Distributed Anomaly Detection using 1-class SVM for Vertically Partitioned Data

Kamalika Das*, Kanishka Bhaduri†, Petr Votava‡

*Stinger Ghaffarian Technologies Inc.,

NASA Ames Research Center, MS 269-3, Moffett Field, CA-94035

Email:Kamalika.Das@nasa.gov

†Mission Critical Technologies Inc.,

NASA Ames Research Center, MS 269-1, Moffett Field, CA-94035

Email:Kanishka.Bhaduri-1@nasa.gov

‡CSU Monterey Bay, NASA Ames Research Center, CA-94035

Email:Petr.Votava-1@nasa.gov

A shorter version of this paper was published in NASA Conference on Intelligent Data Understanding 2010.

Abstract

There has been a tremendous increase in the volume of sensor data collected over the last decade for different monitoring or surveillance tasks. For example, terabytes of Earth Science data is collected from modern satellites, in-situ sensors and different climate models. Similarly, petabytes of flight performance data is downloaded from all commercial airlines at their respective operational hubs. Both these types datasets need to be analyzed for finding interesting patterns or for searching for extremes or outliers. Information extraction from such rich data sources using advanced data mining methodologies is a challenging task not only due to the massive volume of data, but also because these datasets are physically stored at different geographical locations with only a subset of features available at at any location. Moving these petabytes of data over the network to a single location may waste a lot of bandwidth, and can take days to finish. To solve this problem, in this paper, we present a novel algorithm which can identify outliers in the global data without moving all the data to one location where only a subset of the features is stored at any location. We prove both theoretically and experimentally that the algorithm offers high accuracy compared to a centralized algorithm with only a fraction of the cost required for centralization. The experiments are demonstrated on two large publicly available datasets: (1) the NASA MODerate-resolution Imaging Spectroradiometer (MODIS) satellite images and (2) a 'Commercial Modular Aero-Propulsion System Simulation' which generates simulated data of realistic large commercial turbofan engines.

I. Introduction

Anomaly detection or outlier detection [1] is widely used for detecting *abnormal* or *unusual* patterns from data. Depending on how anomalies are defined, different algorithms have been developed for finding anomalies from a dataset, each with different assumptions and complexities [2], [3], [4]. Outlier detection is well studied when all the data is at one location (centralized version); however, the problem is more challenging when the data is at stored at different locations such that each site only has a subset of features (attributes). The goal is to identify anomalies in such distributed datasets by minimizing the number of data elements transferred between the sites.

For example, in Earth science applications, data is collected and generated by growing number of satellites, in-situ sensors and increasingly complex ecosystem and climate models. This growth in volume and complexity is going to continue because in order for the scientists to better understand and predict the Earth system processes, they will require far more comprehensive data sets spanning many years and more complex models. With the launch of NASA's Terra and Aqua missions, and the expected launches of number of missions recommended by the Decadal Survey, the need for more efficient and scalable data processing system is crucial. The volume of data itself is often a limiting factor in obtaining the information needed by the scientists and decision makers. This data volume will grow from hundreds of terabytes to tens of petabytes throughout the lifespan of the proposed Decadal Survey missions. More data means more information, only if there are sophisticated means of sifting through the data for extracting the relevant information from this data avalanche.

A very interesting task relevant to the Earth science community is identification of anomalies within the ecosystems (*e.g.* wildfires, droughts, floods, insect/pest damage, wind damage, logging), so that experts can then focus their analysis efforts on the identified areas. There are dozens of variables that define the health of the ecosystem and both long-term and short-term changes in these variables can serve as early indicators of natural disasters and shifts in climate and ecosystem health. These changes can have profound socio-economic impacts and it is important to develop capabilities for identification, analysis and response to these changes in a timely manner. In order to fully understand the Earth systems, scientists need to be able to analyze together a number of datasets from satellites, ground sensors and models. Every data component has a different observation or predictive capability and therefore a global analysis on a combination of modalities gives better results than studying a particular feature. For example, observing different but related phenomena, predicting climate impacts at different timesteps, or providing observations of the same phenomena through different means, such as ground sensor or a radar are expected to enable better comprehension and more accurate characterization of changes and disturbances in Earth systems.

The situation is greatly complicated by the fact most of the data representing different modalities are stored at geographically distributed archives, such as NASA's Distributed Active Archive Centers (DAAC), each containing data specific to only a subset of the scientific community and thus it is almost impossible to perform a globally consistent analysis. Given this scenario, the current approach would be for the scientist to look at only a subset of the dataset available at one site (and thereby compromise on the quality of the results) or to bring all the data together in one place and then perform the analysis. While the second approach works for lower data volumes, it is not feasible to centralize all the data when it grows beyond what can be gathered using current network infrastructure in a timely manner. Another reason why complete centralization is not possible is because the research is done in number of different science teams and organizations in different countries. While there is a trend to consolidate more data at fewer data centers, the capabilities to extract vital information from large distributed datasets will continue to be a key for the Earth science community to be able to gather significant results by analyzing the growing data volumes being accumulated world wide.

In this paper we describe a novel and efficient algorithm for anomaly detection in distributed databases where each site has only a subset of attributes. The contributions of this work, based on the state of the art in distributed anomaly detection, can be enumerated as:

- To the best of the authors' knowledge, this is the first algorithm that can perform outlier detection when the data is distributed across several sites, with only a subset of features at each site. In the distributed data mining literature this is known as the vertically partitioned scenario.
- We theoretically derive the true positive rate and show that the false positive rate of the algorithm is 0.
- For the proposed algorithm, the amount of communication required is less than 1% of that required for centralization, yet is 99% accurate compared to a centralized algorithm in finding the outliers. The accuracy is a function of the data percentage communicated and can be tuned based on the performance requirements and resources available to the users.

The rest of the paper is organized as follows. In the next section (Section II) we present the work related tho this area of research. We discuss the notations and the one class SVM formulation in Section III. In Section IV we present details about the proposed algorithm. We discuss the theoretical analysis of the algorithm in Section V. Performance of the algorithm on NASA satellite data is presented in Section VI. Finally we conclude the paper in Section VII.

II. Related work

Outlier or anomaly detection refers to the task of identifying abnormal or inconsistent patterns from a dataset. While outliers may seem as undesirable entities in a dataset, identifying them have many potential applications such as in fraud and intrusion detection, financial market analysis, medical research and safety-critical vehicle health management. Broadly speaking, outliers can be detected using *supervised*, *semi-supervised* or *unsupervised* techniques [5][1]. Unsupervised techniques, as the name suggests, do not require labeled instances for detecting outliers. In this category, the most popular methods are distance-based and density based techniques. The basic idea of these techniques is that outliers are points in low density regions or those which are far from other points. In their seminal work, Knorr *et al.* [4] proposed a distance-based outlier detection technique based on the idea of nearest neighbors. The naive solution has a quadratic time complexity since every data point needs to be compared to every other to find the nearest neighbors. To overcome this, researchers have proposed several techniques such as the work by Angiulli and Pizzuti [6], Ramaswamy *et al.* [7], and Bay and Schwabacher [2]. Density-based outlier detection schemes, on the other hand, flag a point as an outlier if the point is in a low density region. Using the ratio of training and test data densities as an outlier score, Hido *et al.* [3] propose a new inlier-based outlier detection technique. Supervised techniques require labeled instances of both normal and abnormal operation data for first building a model (*e.g.* a classifier) and then testing if an unknown data point is a normal one or an outlier. The model can be probabilistic based on Bayesian inferencing [8] or deterministic such as decision trees, support vector machines and neural networks [9]. Semi-supervised techniques only require labeled instances of normal data. Therefore, they are more widely applicable than the fully supervised ones. These techniques build models of normal data and then flag as outliers all those points which do not fit the model.

There exists a plethora of work on outlier detection from spatio-temporal databases. Barua and Alhajj [10] present a technique for outlier detection from meteorological data using a parallel implementation of the well-known wavelet transformation. The authors show that by implementing the algorithm on modern high performance multi-core processors, they achieve both improved speedup and accuracy. Birant and Kut [11] discuss a way of identifying both spatial and temporal outliers in large databases. They argue that existing methods do not identify both these outliers, and hence they propose a new DBSCAN clustering method to first cluster the dataset based on the density of points and then tags as outliers all points which have low density in its neighborhood. Now depending upon the type of outlier detected, either spatial or temporal neighborhood is considered. Both these methods consider outliers as single points. In practice, there may be a group of points which are outliers *e.g.* a tornado or other natural disaster affecting a large area. Zhao *et al.* [12] present an outlier detection method based on wavelet transformation which can detect region outliers. In their approach, they first transform the image to the wavelet domain and then isolate those coefficients which are greater than a threshold. Inverse wavelet transformation on this thresholded pixels are then candidates for outliers which are further filtered by running an outlier detection method. Land cover change detection has been studied by Boriah *et al.* [13] and Potter *et al.* [14]. In [13], the authors have proposed a recursive merging algorithm for change point detection. In their approach, the data is stored as a matrix of N locations and 12 months. Two most similar consecutive annual cycles are merged, and the distance is stored. This is applied recursively until only one annual cycle is left remaining. The change score for any location is based on whether any of the observed distances are extreme. They show how the method detects new golf courses, shopping centers and other land cover changes. For more details on the recent work on change detection for land cover data, readers are referred to [13] and the references therein. Several other techniques also exist for building classification and prediction models for mining geospatial data such as [15].

Although there is this huge body of literature on anomaly detection techniques for Earth Science data, many domain experts still continue to use primitive statistical measures such as points outside $\mu \pm 3\sigma$ of a Gaussian distribution as measures for identifying potential outliers from the huge Earth Sciences datasets. One of the reasons for this is the fact that most of the outlier detection techniques fail to scale to the order of terabytes or petabytes which is the order of the Earth Science data sets currently. Also, none of these techniques can handle the data when it is vertically partitioned across a large number of sites. Although techniques exist for horizontally partitioned scenario (*e.g.* [16]), extending them to vertically partitioned scenario is not obvious. Our proposed algorithm can perform anomaly detection without centralizing all the data to one location and thus, can handle massive datasets.

III. Background

In this section we first define the notations and then discuss ν 1-class SVM (where ν is a user chosen parameter) which forms a building block for our distributed anomaly detection technique. Although we focus on SVM, our distributed algorithm can be used with many other base classifiers such as decision trees, neural networks, rule-based classifiers etc. However, it becomes extremely difficult to adopt a distance-based outlier detection algorithm (*e.g.* k-NN) to our framework mainly because none of the distances can be computed based on a single node's data.

A. Notations

Let $P_0, ..., P_p$ be a set of computation nodes where P_0 is designated as the master node and the others are denoted as the computational nodes. Let the dataset at node P_i ($\forall i > 0$) be denoted by $D_i = \begin{bmatrix} \overrightarrow{x_1^{(i)}} & \ldots & \overrightarrow{x_m^{(i)}} \end{bmatrix}^T$ consisting of m rows where $\overrightarrow{x_j^{(i)}} \in \mathbb{R}^{n_i}$. Here each row corresponds to an observation and each column corresponds to a feature/attribute/sensor measurement. It should be noted here that there should be a one-to-one mapping between the rows across the different nodes. That kind of correspondence, if not available for the raw measured data, can be established using standard cross matching techniques for data preprocessing that exist in the literature *e.g.* the Sloan Digital Sky Survey[1]. In the distributed data mining literature, this is referred to as the vertically partitioned data distribution scenario. The global set of features (n) is the vertical concatenation of all the features over all nodes and is defined as $n = [n_1 \quad n_2 \quad \ldots \quad n_p]$ (using Matlab notation). Hence, the global data D is the $m \times n$ matrix defined as the union of all data over all nodes *i.e.* $D = [\overrightarrow{x_1} \quad \ldots \quad \overrightarrow{x_m}]^T$ with $\overrightarrow{x_j} \in \mathbb{R}^n$. Note that, here we make the implicit assumption that the ℓ-th row of all the sites corresponds to the ℓ-th observation *i.e.* the observations have been cross-matched.

Let $\mathcal{O}_i$ denote the set of local outliers at node P_i, detected by an outlier detection algorithm running on D_i such that $|\mathcal{O}_i| < |D_i|$. We give a precise definition of outlier and an algorithm to detect those in the next section. The global set of outliers found by a centralized algorithm having access to all the data is denoted analogously by the set $\mathcal{O}_c$. The set of outliers found by the distributed algorithm is denoted by $\mathcal{O}_d$.

B. One class ν-SVM

Given a training dataset containing two classes of examples, one class SVMs, introduced by Schölkopf *et al.* [17], is a supervised learning method for drawing a separating hyperplane that separates these two classes. In our discussion, we will refer to positively labeled data points as normal and negatively label data points as outliers. Instead of using both types of examples from the training data for constructing the hyperplane, one class SVM uses only instances with positive labels to do the same. It also uses a parameter ν which denotes the maximum allowance of outliers in the training data. During the training phase, the SVM algorithm optimizes the

[1] `http://cas.sdss.org/astrodr6/en/tools/crossid/upload.asp`

placement of the hyperplane in order to maximize the margin between the hyperplane and the origin, which is the lone representative of the second class with negative label.

In many cases, the decision boundary is non-linear in the input space and the trick is to transform the input data to a higher dimension space; the latter allowing for linear separability. This mapping is often made implicit using a kernel function $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ (d is the dimension of the data) which actually computes the inner product between the input vectors in this (possibly) infinite dimensional space. Throughout this paper, we have used Radial Basis Function (RBF) kernel:

$$k(\overrightarrow{x_i}, \overrightarrow{x_j}) = \exp\left(\frac{-\parallel \overrightarrow{x_i} - \overrightarrow{x_j} \parallel^2}{2\sigma^2}\right) \tag{1}$$

where $\|\cdot\|$ denotes the Euclidean norm and σ defines the kernel width. σ is often needs to be tuned for a particular dataset.

Schölkopf [17] showed that in the high dimensional feature space it is possible to construct an optimal hyperplane by maximizing the margin between the origin and the hyperplane in the feature space by solving the following optimization problem,

$$\begin{aligned} \text{minimize} \quad & Q = \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j k(\overrightarrow{x_i}, \overrightarrow{x_j}) + \rho\left(\nu m - \sum_i \alpha_i\right) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq 1, \quad \nu \in [0,1] \end{aligned} \tag{2}$$

where α_i's are Lagrangian multipliers, ν is a user specified parameter that defines the upper bound on the fraction of the training error and also the lower bound on the fraction of support vectors, and ρ is the offset of the hyperplane from the origin. The optimal solution returns a set of points SV from the training set known as the *support vectors* for which the $0 \leq \alpha_i \leq 1$ and also the value of the bias term ρ. Now, for any test point $\overrightarrow{x_t}$, not in the training set, the optimal decision is based on the following inner product computation:

$$f(\overrightarrow{x_t}) = \sum_{i \in SV} \alpha_i k(\overrightarrow{x_i}, \overrightarrow{x_t}) - \rho \tag{3}$$

The point $\overrightarrow{x_t}$ is an outlier if $f(\overrightarrow{x_t}) < 0$.

C. Overview of algorithm

The distributed outlier detection algorithm that we have developed consists of two steps. In the first step, a local anomaly detection algorithm is executed at each node which identifies outliers based on the features present at these nodes only. Then, these local outliers from each of the nodes are collected at one central node (master node). Along with this, samples from the local nodes are also collected at the master node to build a global model. Then, all these local outliers are tested against the global model. Only those which are tagged as outliers by the global model are then output as the outliers from the distributed algorithm. We will show both theoretically and experimentally that our algorithm has a high true positive rate and zero false positive rate. Figure 1 shows the proposed distributed architecture. We elaborate on each of these steps in the next section.

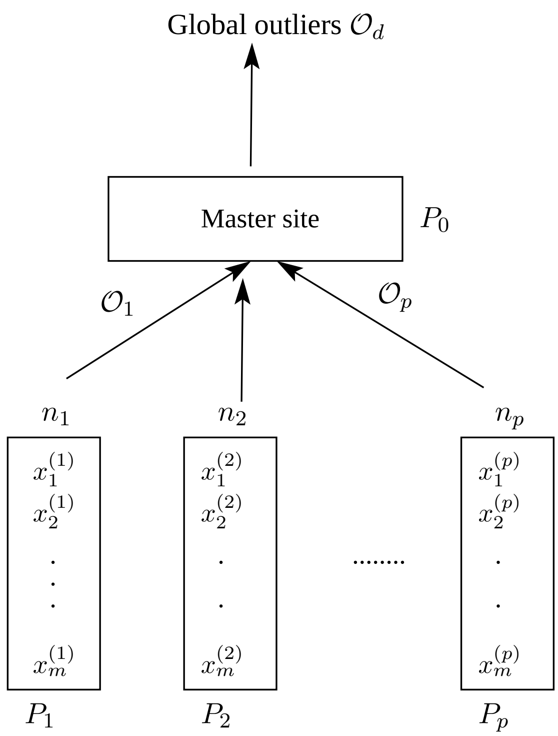


Fig. 1. This figure shows the proposed distributed architecture. P_0 is the master site and the other sites are the computation sites. Local outliers $\mathcal{O}_i$ are sent to P_0, which are then output the final outliers $\mathcal{O}_d$.

IV. Algorithm details

A. Pruning rule

As stated earlier, the goal of distributed outlier detection is two-fold: (1) compute the correct set of outliers (with respect to a centralized execution) and (2) minimize the cost of communicating the data to a central node for computation. Distributed algorithms often define rules based on the data to minimize communication while guaranteeing that the global task is accomplished [18][19][20]. These data dependent rules are such that, if satisfied by all nodes independently, then certain global properties of the dataset hold.

In this paper we use the following observation to prune the number of messages that need to be sent to the master site for determining the global set of outliers:

Pruning rule: *An observation $\vec{x} \in D$ may be a global outlier (with respect to all the features) i.e. $\vec{x} \in \mathcal{O}_d$, if it is an outlier with respect to at least one (or a subset) of the features i.e.* $\exists j \in \{1 \dots p\}, \quad \overrightarrow{x^{(j)}} \in \mathcal{O}_j$.

In other words, we assume that a point cannot be a global outlier if it is not an outlier for any of the local sets (axis). While this statement may not be true in general, it provides us with a way of pruning the number of observations that needs to be sent to the central site. We verify theoretically that the percentage of correct detection increases exponentially with the number of features of the data at each site. Our experimental results show that for two large real datasets, this simple pruning strategy can detect more than 99% of the outliers that a centralized execution would find with less than 1% of the communication cost required for centralization. Figure 2 points out the intuition behind the rule for the 2 dimensional case. In this figure, the green dots represent the normal points while a single red dot represents the anomalous point. As seen, the red dot is quite far from the green dots. We argue that in order for this to happen, the distance along at least one of the axes will be large. In other words, most of the global outliers will be a local outlier in at least one of the distributed sites.

B. Detailed description

The overall distributed anomaly detection algorithm consists of two stages. The pseudo code for the first step is shown in Alg. 1. In this step, each node computes the local outliers

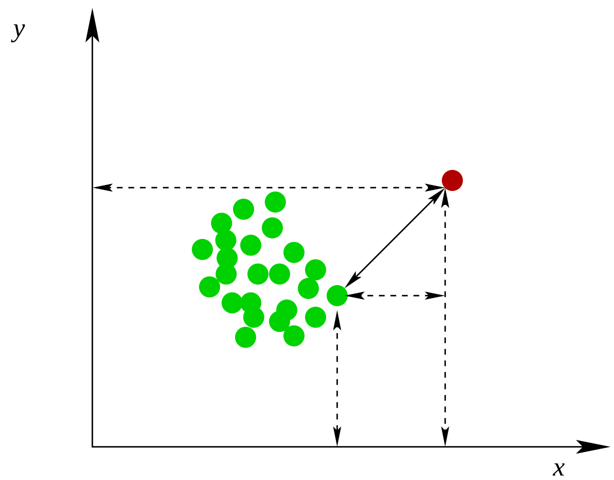


Fig. 2. This figure shows the basic idea of the pruning rule in 2-d. In this figure, the green dots represent the normal points while a single red dot represents the anomalous point. As seen, the red dot is far away from the green dots. The true distance between the red dot and the closest green dot is show by a bold arrow. The distance along the axes are shown using dotted lines. The observation is that for any true outlier, far away from any of the normal points, the distance along the axes will also be higher. Hence we can only analyze the local outliers from each site.

independently. The input to this local step are the dataset at each node D_i, the size of training set T_s, a seed s of the random number generator, and the parameter ν. The algorithm first sets the seed of the random number generator to s. Then it selects a sample of size T_s from D_i and uses it as the training set (T_i). The rest is used for the testing phase H_i. It then builds an SVM model M_i using T_i and ν. Once the model has been built, all points in H_i are tested using the set of support vectors defined by M_i. All those elements in H_i whose test score is negative is returned as the set of outliers $\mathcal{O}_i$.

In the second phase (Alg. 2), the local outliers are aggregated at the master site P_0. A sample of size T_s is drawn from each of the local sites D_i such that the same index (observation) is selected from each node. A global SVM model is then learned on this aggregated sample from all the sites. Each element of $\bigcup_{i=1}^{p} \mathcal{O}_i$ is tested against this global model to assign a score. All those elements in $\bigcup_{i=1}^{p} \mathcal{O}_i$ whose score is less than 0 is then reported as the true set of outliers $\mathcal{O}_d$ by the distributed algorithm.

V. Algorithm analysis

In this section we provide performance analysis of the distributed algorithm.

Algorithm 1: Local outlier detection at each node P_i, $i > 0$

```
Input: Dataset(D_i), Training sample size(T_s), ν, seed s
Output: Outlier set O_i
begin
    setseed(s);
    T_i = Sample(D_i, T_s);                              // Training data
    H_i ← D_i \ T_i;                                     // Test data
    M_i ← SVMTraining(T_i, ν);
    S ← SVMTest(M_i, H_i);                // Assign a score to each point in H_i
    for j=1 to |H_i| do
        if S(j) < 0 then
            O_i(j) ← [H_i(j)   S(j)];
        Send O_i to P_0;
```

Algorithm 2: Global outlier detection at P_0

```
Input: O_1, ..., O_p, Training sample size(T_s), ν
Output: Outlier set O_d
begin
    T = Sample(⋃_{i=1}^{p} D_i, T_s);     // Training data sampled from all sites
    H ← ⋃_{i=1}^{p} O_i;                                 // Test data
    M ← SVMTraining(T, ν);
    S ← SVMTest(M, H);                     // Assign a score to each point in H
    for j=1 to |H| do
        if S(j) < 0 then
            O_d(j) ← [H(j)   S(j)];
```

A. Correctness

Given a point $\overrightarrow{x} \in D$, Table I shows how $\overrightarrow{x}$ can be classified by both the distributed and the centralized algorithm.

We first analyze the case in which there is only one feature per node. Without any prior information about the data distribution, we assume that the data is drawn from an unknown distribution but sampled uniformly and independently for each feature.

Theorem 1 (True positive): Given a point $\overrightarrow{x} \in \mathcal{O}_c$, the probability of correct detection (true positive) of that point is given by

$$P(\overrightarrow{x} \in \mathcal{O}_d | \overrightarrow{x} \in \mathcal{O}_c) = 1 - \prod_{i=1}^{p} \left(1 - \frac{\rho_i}{R_i}\right),$$

where R_i is the maximum projection of the points along the i-th axis and ρ_i is the distance of

		Distributed algorithm	
		Normal	**Outlier**
Centralized algorithm	**Normal**	True positive	False positive
	Outlier	False negative	True negative

TABLE I
CONFUSION MATRIX FOR THE PERFORMANCE OF THE TWO ALGORITHM.

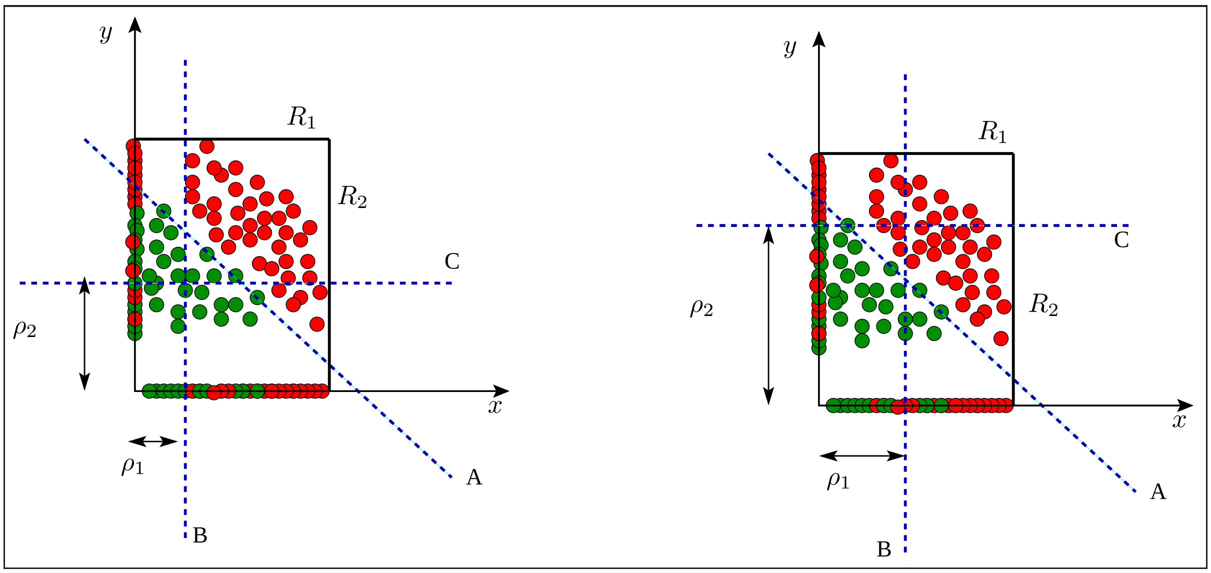


Fig. 3. This figure shows the different hyper planes drawn by the algorithm when using all the variables (A), only y-dimension values (B) and only x-dimension values (C). Note that different anomalies are found using the different hyper-planes.

the hyperplane along the i-th axis, both measured from the origin.

Proof: First we note that

$$\overrightarrow{x} \in \mathcal{O}_d \Longrightarrow \exists j, \quad \overrightarrow{x^{(j)}} \in \mathcal{O}_j.$$

From Figure 3, the distributed algorithm can be viewed as drawing hyperplanes, one for each dimension, which are the projection axis. Let ρ_i and R_i be the distance to the hyperplane and the maximum projection distance of the points along the i-th dimension from the origin. The probability that any point $\overrightarrow{x} \in \mathcal{O}_c$ does not belong to $\mathcal{O}_i$ for the i-th dimension is given by,

$$1 - \frac{\rho_i}{R_i}.$$

Since the data for each axis is drawn independently, the probability that $\overrightarrow{x}$ does not belong to any of the $\mathcal{O}_i$'s is given by

$$\prod_{i=1}^{p}\left(1-\frac{\rho_i}{R_i}\right).$$

Therefore the probability that the test point is included in at least one of the outlier sets is given by

$$1-\prod_{i=1}^{p}\left(1-\frac{\rho_i}{R_i}\right).$$

■

In the above expression, $\prod_{i=1}^{p}\left(1-\frac{\rho_i}{R_i}\right) \to 0$ as $p \to \infty$ since $\left(1-\frac{\rho_i}{R_i}\right) < 1$. Therefore, for large p, the true positive rate of the distributed algorithm approaches that of the centralized algorithm.

Following a similar argument, it is easy to show that the false negative rate of any point is

$$P(\overrightarrow{x} \in \mathcal{O}_d | \overrightarrow{x} \notin \mathcal{O}_c) = \prod_{i=1}^{p}\left(1-\frac{\rho_i}{R_i}\right).$$

Finally, we show that the false positive rate of the algorithm is 0. Note that in the second phase of the distributed algorithm, we sample data from the network and build an SVM. The resulting hyperplane is the same that would have been built if the entire dataset were at one location. Therefore, any point which is not an outlier according to $\mathcal{O}_c$ (*i.e.* one of the red circles in the figure), will also be tagged as a normal points by $\mathcal{O}_d$ since they both use the same hyperplane for testing the points. As as result, the false positive rate of the algorithm is zero.

B. Message complexity

The total number of bytes necessary to centralize all of the data at a single location and run the centralized outlier detection algorithm is:

$$m \times n_1 + m \times n_2 + \cdots + m \times n_p = m \times \sum_{i=1}^{p} n_i$$

For the distributed algorithm, we perform two rounds of communication. First, we centralize the outliers from all the sites and then we gather a sample of size T_s from all of them to build a global model and test the outliers found by each of the local sites. Note that for centralizing the outliers from site P_i, we need to visit other sites too in order to fetch the other dimensions. Thus, total number of messages is given by,

$$\underbrace{|\mathcal{O}_1| \times \sum_{i=1}^{p} n_i + |\mathcal{O}_2| \times \sum_{i=1}^{p} n_i + \cdots + |\mathcal{O}_p| \times \sum_{i=1}^{p} n_i}_{\text{centralizing outliers}} + \underbrace{T_s \times n_1 + \cdots + T_s \times n_p}_{\text{centralizing samples}}$$

$$= \sum_{j=1}^{p} |\mathcal{O}_i| \times \sum_{i=1}^{p} n_i + T_s \sum_{i=1}^{p} n_i = \sum_{i=1}^{p} n_i \times \left(T_s + \sum_{j=1}^{p} |\mathcal{O}_i| \right)$$

Now since $m \gg \sum_{i=1}^{p} |\mathcal{O}_i| + T_s$, the distributed algorithm is far more communication efficient than its centralized counterpart. We demonstrate this empirically in Section VI.

C. Running time

The running time for the traditional ν-SVM algorithm can be written as $O\left(m^2 \sum_{i=1}^{p} n_i\right)$ or $O\left(m \left(\sum_{i=1}^{p} n_i\right)^2\right)$, depending on the solution to the primal or the dual problem. In either of these two cases, distributed computing can reduce the running time by splitting n_i across several nodes. Therefore, the load at one node can be reduced from $O\left(m^2 \sum_{i=1}^{p} n_i\right)$ or $O\left(m \left(\sum_{i=1}^{p} n_i\right)^2\right)$ to $O(m^2 n_i)$ or $O(m n_i^2)$ respectively. This formulation can provide significant savings in terms of computational complexity at each node. We demonstrate this in the experimental section.

VI. Experimental evaluation

This section demonstrates the performance of the proposed algorithm on the California climate dataset and the Commercial Modular Aero-Propulsion System Simulation (CMAPSS) dataset.

A. Dataset description

The first dataset used in this paper is the MODerate-resolution Imaging Spectroradiometer (MODIS) Reflectance product MCD43A4 (version 5) which provides 500-meter reflectance data adjusted using a bidirectional reflectance distribution function (BRDF). The data is collected at intervals of every 8 days as an image file of size 1203×738 where each entry is saved as little-endian 32-bit float value. Each image is saved in 7 separate bands at different wavelengths. Along with the actual reflectance data for each pixel, we also have the latitude and longitude information for them. At the top level, the data is organized by year from 2001 to 2008. Under

Band	Spectral wavelength (nm)
1	620 - 670
2	841 - 876
3	459 - 479
4	545 - 565
5	1230 - 1250
6	1628 - 1652
7	2105 - 2155

TABLE II
SPECTRAL BAND FREQUENCIES FOR MODIS DATA ACQUISITION.

this top level directory structure are separate files for each band (1 - 7) and each 8-day period of the particular year. Within the period the best observations were selected for each location. Each of the files represent a 2D dataset with the naming conventions as follows:

$$MCD43A4.CA1KM.005. < YYYYDDD > . < BAND > .flt32$$

where $< YYYYDDD >$ is the beginning year-day of the period and $< BAND >$ represents the observations in particular (spectral) band (band 1 - band 7). The indexing is 0-based, ranging from 0 - 6 (where 0 = band 1, and 6 = band 7). The spectral band frequencies for the MODIS acquisition are as follows (see Table II):

The second dataset is a simulated commercial aircraft engine data. This data has been generated using the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) [21]. The dataset contains full flight recordings sampled at 1 Hz with 29 engine and flight condition parameters recorded over a 90 minute flight that includes ascent to cruise at 35000 feet and descent back to sea level. Interested readers can refer to this dataset at DASHlink[2].

B. Dataset preparation

In order to apply our anomaly detection method, we have performed the following preprocessing steps on the MODIS dataset:

- We remove all the pixels which have a fill value of -999.

[2] `https://c3.ndc.nasa.gov/dashlink/resources/140/`

- For each band and each image (per day) we first convert the 2-D matrix of pixels into a 1-D representation (as a simple vector) and then append these vectors over all the days and years to create a (very) long vector of intensities for this band. Combining for all the bands, we get the size of this matrix as $12,613,391 \times 7$.
- Along with this, we have also created a latitude and longitude matrix (each of size $12,613,391 \times 2$) for each element in the data matrix.
- We then split the data into 7 sites, each site having $12,613,391$ tuples.

Figure 4 shows the dataset and the final output of the preprocessing step.

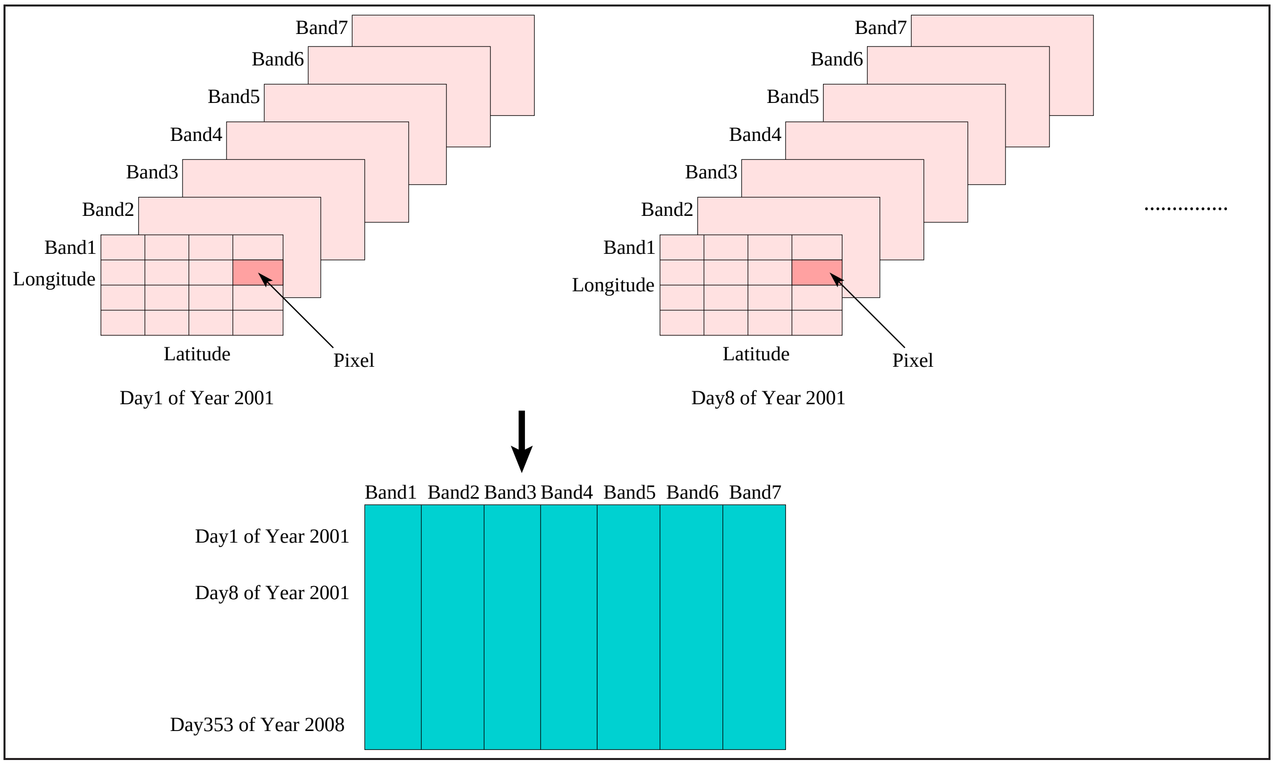


Fig. 4. This figure shows how the MODIS data set is structured. Each file is an image of size 1203×738. There are seven bands (separate images) for each of the 46 days per year (over 8 years), since data is saved every 8th day. The data contains of both the intensity and the latitude and longitudes for each location. First we take each (2-D) image containing the intensities as the pixels and convert it to a (1-D) vector. Then we append these vectors, thereby creating a very long vector. We do this separately for each of the bands, and concatenate them side by side (see figure for details).

For the CMAPSS dataset, we did not perform any preprocessing on the data. It has been used directly from the website. We only divided the data vertically to simulate the distributed scenario.

C. Measurement metric

In all of our experiments we measured these quantities: (1) the percentage of correct detection or detection rate, (2) the running time, and (3) the number of messages exchanged. By percentage of correct detection we mean the number of common outliers which are found both by our distributed algorithm and a centralized algorithm having access to all of the data but using the same sample size T_s for training as the distributed algorithm. When comparing running time, we plot the running time of our method and the centralized algorithm running on all the features. Note that, for our distributed algorithm since each site can run in parallel, we report the average running time over all the sites. We also report the total number of bytes transmitted by the distributed algorithm.

D. Performance evaluation on MODIS dataset

In this section we discuss the performance of the distributed algorithm on the California MODIS dataset. The first figure (Figure 5) shows how the detection rate (both mean and standard deviation) varies as the size of the training sample (T_s) is varied. The results are an average of 10 trials. We have varied T_s from 10,000 (0.79% of the entire dataset) to 1,000,000 (7.92% of the entire dataset). For a uniformly selected training set of size 10,000, the percentage of correct detection is 98.33. It remains almost a constant for different sizes of the training set. For 1 million test points, the correct detection rate is close to 99.79%. This shows that our algorithm is extremely accurate and returns the true set of outliers over different sample sizes. Note that in this context, true set of outliers refers to the outliers found by the centralized algorithm.

The next experiment demonstrates the gain of our algorithm with respect to running time. As shown in Figure 6, the running time of our algorithm diverges from the centralized algorithm as T_s is increased. For smaller T_s, the running time is comparable to the centralized algorithm. As T_s increases, our algorithm starts performing better. This is intuitive since with increasing size of training sample, more computation is needed and thus the running time of the centralized algorithm increases sharply. On the other hand, the distributed algorithm exhibits a slower growth in running time since the total processing load is distributed across all the processors. As shown in Section V-C, the distributed algorithm exhibits super linear complexity at each node which neatly concurs with the graph in Figure 6.

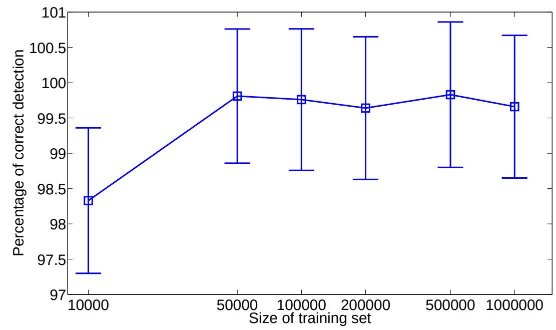


Fig. 5. Variation of the percentage of correct detection with the size of the training set as the latter is varied from 10,000 points (0.79% of the entire dataset) to 1,000,000 points (7.92% of the entire dataset) for MODIS. The samples are selected at random from the entire dataset. Percentage of correct detection means the number of anomalies detected by the distributed method compared to a centralized SVM algorithm using the entire dataset. As evident, the detection rate increases as the sample size increases.

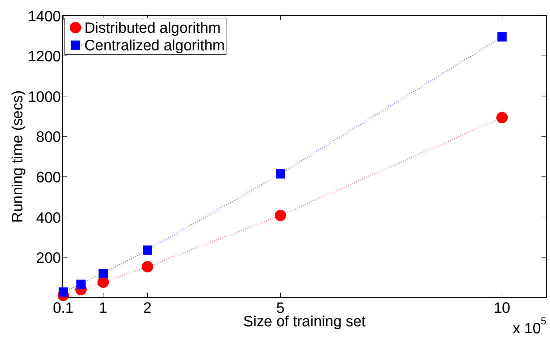


Fig. 6. Variation of running time with the size of the training set for MODIS. The samples are selected at random from the entire dataset. Both the running times of our algorithm and the centralized algorithm are shown. Clearly, the distributed algorithm outperforms the centralized one as the sample size increases.

Message complexity of the algorithm is demonstrated in Figure 7. The x-axis shows the number of samples used for the training and the y-axis refers to the ratio of the bytes transferred by the distributed algorithm to that of the centralized algorithm, expressed in percentage. Note that a value of $y = 100$ means that the distributed algorithm does not provide any communication savings. For all the cases, the percentage message complexity varies between 0.134 and 7.934. This shows that the proposed algorithm is highly communication efficient.

Figure 8 shows the top 50 outliers for training set size of 100,000. These outliers can be an outcome of any of the following underlying phenomenon such as change in vegetation due to fire, algorithmic problems with atmospheric corrections, clouded data, bad sensor or pixels corrupted during transmission. This is the general problem with Earth Science - the complexity

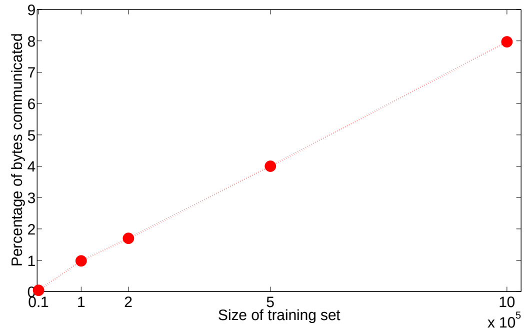


Fig. 7. Variation of the percentage of bytes communicated with the size of the training set for MODIS. The samples are selected at random from the entire dataset. The y-axis refers to the ratio of the bytes transferred by the distributed to the centralized algorithm, expressed in percentage. As depicted, the maximum percentage of bytes transferred is close to 8%, demonstrating the excellent scalability of the proposed algorithm.

of the system itself makes it extremely difficult to find the root cause for anomalies. Sometimes it may be due to a simple change in vegetation due to fire, but sometimes it may be caused by other changes hundreds or thousands of miles away. A thorough analysis of these results is beyond the scope of this work and will be reported in another paper.

E. Performance evaluation on CMAPSS dataset

Figure 9 shows the detection rate of the distributed anomaly detection algorithm for the CMAPSS dataset as the number of training samples is varied from 1,000 to 10,000. The left and the right figures show the same for 1 feature per site and 4 features per site respectively. As shown, the detection rate is high for both these scenarios.

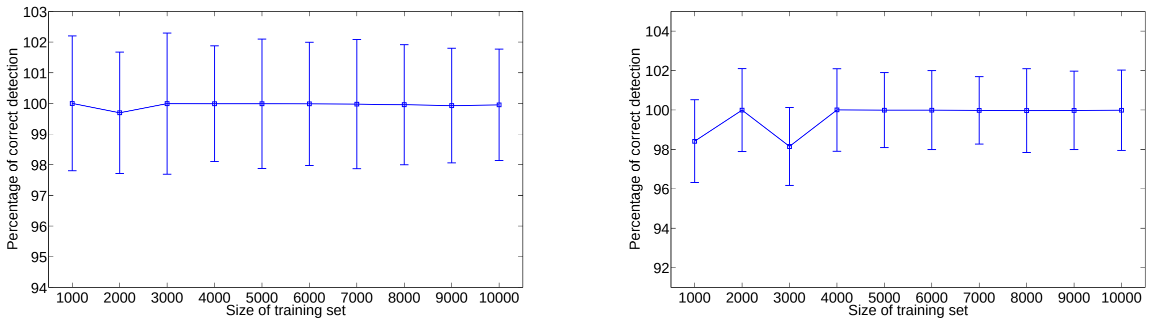
Figure 10 shows the runtime for the same CMAPSS dataset when the size of training set is increased. As expected, the distributed algorithm shows a much slower growth in running time compared to the centralized algorithm.

Our last set of experiments (Fig. 11) show the variation of detection rate and running time as the number of features per site are varied from 1 to 4. As expected, the detection rate shows no variation. This justifies the claim that out method is robust to the number of features at any location. On the other hand, the running time of the algorithm shows a sub linear growth, better than the theoretically derived quadratic growth as shown in Section V.

Fig. 8. Top 50 outliers detected by the distributed algorithm for T_s = 100,000.

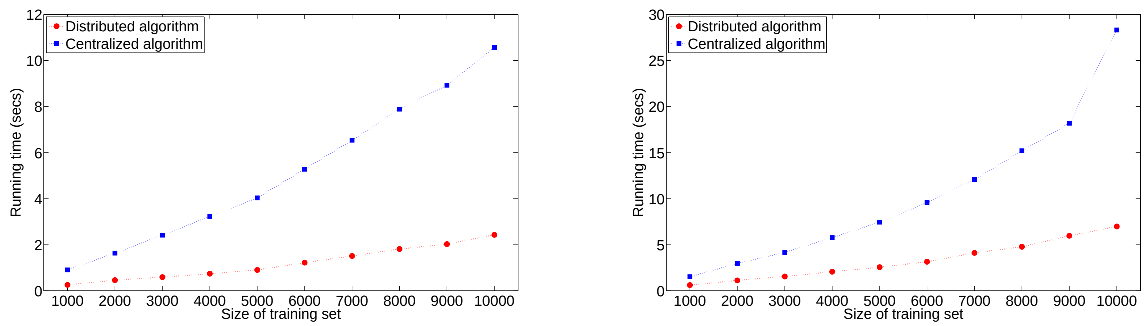
VII. Conclusion

In this paper we have presented a distributed algorithm capable of detecting outliers from distributed data where each site has a subset of the global set of features. To the best of the authors' knowledge, this algorithm is the first which does anomaly detection from vertically partitioned data in a communication efficient manner. Our pruning rule allows us to achieve high accuracy and low communication cost, a must for processing terabytes of data. We have provided a comprehensive theoretical analysis of the algorithm to show its gains. Experimental evaluation is conducted with the NASA MODIS satellite image dataset and the CMAPSS dataset. The results show that the algorithm is approximately 99% accurate with only 1% of the communication needed for centralizing all the data.



(a) 1 feature per site　　(b) 4 features per site

Fig. 9. Variation of the percentage of correct detection with the size of the training set as the latter is varied from 1,000 points to 10,000 points for CMAPSS. The samples are selected at random from the entire dataset. Percentage of correct detection means the number of anomalies detected by the distributed method compared to a centralized SVM algorithm using the entire dataset.



(a) 1 feature per site　　(b) 4 features per site

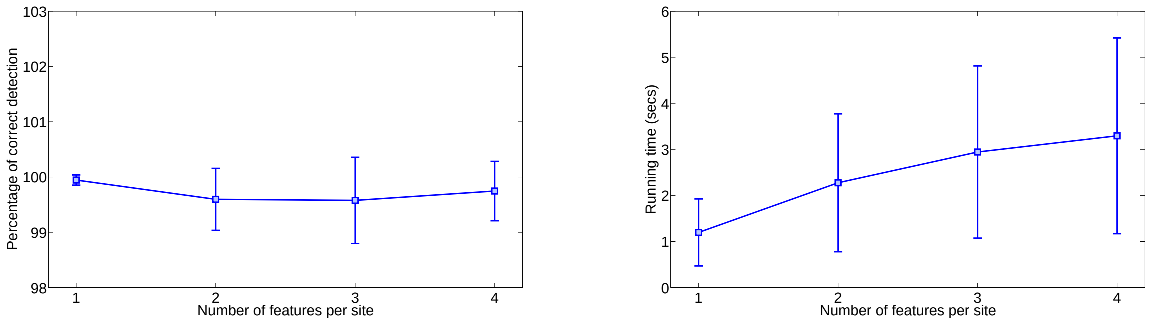
Fig. 10. Variation of running time with the size of the training set for MODIS. The samples are selected at random from the entire dataset. Both the running times of our algorithm and the centralized algorithm are shown. Clearly, the distributed algorithm outperforms the centralized one as the sample size increases.

ACKNOWLEDGEMENT

The data used in this paper are distributed by the Land Processes Distributed Active Archive Center (LP DAAC), located at the U.S. Geological Survey (USGS) Earth Resources Observation and Science (EROS) Center (`lpdaac.usgs.gov`). This work was supported by the NASA Aviation Safety Program, Intelligent Vehicle Health Management project and the TOPS project. The authors would also like to thank the reviewers for their valuable comments and suggestions.

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.



(a) Detection rate vs. number of features per site

(b) Running time vs. number of features per site

Fig. 11. Variation of the percentage of correct detection and running time with the number of features per site. The detection rate varies little while the running time shows a sublinear growth.

[2] S. D. Bay and M. Schwabacher, "Mining distance-based outliers in near linear time with randomization and a simple pruning rule," in *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003, pp. 29–38.

[3] S. Hido, T. Shohei, K. Yuta, H. Kashima, M. Sugiyama, and T. Kanamori, "Inlier-based outlier detection via direct density ratio estimation," in *ICDM '08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 223–232.

[4] E. M. Knorr, R. T. Ng, and V. Tucakov, "Distance-based outliers: algorithms and applications," *The VLDB Journal*, vol. 8, no. 3-4, pp. 237–253, 2000.

[5] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[6] F. Angiulli and C. Pizzuti, "Outlier mining in large high-dimensional data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 2, pp. 203–215, 2005.

[7] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 427–438, 2000.

[8] K. Das and J. Schneider, "Detecting anomalous records in categorical datasets," in *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007, pp. 220–229.

[9] W. Hu, Y. Liao, and V. R. Vemuri, "Robust anomaly detection using support vector machines," in *ICMLA '03: Proceedings of the International Conference on Machine Learning and Applications*, 2003, pp. 168–174.

[10] S. Barua and R. Alhajj, "A parallel multi-scale region outlier mining algorithm for meteorological data," in *GIS '07: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems*, 2007, pp. 1–4.

[11] D. Birant and A. Kut, "Spatio-temporal outlier detection in large databases," in *Proccedings of 28th International Conference on Information Technology Interfaces*, 2006, pp. 179 –184.

[12] J. Zhao, C. Lu, and Y. Kou, "Detecting region outliers in meteorological data," in *GIS '03: Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems*, 2003, pp. 49–55.

[13] S. Boriah, V. Kumar, M. Steinbach, C. Potter, and S. Klooster, "Land cover change detection: a case study," in *KDD '08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp.

857–865.

[14] C. Potter, V. Genovese, P. Gross, S. Boriah, M. Steinbach, and V. Kumar, “Revealing land cover change in california with satellite data,” *Transactions of American Geophysical Union*, vol. 88, no. 26, p. 269, 2007.

[15] S. Shekhar, P. R. Schrater, R. R. Vatsavai, W. Wu, and S. Chawla, “Spatial contextual classification and prediction models for mining geospatial data,” *IEEE Transactions on Multimedia*, vol. 4, pp. 174–188, 2002.

[16] J. Branch, B. Szymanski, C. Giannella, R. Wolff, and H. Kargupta, “In-network outlier detection in wireless sensor networks,” in *International Conference on Distributed Computing Systems*, 2006, p. 51.

[17] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[18] K. Bhaduri and H. Kargupta, “A scalable local algorithm for distributed multivariate regression,” *Statistical Analysis Data Mining*, vol. 1, no. 3, pp. 177–194, 2008.

[19] R. Wolff, , K. Bhaduri, and H. Kargupta, “A generic local algorithm for mining data streams in large distributed systems,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 4, pp. 465–478, 2009.

[20] A. Schuster and R. Wolff, “Communication-efficient distributed mining of association rules,” in *SIGMOD ’01: Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, 2001, pp. 473–484.

[21] D. K. Frederick, J. A. DeCastro, and J. S. Litt, “User’s Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS),” *NASA Technical Manuscript*, vol. 2007-215026, 2007.