be expected from the lack of overlap in the training data, and the the results for the row crops is also good.

The results across the five trials were consistent within one percent and the average performance was 96%, which is somewhat better than 93% from recent results of Tadjudin and Landgrebe (Tadjudin & Landgrebe, 1998a) for their best classifier, bLOOC+DAFE+ECHO, on the same data. Table 3 summarizes the results for the subset scene comparing the SVM classifier, bLOOC+DAFE+ECHO, and a simple Euclidean classifier (Tadjudin & Landgrebe, 1998a). The Euclidean classifier uses only the first order statistics of the training data. Its poor performance is expected for this data due to the overlap of the classes. The details of the bLOOC+DAFE+ECHO classifier is covered in Tadjudin and Landgrebe (Tadjudin & Landgrebe, 1998a).

Class Name	Users Acc.	Number of Samples	Corn-notill	Soybean-notill	Soybean-mintill	Grass-Trees
Corn-notill	94.3	807	**761**	4	38	4
Soybean-notill	95.7	582	1	**557**	23	1
Soybean-mintill	96.1	1541	39	21	**1481**	0
Grass-Trees	100	586	0	0	0	**586**
Number Class.		3516	801	582	1542	591
Produc. Acc.			95.0	95.7	96.0	99.2
OVERALL PERF.	**96.3**					

Table 1. A typical result for the Indian Pines subset scene. The entries in rows 2 - 5 and colums 4 - 7 are the contingency table for test-set results. Bold face numbers are correctly classified samples. A perfect result would have all zeros except on the diagonal. The overall performance is computed by the ratio of the sum of the diagonal elements to the sum of all entries of the contingency table and appears in the lower left

4.3. Effect of Dimension Parameter d

To test the effect of changing the dimension parameter d in the kernel function, Eq. (6), the five Indian Pines 92 Subset scenes with different random choices of training vectors was tested for performance using a kernel function with $d = 1, 2, \ldots, 15$ and $C = 1000$. The results are shown in Fig. 2B. A significant increase in performance occurs by going to a non-linear classifier with $d = 2$. The five lines are for each of the different random choices of training vectors respectively.

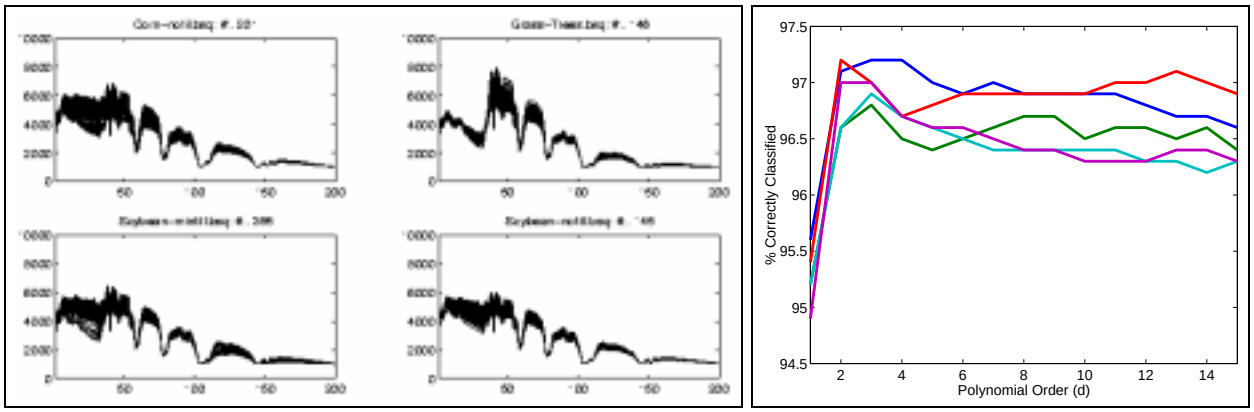

Figure 2. A. (left) Training data for classification in the Indian Pines 92 subset scene. B. (right) Effect on classifier performance by varying the kernel parameter d, in $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d$ The five lines are the results for each of the different random choices of training vectors respectively

4.4. Results for Indian Pines 92 Full Scene

Results for the full scene were produced using only one trial. Here we used the sixteen ground truth classes given in Landgrebe's data (Landgrebe, 1992). We made a random selection of 20% of the ground truth data and tested on the remaining 80%. A difference with the data and results reported by Tadjudin and Landgrebe (Tadjudin & Landgrebe, 1998a), (Tadjudin & Landgrebe, 1998b) is that they studied the scene using 17 classes whereas we only used 16. The difference being that they further resolved the class Soybeans-notill into two subclasses of Soybeans-notill based on fields that were in different locations in the full scene. The results reported in Table 3 show the Support Vector Machine to be somewhat better at 87% as compared to 83% for the bLOOC+DAFE+ECHO classifier, although the difference in the number of classes may have some effect.

4.5. Results for Salinas 98 A

Training using 10% of the ground truth and testing on the remaining 90% gave an average performance of 99.5% over five different trials with different random choice of training vectors.

Class Name	Users Acc.	Number of Samples	brocoli green weeds 1	corn senes. green weeds	lettuce romaine 4wk	lettuce romaine 5wk	lettuce romaine 6wk	lettuce romaine 7wk
brocoli green weeds 1	100.0	388	**388**	0	0	0	0	0
corn senes. green weeds	97.3	1330	0	**1294**	30	3	0	3
lettuce romaine 4wk	99.7	610	0	0	**608**	2	0	0
lettuce romaine 5wk	99.1	1510	0	2	11	**1497**	0	0
lettuce romaine 6wk	99.7	668	0	0	0	0	**666**	2
lettuce romaine 7wk	97.6	792	0	9	0	0	10	**773**
TOTAL		5298	388	1305	649	1502	676	778
Producers Acc.			100	99.2	93.7	99.7	98.5	99.3
OVERALL Perf.	**98.6**							

Table 2. A typical result for Salinas 98 B. The contiingency table has the same meaning as in Table 1.

INDIAN PINES		
METHOD	PERFORMANCE	
	Subset Scene	Full scene
Support Vector Machine	95.9%	87.3%
bLOOC+DAFE+ECHO	93.5%	82.9%
Euclidean	66.7%	48.2%

SCENE	PERCENT TRAINING	PERFORM.
Indian Pines 92 Subset	20 %	95.9%
Indian Pines 92 Full	20 %	87.3%
Salinas 98 Subset A	20 %	99.5 %
Salinas 98 Subset B	1 %	98.2 %
Salinas 98 Subset C	1 %	89 %

Table 3. On the left a comparison of results for the Indian Pines subset scene. The results labeled bLOOC+DAFE+ECHO and Euclidean are taken from the recent work of Tadjudin and Landgrebe (Tadjudin and Landgrebe 98a, 98b). On the right the fraction of ground truth used for training and performance results for all the scenes treated.

4.6. Results for Salinas 98 B

Training using 1% of the ground truth and testing on the remaining 99% gave an average performance of 98.2% over five different trials with different random choice of training vectors. A typical contingency table is shown in Table 2. In particular the classifier can readily distinguish lettuce at 4,5,6, and 7 weeks from planting, although this primarily reflects different ratios of plant coverage to bare soil.

4.7. Results for Salinas 98 C

Training using 1% of the ground truth and testing on the remaining 99% gave an average performance of 89% over two trial with different random choices of training vectors. The results of classifying the whole image for one of these cases is shown in Fig. 3. In this figure (as in all other trials) the performance was measured *only* on the testing pixels which comprise 99 % of the pixels in the regions *bounded by the black lines*. In each such region the majoirty color is the correct class. The classification for the rest of the image cannot be checked since we do not currently have ground truth for those pixels.

4.8. Summary of Performance Results

In Table 3 are assembled all the results. The comparisons with Tadjudin and Landgrebe show the Support Vector method to be competitive with what is to our knowledge the best current results for classification of hyperspectral data. And the results for the Salinas scenes show the method to give good results even for small training set sizes.

5. CONCLUSIONS

We have described a new approach to building a supervised learning machine called the Support Vector Machine, and applied it to classify hyperspectral remote sensing data. The inherent high dimensionality of this data is challenging for traditional classifiers, due to the Hughes effect, and usually a feature selection preprocessing step is performed to first reduce the dimensionality of the data. The Support Vector Machine

does not suffer from this handicap, and is thus suitable for use with hyperspectral data. The results we have obtained show it to be competitive with other recently developed classifiers for hyperspectral data when applied to the same data sets and in the case of the recently acquired low altitude AVIRIS data to yield good performance with very limited training data.

In this work the choice of kernel function is ad-hoc, as are the choices of the kernel function parameter d, and the separability parameter, C. However, the Support Vector Machine can be placed into the Structural Risk Minimization approach of Vapnik (Vapnik, 1995), and using rigorous bounds from recent results from probability theory, a more rigorous approach can be taken to choosing these parameters.

Also we note that all the results we have shown are completely in the spectral domain and no aspect of the spectral coherence of the data has been used. The results would be identical if all the classifier bands were permuted consistently throughout the data. And, we have not utilized the *spatial coherence* of the data. We note recent studies on the classification of hand written digits (Schölkopf *et al.*, 1998) show that performance gains can be made by incorporating prior knowledge into the construction of the Support Vector Machine and we believe similar gains can be made for classifying hyperspectral data using the coherence in the data.

6. ACKNOWLEDGEMENTS

The authors are grateful to T. Joachim (Joachims, 1998b) for making svm-light available (Joachims, 1998a), and to A. Smola for making PR_LOQO available (Smola, 1998).

References

Aizerman, M.A., Braverman, E.M., & Rozoner, L. I. 1964. Theoretical Foundations of the Potential Function Method in Pattern Recognition. *Automation and Remote Control*, **25**, 821–837.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. 1992. A Training Algorithm for Optimal Margin Classifiers. *Pages 144–152 of: Fifth Annual Workshop on Computational Learning Theory*. ACM.

Burges, C. J. C. 1998. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, **2**(2), 121–167. Also available at `http://svm.research.bell-labs.com/SVMdoc.html`.

Cortes, C. 1995. *Prediction of Generalization Ability in Learning Machines*. Ph.D. thesis, University of Rochester.

Cortes, C., & Vapnik, V. N. 1995. Support Vector Networks. *Machine Learning*, **20**, 1–25.

Gualtieri, J. A., & Cromp, R. F. 1998. Support Vector Machines for Hyperspectral Remote Sensing Classification. *Pages 221–232 of:* Merisko, R. J. (ed), *27'th AIPR Workshop, Advances in Computer Assisted Recognition, Proceeding of the SPIE, Vol. 3584*. SPIE.

Gunn, S. 1997 (November). *Support Vector Machines for Classification and Regression*. Tech. rept. Image Speech and Intelligent Systems Group, University of Southampton. The site `http://www.isis.ecs.soton.ac.uk/resources/svminfo/` contains the tutorial and matlab code which implements SVM's.

Hughes, G. F. 1968. On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, **14**(1).

Joachims, T. 1998a. The SVM^{light} package is written in gcc and distributed for free for scientific use from `ftp://ftp-ai.cs.uni-dortmund.ed/FORSCHUNG/ VERFAHREN/SVM_LIGHT/svm_light.eng.html` It can use one of several quadratic optimizers. For our application we used the A. Smola's PR_LOQO package (Smola, 1998).

Joachims, T. 1998b. Making large-Scale SVM Learning Practical. *In:* Schölkopf, B., Burges, C., & Smola, A. (eds), *Advances in Kernel Methods - Support Vector Learning.* MIT Press. Currently available at `http://www-ai.cs.uni-dortmund.de/DOKUMENTE/Joachims_98b.ps.gz`.

Johnson, L. F. 1998. *AVIRIS Hyperpsectral Radiance Data from: f981009t01r_07.* The full flight line is 961 samples by 3479 lines consisting of 224 bands of BIP calibrated radiance data that is scaled to lie in [0,10000] as 2bit signed integers. We have studied part of scene 3. The full flight line is called f981006t01p01_r07 as listed in the AVIRIS JPL repository (`ftp://makalu.jpl.nasa.gov/pub/98qlook/1998_lowalt_index.html`). The flight line is [Lat_Start: +36-15.0 Long_Start: -121-13.5 Lat_End: +36-21.0 Long_End: -121-13.5 Start_Time: 1945:30GMT End_Time: 1950:15GMT.

Landgrebe, D. 1992. *Indian Pines AVIRIS Hyperpsectral Reflectance Data: 92AV3C.* A 145×145 pixel and 220 band subset in BIL format of reflectance data that has been scaled to lie in [0,10000] as 2bit signed integers. It is available at `ftp://shay.ecn.purdue.edu/pub/biehl/MultiSpec/92AV3C`. Ground truth is also available at `ftp://shay.ecn.purdue.edu/pub/biehl/PC_MultiSpec/ThyFiles.zip`. The full data set is called Indian Pines 1 920612B as listed in the AVIRIS JPL repository (`http://makalu.jpl.nasa.gov/locator/index.html`). The flight line is Lat_Start: 40:36:39 Long_Start: -87:02:21 Lat_End: 40:10:17 Long_End: -87:02:21 Start_Time:19:42:43GMT End_Time:19:46:2GMT.

Landgrebe, D. 1998. *Multispectral Data Analysis: A Signal Theory Perspective.* Available at `http://dynamo.ecn.purdue.edu/~biehl/MultiSpec/Signal_Theory.pdf`.

Landgrebe, D. 1999. Information Extraction Principles and Methods for Multispectral and Hyperspectral Image Data. *Chap. 1 of:* Chen, H. (ed), *Information Processing for Remote Sensing.* World Scientific Publishing Co. Also available at `http://dynamo.ecn.purdue.edu/~landgreb/publications.html`.

Schölkopf, B., Simard, P., Smola, A., & Vapnik, V. 1998. Prior Knowledge in Support Vector Kernels. *Pages 640–646 of:* Jordan, M. I., Kearns, M.J., & Solla, S. A. (eds), *Advances in Neural Information Processing Systems,10.* MIT Press.

Smola, A. 1998. The PR_LOQO quadratic optimization package is distributed for research purposes by A. Smola at `http://svm.first.gmd.edu.de/software/loqosurvey.html`.

Smola, A. J., Schölkopf, B., & Müller, R. J. 1998. The Connection between Regularization Operators and Support Vector Kernels. *Neural Networks*, **to appear**.

Support-Vector-Machines.
1999. *Web sites for SVM.* `http://svm.first.gmd.de/`, `http://svm.research.bell-labs.com/`, `http://www.dcs.rhbnc.ac.uk/research/compint/areas/comp_learn/sv/index.shtml`, `http://wwwsyseng.anu.edu.au/lsg/`.

Tadjudin, S., & Landgrebe, D. 1998a (May). *Classification of High Dimensional Data with Limited Training Samples.* Ph.D. thesis, School of Electrical Engineering and Computer Science, Purdue University. available as TR-ECE-98-9 from `http://dynamo.ecn.purdue.edu/~landgreb/Saldju_TR.pdf`.

Tadjudin, S., & Landgrebe, D. 1998b. Covariance Estimation For Limited Training Samples. *In: Int. Geoscience and Remote Sensing Symposium.* Seattle, WA: IEEE. Available at `http://dynamo.ecn.purdue.edu/~landgreb/SaldjuCovarEst.pdf`.

Vapnik, V. N. 1982. *Estimation of dependencies based on empirical data.* Springer.

Vapnik, V. N. 1995. *The Nature of Statistical Learning Theory.* Springer.

Vapnik, V. N. 1998. *Statistical Learning Theory.* John-Wiley and Sons, Inc.

Vapnik, V. N., & Chervonenkis, A. Ja. 1974. *Theory of pattern Recognition.* Nauka. In Rusian.