Virtual Sensors: Using Data Mining Techniques to Efficiently Estimate Remote Sensing Spectra

Ashok N. Srivastava, *Member, IEEE,* Nikunj C. Oza, *Member, IEEE,* and Julienne Stroeve, *Member, IEEE*

***Abstract*— Various instruments are used to create images of the Earth and other objects in the universe in a diverse set of wavelength bands with the aim of understanding natural phenomena. Sometimes these instruments are built in a phased approach, with additional measurement capabilities added in later phases. In other cases, technology may mature to the point that the instrument offers new measurement capabilities that were not planned in the original design of the instrument. In still other cases, high resolution spectral measurements may be too costly to perform on a large sample and therefore lower resolution spectral instruments are used to take the majority of measurements. Many applied science questions that are relevant to the earth science remote sensing community require analysis of enormous amounts of data that were generated by instruments with disparate measurement capabilities. This paper addresses this problem using Virtual Sensors: a method that uses models trained on spectrally rich (high spectral resolution) data to ”fill in” unmeasured spectral channels in spectrally poor (low spectral resolution) data. The models we use in this paper are Multi-Layer Perceptrons (MLPs), Support Vector Machines (SVMs) with Radial Basis Function (RBF) kernels and SVMs with Mixture Density Mercer Kernels (MDMK). We demonstrate this method by using models trained on the high spectral resolution Terra MODIS instrument to estimate what the equivalent of the MODIS 1.6 micron channel would be for the NOAA AVHRR/2 instrument. The scientific motivation for the simulation of the 1.6 micron channel is to improve the ability of the AVHRR/2 sensor to detect clouds over snow and ice.**

***Index Terms*— Data Mining, Neural Networks, Support Vector Machine, Kernel Methods, Remote Sensing, MODIS, AVHRR, cloud detection.**

I. Introduction

THIS paper describes the development of data mining algorithms that learn to estimate unobserved spectra from remote sensing data. The idea is that data mining algorithms trained on spectrally-rich (high spectral resolution) data can be used to generate estimates of what those measurements would have been for data that are spectrally-poor (low spectral resolution), This enables us to glean more information from that spectrally-poor data. This is an important problem to solve because spectrally-poor data may be available for longer periods of time than spectrally-rich data. This happens because of improvements in measurement capabilities due to instruments being built in phases, technological improvements, or the need to reduce measurement costs. Many applied science questions that are relevant to the remote sensing community need to be addressed by analyzing very large amounts of data that were generated by instruments with different measurement capabilities.

For example, consider the relationship between the AVHRR/2 (Advanced Very High Resolution Radiometer) and the MODIS (Moderate Resolution Imaging Spectroradiometer) instruments. AVHRR/2 generates images in only five spectral channels, whereas MODIS generates images in 36 different spectral channels. However, AVHRR/2 data has been available since 1981 whereas MODIS has only been available since 1999. MODIS channels 1, 2, 20, 31, and 32 correspond reasonably well to the five AVHRR/2 channels. We can use data mining methods to model any MODIS channel not available in AVHRR/2 as a function of these five MODIS channels. We can then use the learned model to generate an estimate of what that MODIS channel would have been had it been available in AVHRR/2 given the five actual AVHRR/2 channels as input. If the learned model is of high quality, we can use it to obtain estimates of MODIS channels for years prior to 1999 when MODIS came on-line. We refer to this as a *Virtual Sensor* because it estimates unmeasured spectra. In this paper, we use Virtual Sensors to generate an estimate of MODIS channel 6 (1.6 microns) for AVHRR/2 because a spectral channel at 1.6 microns is useful for discriminating clouds from snow- and ice-covered surfaces. We chose this task to demonstrate the usefulness of Virtual Sensors in this paper.

In the next section, we discuss the scientific motivation for using Virtual Sensors to simulate MODIS channel 6 for the AVHRR/2 instrument. In Section III, we describe Virtual Sensors formally and as a general method going beyond the specific application that we discuss in Section II. In Section IV, we briefly review some standard machine learning algorithms that we use to perform the modeling necessary to create a Virtual Sensor. In Section V we discuss our experimental results. Section VI concludes the paper and discusses future work.

II. Virtual Sensors for Cryosphere Analysis

Intensification of global warming in recent decades has raised interest in year-to-year and decadal-scale climate variability in the Polar Regions. This is because these regions are believed to be among the most sensitive and vulnerable to climatic changes. The enhanced vulnerability of the Polar Regions is believed to result from several positive feedbacks, including the temperature-albedo-melt feedback and the cloud-radiation feedback. Recent observations of regional anomalies in ice extent, thinning of the margins of the Greenland ice

Manuscript received March 15, 2004; revised November 15, 2004. This work was supported by the NASA Intelligent Systems Intelligent Data Understanding Program.

A. N. Srivastava and N. C. Oza are at the NASA Ames Research Center. J. Stroeve is with the National Snow and Ice Data Center

sheet, and reduction in the northern hemispheric snow cover, may reflect the effect of these feedbacks. Remote sensing products provide spatially and temporally continuous and consistent information on several polar geophysical variables over nearly three decades. This period is long enough to permit evaluation of how several cryospheric variables change in phase with each other and with the atmosphere and can help to improve our understanding of the processes in the coupled land-ice-ocean-atmosphere climate system. Cloud detection over snow- and ice-covered surfaces is difficult using sensors such as AVHRR/2. This is because of the lack of spectral contrast between clouds and snow in the channels on the earlier AVHRR/2 sensors. Snow and clouds are both highly reflective in the visible wavelengths and often show little contrast in the thermal infrared.

The AVHRR Polar Pathfinder Product (APP) consists of twice daily gridded (at 1.25 and 5km spatial resolution) surface albedo and temperature from 1981 to 2000. A cloud mask accompanies this product but has been found to be inadequate, particularly over the ice sheets [1]. The 1.6 micron channel on the MODIS instrument as well as the AVHRR/3 sensor can significantly improve the ability to detect clouds over snow and ice. Therefore, by developing a virtual sensor to model the MODIS 1.6 micron channel (channel 6) as a function of the AVHRR/2 channels, we can improve the cloud mask in the APP product, and subsequently improve the retrievals of surface temperature and albedo in the product. In doing so we will be able to improve the accuracy in documenting seasonal and inter-annual variations in snow, ice sheet and sea ice conditions since 1981.

III. Virtual Sensors in general

In this section, we discuss Virtual Sensors in general, going beyond the specific application discussed in section II. For purposes of the discussion presented here, we model the data as matrices of time series (following the notation in [2]). The spatiotemporal random function $Z(\mathbf{u}, \lambda, t)$ is modeled as a finite number n of spatially correlated time series with the following representation:

$$\begin{aligned} Z(\mathbf{u}, \lambda, t) &= [Z_{\mathbf{u}}(\lambda, t)] \\ &= [Z_{u_1}(\lambda, t), Z_{u_2}(\lambda, t), ..., Z_{u_n}(\lambda, t)]^T \end{aligned} \quad (1)$$

In Equation 1, $\mathbf{u}$ represents the spatial coordinate, λ represents the vector of measured wavelength(s), and t represents time. The superscript ^{T} indicates the transpose operator. If multiple wavelengths are measured, then each Z_i is actually a matrix, and the function $Z(\mathbf{u}, \lambda, t)$ represents a data cube of size $(n \times \Lambda \times T)$, where these symbols represent the number of spatial locations, the total number of measured wavelengths, and the total number of time samples, respectively. In this notation, the spatial coordinate $\mathbf{u}$ represents the coordinates (or index) of a measurement at a particular location in the field of view. Conceptually, the equation above describes a set of n $(\Lambda \times T)$ matrices. In the event that the spatial coordinate indexes image pixels, it is useful to think of Equation 1 as describing a time series of data cubes (spectral images) of size $n \times n \times \Lambda$.

Consider a situation where one is given a sensor $\mathcal{S}_1$ which takes k spectral measurements in wavelength bands $\mathbf{B}_1 = \{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ at time t_1. Suppose that we have another sensor $\mathcal{S}_2$ which has a set of spectral measurements taken at time t_2, $\mathbf{B}_2 = \{\lambda_1, \lambda_2, \ldots, \lambda_k, \lambda_{k+1}, \lambda_{k+2}, \ldots, \lambda_{k+l}\}$ that partially overlaps the spectral features contained in $\mathbf{B}_1$ in terms of power in the spectral bands. Thus, $\mathbf{B}_1$ (or, in general, $\mathbf{B}_1 \cap \mathbf{B}_2$) are the common spectral measurements. Note that these measurements are common only in their power. $\mathbf{B} = \mathbf{B}_2 \setminus \mathbf{B}_1 = \{\lambda_{k+1}, \lambda_{k+2}, \ldots, \lambda_{k+l}\}$ represents the measurements available in $\mathbf{B}_2$ that are not available in $\mathbf{B}_1$. We investigate the problem of building an estimator $\Gamma(Z(\mathbf{B}))$ that best approximates the joint distribution $P(Z(\mathbf{B})|Z(\mathbf{B}_1))$, where $Z(\mathbf{B})$ is the data cube for the wavelength bands $\mathbf{B}$. Thus, we have:

$$\Gamma(Z(\mathbf{B})) \approx P(Z(\mathbf{B})|Z(\mathbf{B}_1)) \quad (2)$$

The value of building an estimator for P is clear particularly in situations where $\mathcal{S}_1$ has been in operation for a much longer period of time than $\mathcal{S}_2$. $\mathcal{S}_1$ may have fewer spectral channels in which measurements are taken compared to $\mathcal{S}_2$. However, it may be of scientific value to be able to estimate what the spectral measurements in wavelengths $\mathbf{B}$ would have been if $\mathcal{S}_1$ could have measured them.

The joint distribution given by $P(Z(\mathbf{B})|Z(\mathbf{B}_1))$ contains all the information needed to recover the underlying structure captured by the sensor $\mathcal{S}_2$. If perfect reconstruction of this joint distribution were possible, we would no longer need sensor $\mathcal{S}_2$ because all the relevant information could be generated from the smaller subset of spectral measurements $\mathbf{B}_1$ and the estimator Γ. Of course, such estimation is often extremely difficult because there may not be sufficient information in the bands $\mathbf{B}_1$ to perfectly reconstruct the distribution. Also, in many cases, the joint distribution cannot be modeled properly using parametric representations of the probability distribution since that may require a significant amount of domain knowledge and may be a function of the ground cover, climate, sun position, time of year, and numerous other factors.

In this paper, we describe methods to estimate the first moments of this distribution. Some methods allow us to model the second moment of the distribution as well:

$$\mu(Z(\mathbf{B})) = \int \Gamma(Z(\mathbf{B}))Z(\mathbf{B})d\mathbf{B} \quad (3)$$

$$\sigma^2(Z(\mathbf{B})) = \int [\Gamma(Z(\mathbf{B})) - \mu(Z(\mathbf{B}))]^2 Z(\mathbf{B})d\mathbf{B} \quad (4)$$

We use the function Γ in the above computations as an estimate of the (unknown) joint distribution P. Several computational problems as well as problems due to the underlying physical measurement process arise when we attempt to estimate Γ.

Figure 1 gives a schematic view of the general virtual sensor problem. The solid and dotted lines correspond to sensors $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively. A Virtual Sensor can be built when there are some overlapping sensor measurements as depicted in the figure. Notice that if there are no overlapping sensor

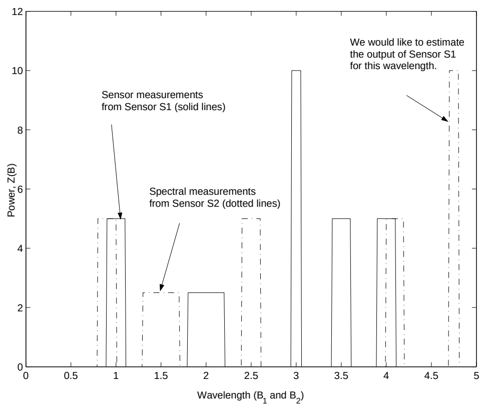

Fig. 1. This figure helps illustrate the need for a Virtual Sensor. We have spectral measurements from two sensors $\mathcal{S}_1$ and $\mathcal{S}_2$, (solid and dotted lines, respectively). We wish to estimate the output of sensor $\mathcal{S}_1$ for a wavelength where there is no actual measurement from the sensor. Note that some sensor measurements overlap perfectly, as in the case of wavelength $= 3$, and in other cases, such as wavelength = 1, there is some overlap in the measurements.

measurements, we are unable to build an estimator. In real-world problems, some measurements may overlap perfectly, while others have a partial overlap. Generally speaking the measurements from sensor $\mathcal{S}_1$ are not available at all wavelengths.

In the event that all k wavelength bands in $\mathcal{S}_1$ overlap with a corresponding subset of k bands in $\mathcal{S}_2$, but $\mathcal{S}_2$ has bands not available in $\mathcal{S}_1$, the estimation process is more straightforward. When partial overlap occurs between two sensors for a given wavelength, calculations need to be performed to estimate the amount of power that would have been measured in the overlapping bands. This can be done using interpolation methods.

We now outline the procedure for creating a Virtual Sensor. At a minimum, we assume that for sensor $\mathcal{S}_1$ we have measurements $Z_1(\mathbf{B}_1)$ from one image, and for another sensor $\mathcal{S}_2$ we assume that we have another image $Z_2(\mathbf{B}_2)$. The procedure for creating a Virtual Sensor is as follows, assuming that we need to build a predictor for channel $\mathbf{b}_{k+1}$ (recall that k is the number of bands in $\mathbf{B}_1$):

1) Find parameters θ that minimize the squared error (or another suitable metric) $\left[E[\Gamma(Z_2(\mathbf{B}_1),\theta)] - Z_2(\mathbf{b}_{k+1})\right]^2$. This is the Virtual Sensor model fitting step.

2) Apply Γ to the data from sensor $\mathcal{S}_1$ to generate an estimate of $E[\Gamma(Z_1(\mathbf{b}_{k+1}),\theta)]$. This is the step where the estimation of the unknown spectral contribution occurs.

3) Evaluate the results based on science based metrics and other information known about the image.

The procedure described above is standard in the data mining literature. From the remote sensing perspective, it is interesting to see the potentially systematic differences between the performances of the estimator on data from sensors $\mathcal{S}_1$ and $\mathcal{S}_2$. These will tell us how much the differences between the overlapping bands of the two sensors affect the accuracy of the Virtual Sensor relative to the true sensor.

TABLE I
LINEAR CORRELATIONS BETWEEN MODIS CHANNELS

Channel	1	2	20	31	32	6
1	1.0000	0.9980	0.8778	0.8785	0.8784	0.6287
2	0.9980	1.0000	0.8786	0.8774	0.8773	0.6564
20	0.8778	0.8786	1.0000	0.9977	0.9977	0.7369
31	0.8785	0.8774	0.9977	1.0000	1.0000	0.6979
32	0.8784	0.8773	0.9977	1.0000	1.0000	0.6984
6	0.6287	0.6564	0.7369	0.6979	0.6984	1.0000

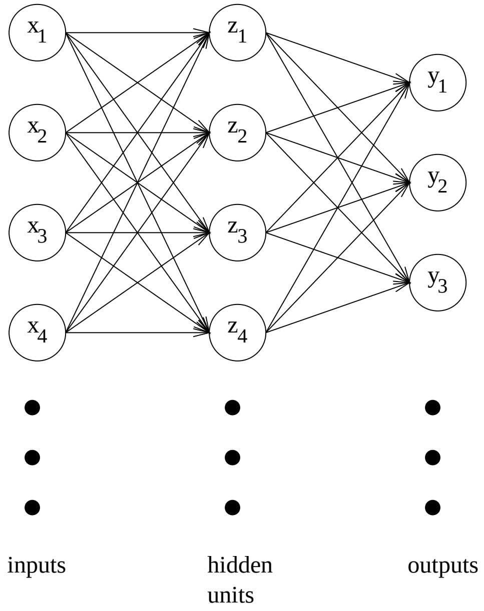

Fig. 2. An example of a MultiLayer Perceptron (MLP).

Note that this procedure will only work if sufficient information exists to predict $Z(\mathbf{B})$ given data $Z(\mathbf{B}_1)$. One simple procedure for determining this is to look at the linear correlation between the spectra. Figure I shows the inter-channel linear correlations for the MODIS channels that we use in this study (channels 1, 2, 20, 31, 32, and 6). In this paper we build models to predict MODIS channel 6. Notice that channel 6 has moderate linear correlations with the other channels. This gives us hope that we can predict MODIS channel 6 given the channels common to MODIS and AVHRR/2. However, the large correlations among the five common channels mean that they contain much redundant information; therefore, prediction may be difficult.

IV. STANDARD MACHINE LEARNING METHODS

This section describes three estimation methods that we have used to build a Virtual Sensor: a feed-forward neural network (also called a multilayer perceptron, (MLP)), a Support Vector Machine (SVM), and an SVM with a Mixture Density Mercer Kernel.

A. Multi-Layer Perceptrons

We first describe multilayer perceptrons, a type of neural network [3]. The central idea of neural networks is to construct linear combinations of the inputs as derived features, and then model the target as a nonlinear function of these derived features. Neural networks are often depicted as a directed graph consisting of nodes and arcs. An example is shown in Figure 2. Each column of nodes is a layer. The leftmost layer is the input layer. A data point to be classified is entered into the input layer. The second layer is the hidden layer and the third layer is the output layer. Information flows from the input layer to the hidden layer and then to the output layer via a set of arcs (depicted in Figure 2 as arrows). Note that the nodes within a layer are not directly connected. In our example, every node in one layer is connected to every node in the next layer, but this is not required in general. Also, a neural network can have more or less than one hidden layer and can have any number of nodes in each hidden layer.

Each non-input node, its incoming arcs, and its output (which is passed out through all of its outgoing arcs) constitute a neuron, which is the basic computational element of a neural network. Each incoming arc multiplies the value coming from its origin node by the weight assigned to that arc and sends the result to the destination node. The destination node adds the values presented to it by all the incoming arcs, transforms it with a nonlinear activation function (to be described later), and then sends the result along all of its outgoing arcs. For example, the return value of a hidden node z_j in our example neural network is

$$z_j = g\left(\sum_{i=1}^{|A|} w_{i,j}^{(1)} x_i\right), \tag{5}$$

where $|A|$ is the number of input units, $w_{i,j}^{(k)}$ is the weight on the arc in the kth layer of arcs that goes from unit i in the kth layer of nodes to unit j in the next layer (so $w_{i,j}^{(1)}$ is the weight on the arc that goes from input unit i to hidden unit j) and g is a nonlinear activation function. A commonly used activation function is the sigmoid function:

$$g(a) \equiv \frac{1}{1 + exp(-a)}. \tag{6}$$

The return value of an output node y_j is

$$y_j = g\left(\sum_{i=1}^{Z} w_{i,j}^{(2)} z_i\right) \tag{7}$$

where Z is the number of hidden units and $w_{i,j}^{(2)}$ is the weight on the arc from hidden unit i to output unit j. The outputs are clearly nonlinear functions of the inputs.

Neural networks are trained to fit data by a process that is essentially nonlinear regression. Given each entry in the training dataset, the network's current prediction is calculated. The difference between the true function value and the prediction is the error. The derivative of this error with respect to each weight in the network is calculated and the weights are adjusted accordingly to reduce the error.

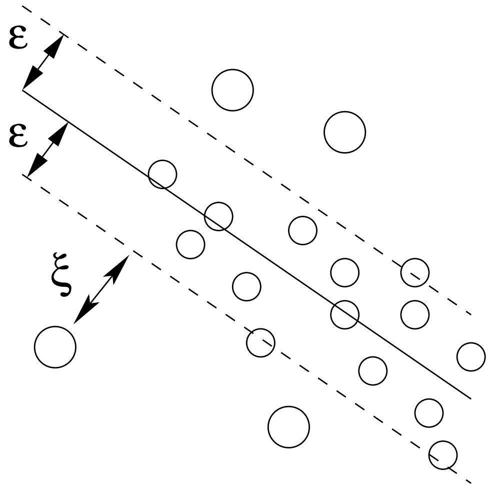

Fig. 3. Support Vector Machine for regression. The solid line is the line fitted to the points (represented as circles). The dashed lines are a distance ϵ from the fitted line. The points within the dashed line are considered to have zero error by an ϵ-insensitive loss function.

B. Support Vector Machines

Support Vector Machines for classification and regression are described in detail in [4], but here we briefly describe Support Vector Regression (SVR), which we use in this paper. In real-world problems, traditional linear regression cannot be expected to fit a data set perfectly (i.e., with zero error). For this reason, nonlinear regression is often used with the hope that a more powerful nonlinear model will achieve a better fit to the data than a linear model. However, this power often comes with two drawbacks. The first drawback is that the error surface as a function of the parameters of a nonlinear model (such as the multilayer perceptrons discussed above) often have many local optima that are not globally optimal. Nonlinear regression algorithms such as backpropagation for MLPs often find these local optima, which can result in a model that does not predict well on unseen data. The second drawback is that nonlinear model fitting is often overly sensitive to the locations of the training points, so that they overfit the training points and do not perform well on new data.

Support Vector Regression performs nonlinear regression by solving a convex optimization problem, which has one globally optimal solution. This solves the first drawback discussed above of ending up with a locally optimal parameter setting. SVR addresses the second drawback in three ways. The first way is to use an ϵ-insensitive loss function. If y is the true response and $f(\mathbf{x})$ is the predicted response for the input $\mathbf{x}$, then the loss function is

$$|y - f(\mathbf{x})|_\epsilon = max\{0, |y - f(\mathbf{x})| - \epsilon\} \tag{8}$$

That is, if the error between the true response and the predicted response is less than some small ϵ, then the error on that point is considered to be zero. For example, in Figure 3, the solid line, which is the fitted line, is within ϵ of all the points between the two dashed lines; therefore, the error is considered to be zero for those points. If ϵ is set to the level of the typical noise that one can expect in the response variable, then support vector regression is less likely to expend effort fitting the noise in the training data at the expense of generalization performance, i.e., it is less likely to overfit.

The second way support vector regression addresses the overfitting problem is to allow some error beyond ϵ for each training point but minimize the total such error over all the points. In Figure 3, ξ is the additional error for one particular point. The sum of the errors of all the training points is minimized as part of solving the optimization problem. This also reduces the effort expended in fitting the noise in the training data.

The third way that SVR addresses the above problems is to map the data from the original data space into a much higher (possible infinite) dimensional *feature space* and perform linear regression in that space. The idea is that the linear model in the feature space may correspond to a complicated nonlinear model in the original data space. Clearly, one needs a practical way to deal with data that is mapped to such a high-dimensional space, which intuitively seems impossible. However, one is able to do this using the *kernel trick*. By introducing Lagrange multipliers and obtaining the dual of the original SVR optimization problem (see [4] for the details), one obtains the following:

$$\text{maximize}_{\alpha,\alpha^* \in \mathcal{R}} -\epsilon \sum_{i=1}^{m} (\alpha_i^* + \alpha_i) + \sum_{i=1}^{m} (\alpha_i^* - \alpha_i) y_i \tag{9}$$

$$-\frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \mathbf{x}_i \cdot \mathbf{x}_j \tag{10}$$

$$\text{subject to } 0 \le \alpha_i, \alpha_i^* \le C \text{ for all } i \in \{1, 2, \ldots, m\} \tag{11}$$

$$\text{and } \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) = 0. \tag{12}$$

The resulting regression estimate is of the form

$$f(\mathbf{x}) = \sum_{i=1}^{m} (\alpha_i^* - \alpha_i) \mathbf{x}_i \cdot \mathbf{x}_j + b. \tag{13}$$

Note that the inputs ($\mathbf{x}$'s) only appear in dot products in the above solution. Therefore, one can map the inputs into a very high or even infinite dimensional space H using a function $\Phi : \mathcal{R}^d \rightarrow H$ and the dot product $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ will still be a scalar. Of course, Φ would be too difficult to work with because of the high dimensionality of H. However, there exist *kernel functions* $K(x_i, x_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ such that K is practical to work with even though the Φ induced by that K is not. For example, the Gaussian kernel (also referred to as the RBF kernel),

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|}{2\sigma^2}} \tag{14}$$

gives rise to a Φ that is infinite-dimensional. However, we do not need to work directly with Φ or even know what it is because the Φ's only appear within dot products, which can be replaced by K. Therefore, the new regression estimate after mapping the inputs from the data space to the feature space is

$$f(\mathbf{x}) = \sum_{i=1}^{m} (\alpha_i^* - \alpha_i) K(\mathbf{x}_i, \mathbf{x}_j) + b. \tag{15}$$

In summary, the Support Vector Machine allows us to fit a nonlinear model to data without the local optima problem that other procedures suffer from and with less tendency to overfit.

The kernel function K can be viewed as a measure of similarity between two data points. For example, with the Gaussian kernel, the value $K(\mathbf{x}_i, \mathbf{x}_j)$ increases as the distance between the pair of points $\mathbf{x}_i$ and $\mathbf{x}_j$ decreases. There is significant current research attempting to determine which kernel functions are most appropriate for different types of problems. One such novel kernel function is the Mixture Density Mercer Kernel (MDMK) which is discussed in the next section.

C. Mixture Density Mercer Kernels

The Mixture Density Mercer Kernel (MDMK) [5] is a method of learning a kernel function directly from the data. Some kernel functions, like the Gaussian Kernel discussed in the preceding section, are predefined. In fact, the Gaussian Kernel is just a nonlinear function of the Euclidean distance between points. Rather than assuming a priori that the Euclidean distance or some other distance function is correct, the MDMK generates a measure of similarity that attempts to represent the similarity between points based on their higher level features. These higher level features could be measured in a variety of ways. In the subsequent paragraphs, we illustrate one way of measuring higher level features.

Our idea is to use a collection or, more formally, an *ensemble* of probabilistic mixture models as a similarity measure. Two data points will have a large similarity if multiple models agree that they should be placed in the same cluster or mode of the distribution. Those points where there is some disagreement will be assigned intermediate similarity scores and points for which most models disagree will be assigned low similarity scores. The shapes of the underlying mixture distributions can significantly affect the similarity measurement of the two points. Experimental results uphold this intuition and show that in regions where there is "no question" about the membership of two points, the Mixture Density Kernel behaves identically to a standard mixture model. However, in regions of the input space where there is disagreement about the membership of two points, the behavior may be quite different from the standard model, i.e., the similarity measures returned may be very different. Since each mixture density model in the ensemble can be encoded with domain knowledge by constructing informative priors, the MDMK will also encode domain knowledge. The MDMK is defined as follows:

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= \Phi^T(\mathbf{x}_i)\Phi(\mathbf{x}_j) \qquad (16) \\ &= \frac{1}{Z(\mathbf{x}_i, \mathbf{x}_j)} \sum_{m=1}^{M} \sum_{c_m=1}^{C_m} P_m(c_m|\mathbf{x}_i) P_m(c_m|\mathbf{x}_j) \end{aligned}$$

The feature space is thus defined explicitly as follows:

$$\begin{aligned}\Phi(\mathbf{x}_i) \quad \propto \quad & [P_1(c=1|\mathbf{x}_i), P_1(c=2|\mathbf{x}_i), \ldots, \\ & P_1(c=C|\mathbf{x}_i), P_2(c=1|\mathbf{x}_i), \ldots, P_M(c=C|\mathbf{x}_i)]\end{aligned}$$

The first sum in equation 16 sweeps through the M models in the ensemble, where each mixture model is a Maximum A Posteriori estimator of the underlying density trained by sampling (with replacement) the original data. C_m defines the number of mixtures in the mth model, and c_m is the cluster (or mode) label assigned by the model. The quantity $Z(\mathbf{x}_i, \mathbf{x}_j)$ is a normalization such that $K(\mathbf{x}_i, \mathbf{x}_i) = 1$ for all i. The fact that the Mixture Density Kernel is a valid kernel function arises directly from the definition.

The Mixture Density Kernel function can be interpreted as follows. Suppose that we have a hard classification strategy, where each data point is assigned to the most likely posterior class distribution. In this case the kernel function counts the number of times the M mixtures agree that two points should be placed in the same cluster mode. In soft classification, two data points are given an intermediate level of similarity (between 0 and 1) which will be less than or equal to the case where all models agree on their membership, in which case the entry would be unity. Further interpretation of the kernel function is possible by applying Bayes rule to the defining equation of the Mixture Density Kernel. Thus, we have:

$$\begin{aligned}K(\mathbf{x}_i, \mathbf{x}_j) \quad &= \quad \frac{1}{Z(\mathbf{x}_i, \mathbf{x}_j)} \sum_{m=1}^{M} \sum_{c_m=1}^{C_m} \frac{P_m(\mathbf{x}_i|c_m)P_m(c_m)}{P_m(\mathbf{x}_i)} \times \\ & \qquad \frac{P_m(\mathbf{x}_j|c_m)P_m(c_m)}{P_m(\mathbf{x}_j)} \qquad (17) \\ &= \quad \frac{1}{Z(\mathbf{x}_i, \mathbf{x}_j)} \sum_{m=1}^{M} \sum_{c_m=1}^{C_m} \frac{P_m(\mathbf{x}_i, \mathbf{x}_j|c_m)P_m^2(c_m)}{P_m(\mathbf{x}_i, \mathbf{x}_j)}\end{aligned}$$

The second step above is valid under the assumption that the two data points are independent and identically distributed. This equation shows that the Mixture Density Kernel measures the ratio of the probability that two points arise from the same mode to the unconditional joint probability. If we simplify this equation further by assuming that the class distributions are uniform, the kernel tells us on average (across models) the amount of information gained by knowing that two points are drawn from the same mode in a mixture density.

V. Results

All the MODIS and AVHRR/2 data used in the analysis were geolocated and gridded to a 1.25 km Equal Area Scalable Earth Grid (EASE-grid) [6] containing the Greenland ice sheet and the surrounding ocean (which is mixture of open water and sea ice). Thirteen MODIS images from the year 2000 were processed (one for each day, 140-149 and 151-153). Corresponding AVHRR/2 images were available for the same dates, but at different orbital cross-over times. The results discussed in this section are obtained by training the three different methods on a small subset of a MODIS image from the Greenland ice sheet on day 140 of the year 2000. A small subset was chosen to train the models because of the high running time of the SVM models. The models were trained on day 140 and tested on MODIS and AVHRR/2 images from days 140-153. This approach maximizes the range of differences in time of year between the training and test images and allows for analysis of how much prediction loss occurs as a result of this difference. In running the models, only pixels for which the MODIS channel 1 (0.65 microns) top-of-atmosphere (TOA) reflectance[1] was greater than 0.3 were used, thereby removing pixels that are over open water and keeping only the snow/ice-covered areas. This turned out to be about half of the MODIS day 140 image (1.6 million pixels). Out of these pixels, we chose about 2500 of them at random for training. In all cases, the inputs were the five MODIS

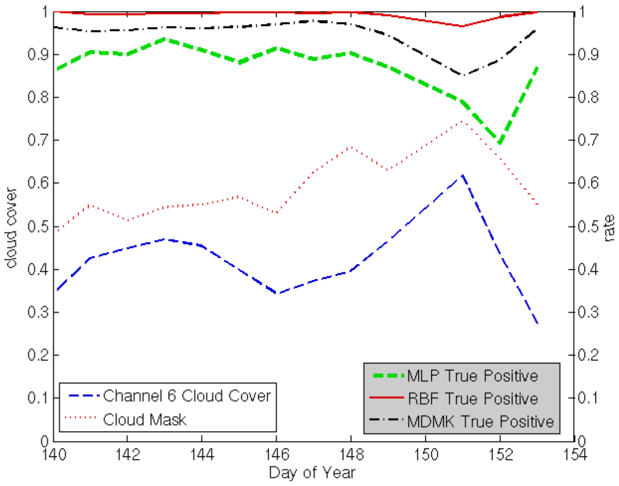

Fig. 4. MODIS predictions from year 2000, days 140-153. This figure shows the percent cloud cover in each image determined using channel 6 and using the cloud mask, and the true positive rates for MLPs, SVMs with RBF kernels, and SVMs with MDMK kernels on these images.

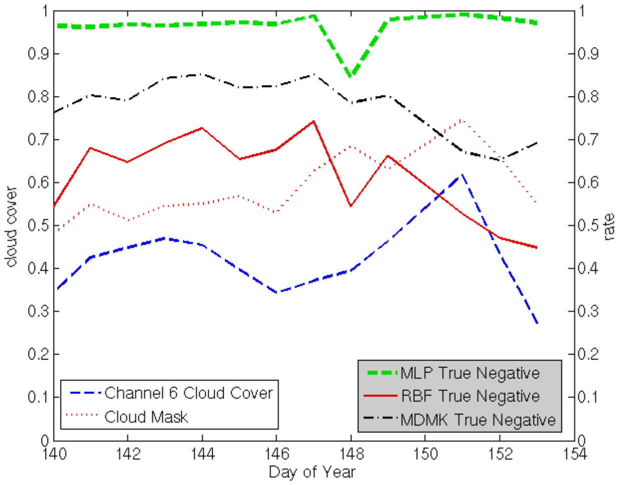

Fig. 5. MODIS predictions from year 2000, days 140-153. This figure shows the percent cloud cover in each image determined using channel 6 and using the cloud mask, and the true negative rates for MLPs, SVMs with RBF kernels, and SVMs with MDMK kernels on these images.

[1] This is the reflectance received by the sensor from the Earth's atmosphere. This is normalized by the cosine of the solar zenith angle.

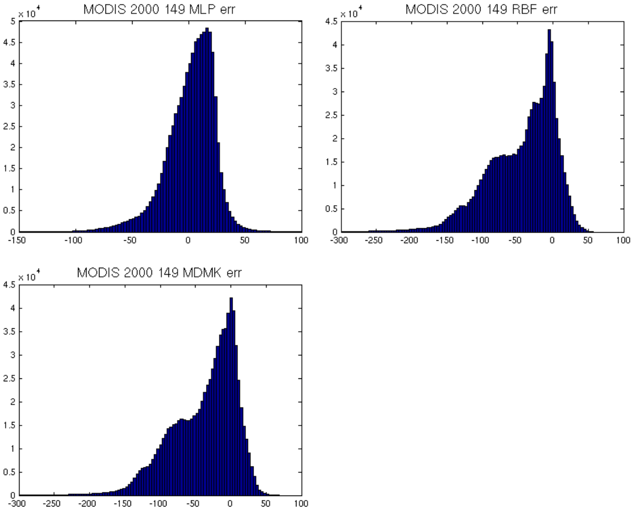

Fig. 6. Histogram of percentage error of MLP (upper left), SVM with RBF kernel (upper right), and SVM with MDMK kernel (lower left) relative to the true channel 6. This was calculated for MODIS year 2000 day 149 time 1825 for Greenland only.

channels that correspond most closely to the five AVHRR/2 channels (see the Appendix for tables with AVHRR/2 and MODIS instrument specifications for the channels used in this paper). That is, the inputs were the MODIS channels 1, 2, 20, 31 and 32. The output to be predicted was MODIS channel 6.

A. MODIS Results

Figure 4 summarizes the amount of cloud cover for each day (defined using a threshold of 0.2 on the MODIS channel 6 images) together with the true positive retrieval rates by the MLP, SVM with RBF kernel, and SVM with MDMK kernel. The true positive retrieval rate is defined as the number pixels predicted to have cloud cover that actually have cloud cover divided by the total number of pixels that actually have cloud cover. The threshold of 0.2 was chosen for channel 6 because the MODIS cloud mask team uses this threshold. Included in the figure is the percentage of cloud cover from the MODIS cloud mask (MOD35) product. In computing the fraction of cloud cover from the MOD35 product we counted as cloudy only the pixels that were classified as “cloudy” (i.e., we did not count those pixels classified as “probably cloudy,” “probably clear,” or “clear.” Notice that the MODIS cloud mask product predicts about 20% more clouds than using a threshold of 0.2 on MODIS channel 6. There are several possible reasons for this. Firstly, the MODIS cloud product uses other threshold tests besides the test on channel 6 reflectance. Secondly, studies have suggested that the MODIS cloud mask tends to overpredict the amount of clouds over snow [7]. Figure 5 shows the true negative retrieval rates of the three models together with the amount of cloud cover for each day. The true negative retrieval rate is the number of pixels predicted to not have cloud cover that actually do not have cloud cover divided by the total number of pixels that actually do not have cloud cover. Overall, we see that the SVM with RBF kernel has the greatest tendency to predict that a cloud is present, followed by the SVM with MDMK kernel and the MLP. Overall, the MLP seems to have the best combination of high true positive and true negative retrieval rates. However, as we will see later, the SVM-based methods, especially the MDMK kernel, discover certain structure in the data not discovered by the MLP.

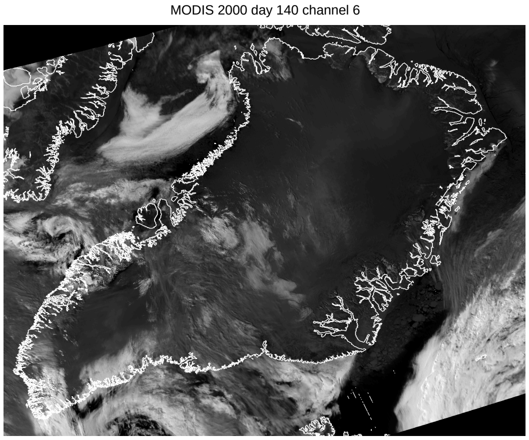

Fig. 8. MODIS year 2000 day 140 time 1830 true channel 6.

Figure 7 shows the MODIS true channel 6 (upper left) and the channel 6 predictions returned by the MLP (upper right), SVM with RBF kernel (lower left), and SVM with MDMK kernel (lower right). In all four images, the Greenland coastline is depicted in white, but only the upper half of the ice sheet is shown. The histogram of the percentage differences between the true channel 6 reflectance and the model-predicted channel 6 reflectances are shown in Figure 6[2]. The MLP appears to accurately model areas that are of low reflectance in the MODIS channel 6 (e.g. no clouds) as seen by the high rate of true negative retrieval. The MLP model is slightly less successful in correctly modeling the high reflectance (e.g. clouds), but the overall true positive retrieval rate is still relatively high (70 to 90%). The SVMs with RBF and MDMK model tends to overpredict the reflectance in MODIS channel 6, particularly in areas that are of low reflectance (e.g., no clouds).

B. AVHRR Results

We now discuss the results of testing our MODIS-trained models on two AVHRR/2 images. We evaluate these results by examining the available AVHRR/2 images, deciding where clouds are present based on textural variations, and observing whether the models' predictions capture these predictions. This subjective evaluation is necessary because the APP cloud mask is inadequate and the true 1.6 micron channel is unavailable.

Just as in the MODIS results, in the AVHRR/2 results the Greenland coastline is depicted. Figure 9(a) shows the visible (channel 1) TOA reflectance from AVHRR/2 for day 140 over the Greenland ice sheet. The image shows not only the

[2] These are calculated as the true channel 6 minus the predicted channel 6 divided by the true channel 6 multiplied by 100. Therefore, numbers less than 0 indicate that the model overpredicted while numbers greater than 0 indicate that the model underpredicted.

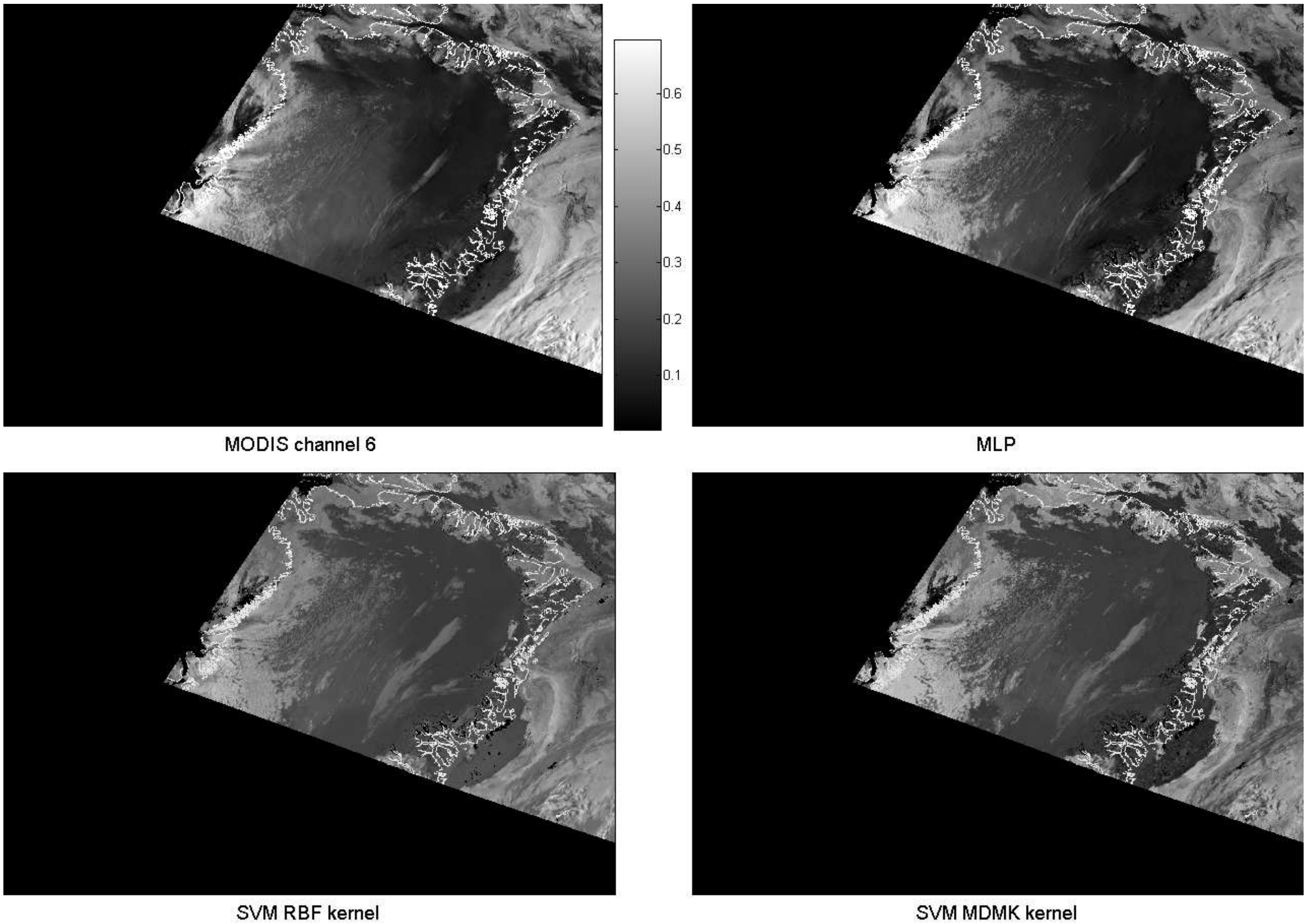

Fig. 7. MODIS predictions from year 2000, day 149 time 1825. (a) Upper Left. Channel 6. (b). Upper Right. Prediction of an MLP. (c) Lower Left. Prediction of an SVM with RBF kernel. (d). Lower Right. Prediction of an SVM with MDMK kernel. The black areas with straight boundaries are regions containing no data.

Greenland ice sheet with its coastline outlined in white, but also open water areas and sea ice. The same is true in the MODIS images shown in Figure 7, except in Figure 9 the entire Greenland ice sheet is shown. Clouds are evident in the visible image by textural variations in the south, central and northwestern part of the ice sheet. Some clouds also appear brighter and some darker than the underlying snow. In Figure 9(b) through (d) the predicted TOA reflectances for a channel at 1.6 microns are shown. We also show the MODIS channel 6 TOA reflectance for this day in Figure 8. However, note that this image is collected at a slightly different orbital time from the AVHRR image. Thus some differences are to be expected as a result of changes in cloud conditions with time. Even so, the MODIS channel 6 image is useful for helping to validate how well the models predict TOA reflectance at 1.6 microns. The MLP prediction (Figure 9(b)) indicates that the majority of the ice sheet is cloud free (very low reflectance at 1.6 microns), particularly for the northern half of the ice sheet. However, some of the clouds that are seen as textural variations in Figure 9(a) are captured in 9(b) as bright (higher reflectance) areas in the image, particularly in the central and southern regions of the ice sheet. Comparing Figure 9(b) with a qualitative assessment of the clouds in Figure 9(a), it appears that the majority of the clouds are captured, although the few scattered clouds in the northwest part of the ice sheet are not detected. The SVM RBF (Figure 9(c)) picks up the clouds (brighter areas in the image), but this method also starts to distinguish between different snow types as evident by the slightly different reflectance values along the western margin of the ice sheet. Further discrimination of different snow types is observed in the SVM MDMK (Figure 9(d)) image. For both the RBF and MDMK models, the tendency is to overpredict channel 6. Thus, additional information would be needed in order to distinguish between atmospheric variations (i.e. clouds) and variations in the snow/ice conditions. Note also that, off the northwestern coast of Greenland, sea ice areas that are cloud free appear as clouds (higher reflectance) in the predictions. Thus, additional information such as surface type may offer further improvements in the models' ability to detect clouds over snow and ice.

Figure 10(a)-(d) shows the same results as discussed above but for day 150. The visible image (Figure 10(a)) suggests that the entire western margin and the north central/eastern parts of the ice sheet are cloudy. The MODIS channel 6 image collected at a different time of day (Figure 11) indicates that most of the ice sheet is actually cloud free except for areas along the west-central part of the ice sheet and in the north. Comparing the MLP (Figure 10(b)) results with the clouds

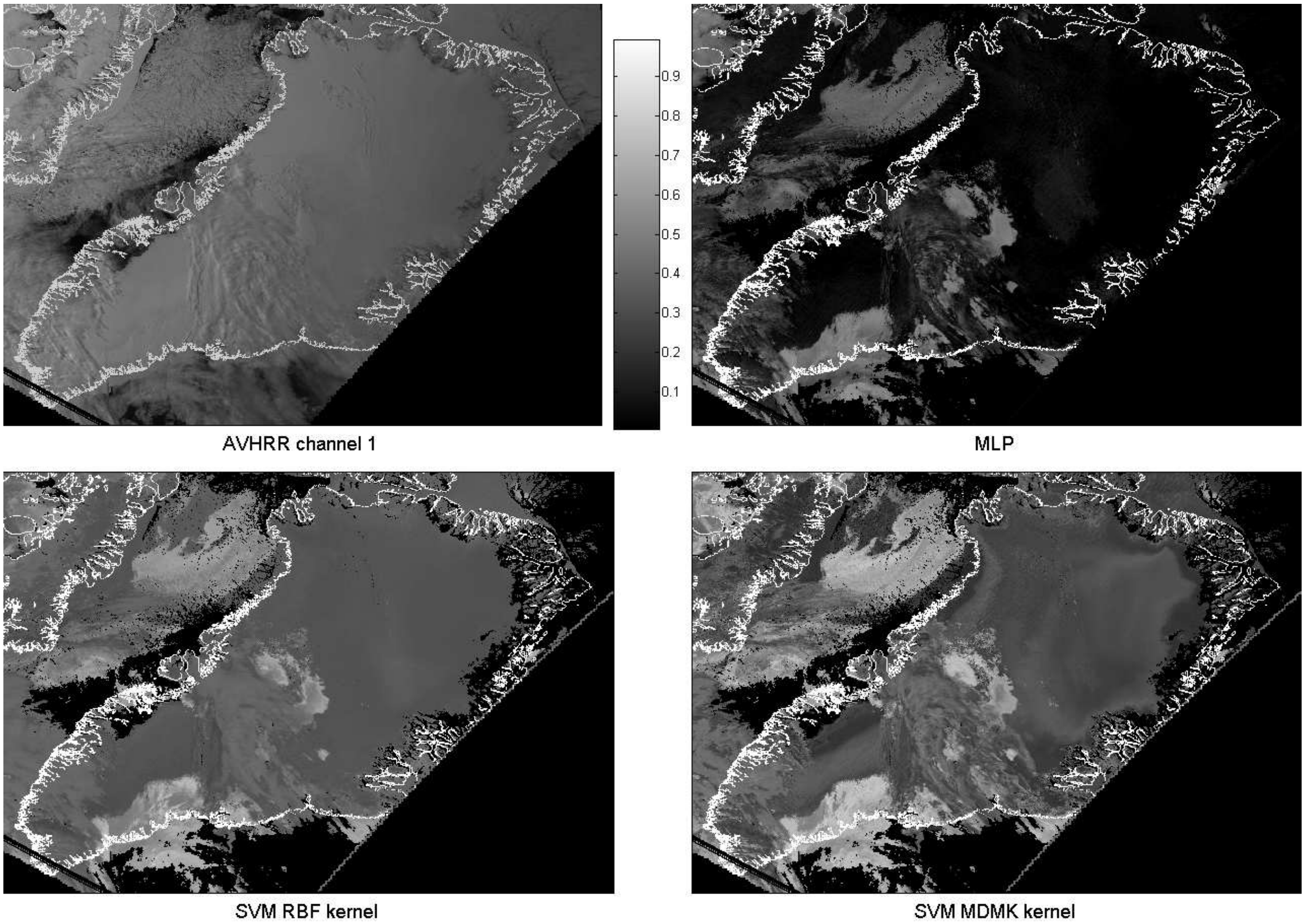

Fig. 9. AVHRR predictions from year 2000, day 140, time 1839. (a) Upper Left. Channel 1. (b). Upper Right. Prediction of an MLP. (c) Lower Left. Prediction of an SVM with RBF kernel. (d). Lower Right. Prediction of an SVM with MDMK kernel. The black areas with straight boundaries are regions containing no data.

indicated as textural variations in the AVHRR channel 1 image (Figure 10(a)) shows that this model captures some of the scattered clouds along the western margin of the ice sheet, but also misses quite a few of them, especially in the southern part and also the central-northern part of the ice sheet. Similarly, in the northeastern part of the ice sheet, the MLP is not capturing all the clouds observed in the visible image. The SVM RBF (Figure 10(c)) model does a better job of detecting the clouds in the northeastern region of Greenland as well as along the west coast. The SVM MDMK model further detects some clouds that are missed by the SVM RBF model (e.g. along the south-west edge of Greenland) and also begins to highlight more of the different snow/ice types.

These two different examples help to illustrate that simulating a 1.6 micron sensor channel does not necessarily capture all the clouds. In general, snow has very low reflectance at 1.6 microns, whereas clouds have high reflectance. Thus, we would expect snow cover to be bright in the visible channel and dark at 1.6 microns. However, cloud reflectance at 1.6 microns depends in part on the cloud type and may be bright or less bright (e.g. gray).

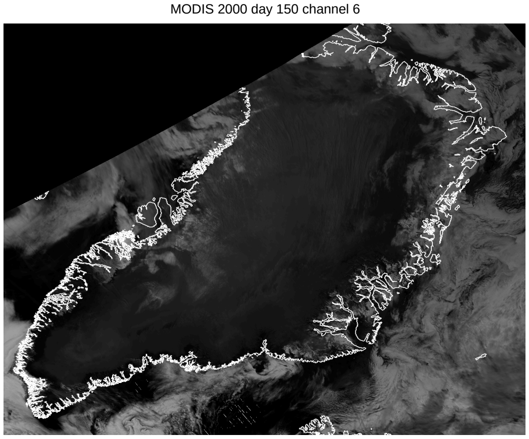

Fig. 11. MODIS year 2000 day 150 time 1905 true channel 6.

In the day 140 example, the MLP prediction does capture most all of the clouds observed in the visible image. For this day, the 1.6 micron is a good cloud classifier. On day 150 however, the MLP prediction does not perform quite as well. Even though it may still accurately predict the TOA reflectance at 1.6 microns, some clouds are missed.

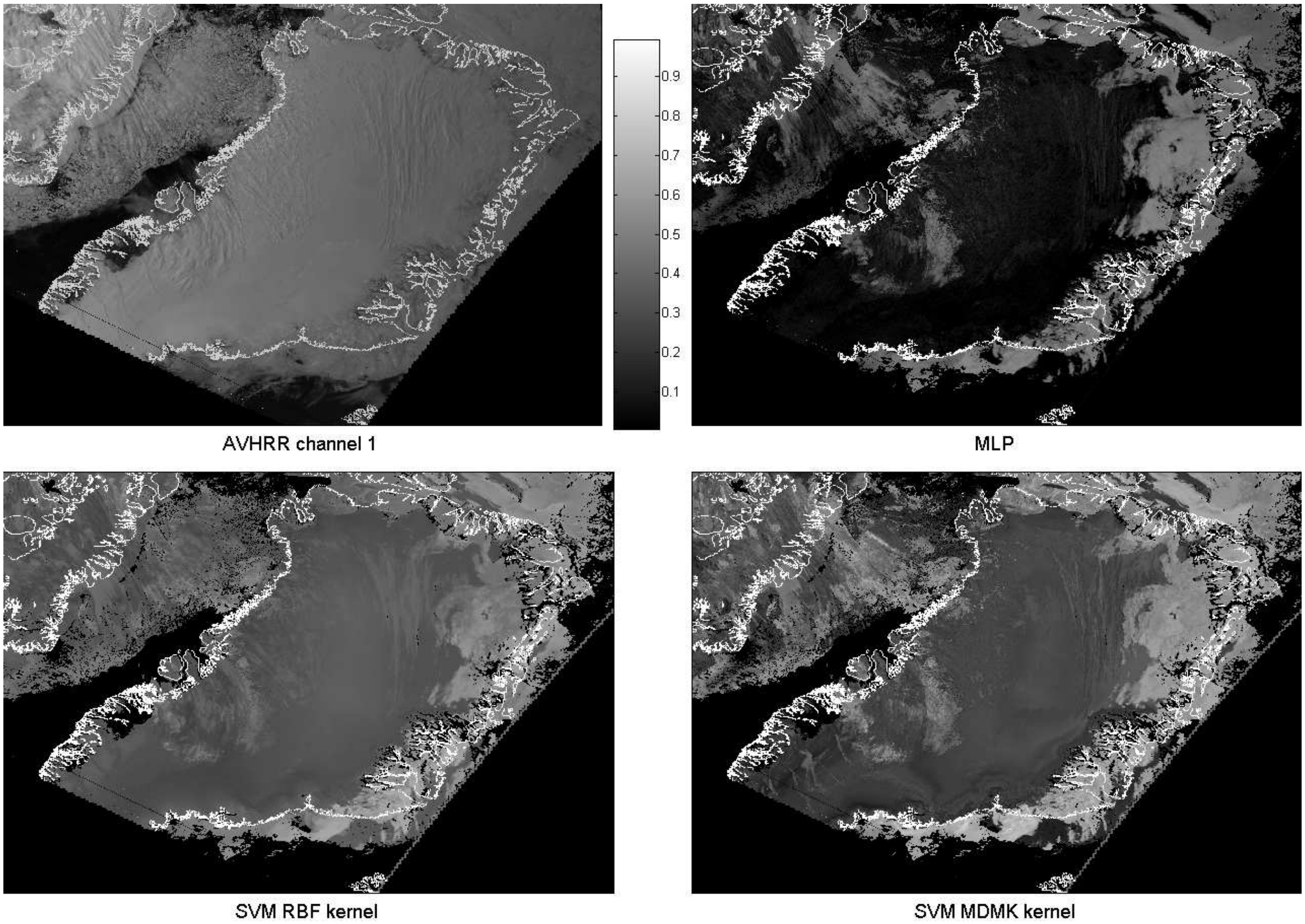

Fig. 10. AVHRR predictions from year 2000, day 150, time 1825. (a) Upper Left. Channel 1. (b). Upper Right. Prediction of an MLP. (c) Lower Left. Prediction of an SVM with RBF kernel. (d). Lower Right. Prediction of an SVM with MDMK kernel. The black areas with straight boundaries are regions containing no data.

VI. Conclusion

In this paper we have presented the development of data mining algorithms to estimate unobserved spectra. We call this estimation method “Virtual Sensors.” We presented some results on a particular instantiation of Virtual Sensors: the estimation of MODIS channel 6 for AVHRR/2. Our motivation for choosing this particular problem is to aid in the discrimination of clouds from snow and ice. This is a challenging problem that is essential to solve in order to map the cryosphere using visible and thermal imagery. Clouds often have spectral reflectances and temperatures similar to snow. Most cloud detection algorithms operationally employ a series of spectral threshold tests to determine if a pixel is clear or cloudy. Having a channel centered around 1.6 microns has significantly improved the ability to discriminate between clouds and snow using new sensors such as MODIS and AVHRR/3. Unfortunately, a vast amount of data have been collected before these sensors existed that did not have a channel designed to detect clouds over snow and ice-covered surfaces. These data sets have large importance for climate studies since they provide over 20 years worth of observations. Thus, being able to improve the cloud masking abilities of these previous sensors will allow for improved monitoring of several cryospheric variables, such as surface albedo, surface temperature, snow and ice cover.

In the above analysis, we used calibrated TOA reflectances from the MODIS and AVHRR/2 instruments. These reflectance values are dependent upon the specific viewing and illumination geometry of the orbit considered. This may lead to some errors since snow and clouds do not reflect the incoming solar radiation isotropically. The magnitude of this effect remains to be determined. However, the angular variability of the reflectance may possibly fall into the “noise” of the data so that our methods can be applied prior to using methods to correct for the angular variability of the TOA reflectance.

We plan to extend our work on the problem of estimating MODIS channel 6 for AVHRR/2 images in several directions. In order to determine if our methods have promise and can quickly learn a good model, we trained on very little data. We plan to train on additional data over different times of year to understand how much improvement is possible. We plan to develop more scalable algorithms that will allow us to train on large amounts of data in a practical amount of time. For example, active learning algorithms only process examples on which the current model’s predictions are significantly in error and do not waste effort on the remaining examples the way traditional machine learning algorithms do. Online learning algorithms process training examples only once rather than repeatedly cycling through them the way traditional algorithms do. We also plan to perform a more detailed analysis of the results over more images from different years and different times of year in order to better understand the situations in which different data mining algorithms are most effective. This may lead to the development of a hybrid scheme (e.g., ensemble) that performs better than any one method. Along these lines, our MDMK kernel enables us to build an ensemble of mixture models that use a variety of different kernel functions. Our algorithms currently only train on and generate predictions for individual pixels in individual images. Spatial correlation and temporal correlation will be accounted for in our future work.

We also plan to go beyond the particular problem of predicting channel 6 to predicting other channels and quantities that are of scientific importance. We will attempt to quantify cross-channel information through further mutual information studies.

APPENDIX I
INSTRUMENT SPECIFICATIONS

Tables II and III contain specifications of the AVHRR/2 and MODIS instruments, respectively.

TABLE II
AVHRR/2 INSTRUMENT SPECIFICATIONS

Channel Number	Wavelength (microns)	Purpose
1	0.58 to 0.68	Cloud Cover Snow Cover Vegetation Index
2	0.725 to 1.00	Earth Radiation Budget Surface Water Boundaries Vegetation Index
3	3.55 to 3.93	Water Vapor Correction Thermal Mapping
4	10.3 to 11.3	Thermal Mapping
5	11.5 to 12.5	Water Vapor Correction Thermal Mapping

TABLE III
MODIS INSTRUMENT SPECIFICATIONS

Band	Bandwidth (microns)	Primary Use
1	0.62 - 0.67	Land/Cloud/Aerosols Boundaries
2	0.841 - 0.876	Land/Cloud/Aerosols Boundaries
3	0.459 - 0.479	Land/Cloud/Aerosols Properties
4	0.545 - 0.565	Land/Cloud/Aerosols Properties
5	1.23 - 1.25	Land/Cloud/Aerosols Properties
6	1.628 - 1.652	Land/Cloud/Aerosols Properties
20	3.660 - 3.840	Surface/Cloud Temperature
31	10.780 - 11.280	Surface/Cloud Temperature
32	11.770 - 12.270	Surface/Cloud Temperature

ACKNOWLEDGMENT

The authors would like to thank Brett Zane-Ulman for preparation of many of the figures and results presented. This work was supported by the Intelligent Data Understanding segment of the NASA Intelligent Systems Program.

REFERENCES

[1] J. Stroeve, “Assessment of greenland albedo variability from the avhrr polar pathfinder data set,” *Journal of Geophysical Research*, vol. 33, pp. 989–1034, 2002.

[2] P. C. Kyriakidis and A. G. Journel, “Geostatistical space-time models: A review,” *Mathematical Geology*, vol. 31, no. 6, pp. 651–684, 1999.

[3] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford University Press, 1995.

[4] B. Schölkopf and A. Smola, *Learning with Kernels*. MIT Press, 2002.

[5] A. N. Srivastava, “Mixture density mercer kernels: A method to learn kernels directly from data,” *Proceedings of the 2004 SIAM Data Mining Conference*, 2004.

[6] R. Armstrong and M. Brodzik, “Earth-gridded ssm/i data set for cryospheric studies and global change monitoring,” in *AI Symposium of COSPAR Scientific Commission A*, Hamburg, Germany, 1995, pp. 115–163.

[7] G. Scharfen and S. Khalsa, “Assessing the utility of modis for monitoring snow and sea ice extent,” in *European Association of Remote Sensing Laboratories Proceedings*, vol. 2, 2003, pp. 122–127.

Ashok N. Srivastava Dr. Ashok N. Srivastava is a Principal Scientist and Group Leader in the Data Mining and Complex Adaptive Systems Group at NASA Ames Research Center. He has fourteen years of research, development, and consulting experience in machine learning, data mining, and data analysis in time series analysis, signal processing, and applied physics. Dr. Srivastava has had significant experience both in research (NASA, NIST, IBM) as well as the business world at IBM (Senior Consultant) and Blue Martini Software (Senior Director).

Dr. Srivastava’s machine learning research interests include topics in kernel methods, assessment of linear and nonlinear covariability, understanding and forecasting time-based data, and image processing. He is also interested in distributed data mining and scalability issues in federated data systems. A primary area of applied research is in the development of onboard satellite algorithms for automatic detecting and discovery of geophysical processes.

Nikunj C. Oza Dr. Nikunj C. Oza has been a Research Scientist at NASA Ames Research Center since September, 2001. He received his B.S. in Mathematics with Computer Science from the Massachusetts Institute of Technology (MIT) in 1994, and M.S. and Ph.D. in Computer Science from the University of California at Berkeley in 1998 and 2001, respectively. His research interests include ensemble learning, online learning, and applications of machine learning to such problems as fault detection and remote sensing.

Julienne Stroeve Dr. Julienne C Stroeve has been a research scientist at the National Snow and Ice Data Center (NSIDC) since 1996. She received her B.S. (1989) and M.S. (1991) in Aerospace Engineering from the University of Colorado. Her Ph.D. was received in 1996 from the Geography Department at the University of Colorado where her thesis dealt with deriving a radiation climatology of the Greenland ice sheet using satellite imagery. Her research interests include optical, thermal and microwave remote sensing of snow and ice-covered surfaces, cryosphere-climate interactions, atmospheric radiative transfer modeling, and image processing.