Does Feature Matter: Anomaly Detection in Sensor Networks

Rui Li[†], Kebin Liu[*§], Yuan He[*§], Jizhong Zhao[†]
[†]Department of Computer Science and Technology, Xi'an Jiaotong University, China
[*]TNLIST, School of Software, Tsinghua University, China
[§]CSE Department, Hong Kong University of Science and Technology, China
Email: {rli, zjz}@mail.xjtu.edu.cn, {kebin, heyuan}@cse.ust.hk

ABSTRACT

Anomaly detection, for uncovering faults and failures, is a crucial task for wireless sensor networks (WSNs). There have been substantive research efforts in this field such as source-level troubleshooting, rule-based inference, and time sequence event analysis. Most existing approaches, however, rely on the collection of a large amount of information. Due to the lack of management on information features, the redundancy of collected information greatly degrades the efficiency of diagnosis in large-scale WSNs. To address this issue, we propose RFS (Ranking-based Feature Selection), a three-stage approach to efficiently select representative feature sets for diagnostic tasks and effectively characterize the network status. RFS is a compatible component that can be integrated with most state-of-the-art diagnosis approaches. We conduct extensive experiments based on a large-scale outdoor WSN system, GreenOrbs, to examine the performance of RFS. The results demonstrate that RFS achieves effective anomaly detection in a large-scale WSN with low overhead.

Categories and Subject Descriptors

C.2 [**Computer System Organization**]: Computer Communication Networks; C.2.3 [**Computer Communication Networks**]: Network Operations—*Network management, Network monitoring*

General Terms

Design, Performance

Keywords

Anomaly Detection, Wireless Sensor Network

1. INTRODUCTION

Wireless Sensor Networks (WSNs) enable wide-spread applications, such as environment surveillance [12], clinical monitoring and health care for human bodies with Body Sensor Networks [2], etc. Body Sensor Networks (BSNs) are a generation of wireless sensor networks that attracts many efforts as well [7]. Due to the self-organized nature and unreliable wireless communications, WSNs, however, are vulnerable to anomalies and anomaly detection in WSNs is far from trivial. The unreliable wireless links and software/hardware failures may lead to packet loss, data faults, and network dynamics. To make it even worse, it is usually hard to fully trace the running status of an operational system, making diagnosis and management extremely challenging. When network failures occur, it is often infeasible to manually check the deployed nodes in an in-situ manner [3], especially in some hostile environments, e.g., GreenOrbs [15].

As measures to tame the faults, failures, resource constraints and uncertain behaviors in WSNs, research on debugging and diagnosis receive plenty of attention. Early studies on debugging focus more on the designs and tools for testbeds, simulations, and emulations [11]. Source-level debugging tools are studied in [1], which lack resiliency and robustness against post-deployment anomalies. A diagnosis process has to collect a large amount of information from the network, unless the diagnosis approach is appropriately designed [14]. Existing diagnostic methods usually assume that increase in the amount of collected information brings better performance of anomaly detection. But a fundamental question is often overlooked:

Is all this information equally useful for anomaly detection?

The answer is no. Information-intensive diagnosis approaches often suffer from high cost to uncover faults, since they need various types of information as input parameters. For example, the statistical data for diagnosis in GreenOrbs contains more than 20 dimensions, however, packet retransmission times and no ACK retransmission times are redundant information for diagnosis. How to eliminate the redundant information without sacrificing diagnostic performance remains unsolved.

Most of the diagnostic tools run at the sink side of a WSN, utilizing the global network data. Tremendous demand on data columns from the network makes diagnostic tools unaffordable. As a result, how to reduce data dimensions, using as few features as possible to characterize a WSN is non-trivial and needs more efforts. To push the problem one step further, which are representative features of a WSN with respect to anomaly detection? How to make the best

BodyNets 2011, Beijing, China, 7-8 November 2011, ISBN (978-1-936968-29-9)

BODYNETS 2011, November 07-08, Beijing, People's Republic of China

DOI 10.4108/bodynets.2011.247058

feature selections for anomaly detection?

Correlations provide guidance to data reduction in the diagnosis process. We propose RFS, Ranking-based Feature Selection, which traces the changes of a feature subset, so as to characterize the system state of a WSN. The main approach of RFS is a three-stage workflow, including feature ranking, correlation-based feature subset selection, and validation of feature subset selection. To evaluate the effectiveness and efficiency of RFS, we carry out extensive experiments based on the real data sets from GreenOrbs system, which was launched in early 2009 and has been in continuous operation ever since then.

Major contributions of this study are as follows:

1. We explore the information redundancy based on a real operating WSN system, and propose a simple but efficient approach to deal with the redundant data.

2. We propose a three-stage approach RFS, which employs ranking-based feature selection to generate representative feature subset. With subset changes over time, the anomalies can be detected efficiently.

3. We evaluate RFS through extensive experiments using data traces from GreenOrbs and validate the effectiveness and efficiency of RFS under different network scenarios.

The rest of this paper is organized as follows. Section 2 briefly discusses the related literature on diagnosis and feature selection. Section 3 presents a motivating example, followed by the elaboration on the design of RFS in Section 4. Section 5 describes the network data traces that we use to evaluate RFS. Section 6 evaluates RFS and illustrates case studies of GreenOrbs' anomalies. We conclude and discuss the plan of future work in Section 7.

2. RELATED WORK

In this section, we briefly discuss the literature of diagnosis in WSNs and techniques of feature selection.

2.1 Diagnosis in WSNs

For the most debugging tools [13, 17] that target at identifying software bugs in sensor nodes, are belong to before-deployment diagnosis. Declarative Tracepoints [1] reports a debugging system, which could automatically watch monitors program states to detect bugs. The system does not modify the application source code and just inserts a SQL-like language to the debugging states. Clairvoyant [19] enables the source-level debugging for WSNs which allows users to remotely execute debugging commands, such as step and breakpoint. Dustminer [8] focuses on troubleshooting interactive complexity bugs by mining discriminant patterns from the event logs stored on sensor nodes. Debugging tools are effective in finding network failures, however, they often incurring huge control and storage overhead.

Operational period diagnosis attracts many research efforts. SNMS [9] constructs network infrastructure for logging and retrieving state information at runtime. EmStar [4] supports simulation, emulation, and visualization of an operating sensor networks. Sympathy [16] actively collects metrics, such as data flow and neighbor table, from sensor nodes and determines the root-causes based on a decision tree scheme. PAD [14] reports the concept of passive diagnosis which leverages a packet marking strategy to derive network state and deduces the faults with a probabilistic inference model. And due to the self-organize nature and resource constraints of sensor networks, enterprise network diagnosis approaches are infeasible for WSNs.

2.2 Feature Selection

Feature selection algorithms can be classified into two categories: feature ranking and feature subset selection. For feature ranking, it ranks the features by eliminating the features that do not achieve the metric score, according to certain metrics, such as RELIEF [5]. For feature subset selection, it searches through a probability set for optimal subset selection. Wrappers [10] utilizes the learning machine of interest as a black box to score subsets of variable according to their predictive power. Filters [20] selects subsets of variables during a pre-processing step, which is independent from the chosen predictor. Embedded [6] approaches perform variable selection in the process of training and are usually specific to given learning machines.

In short, feature selection in the views of statistics and machine learning merely use features for data classification and prediction. They only consider the characteristics like data distribution and data types, which cannot be directly applied to the cases of anomaly detection in WSNs. The combination of feature ranking and feature subset selection is a promising direction in diagnosis and anomaly detection of WSNs.

3. A MOTIVATING EXAMPLE

Canonical correlation coefficients are widely used methods for featuring relevance measurements. They achieve high performance when the feature scores and target scores are monotonic. We adopt canonical correlation coefficients to measure the data collected in our GreenOrbs project, where the data is monotonic in statistics. For instance, we calculate the correlation coefficient scores under relative low traffic load. And find that perfectly correlated features, in other words, the correlation coefficient score is greater than a threshold (take 0.95 for example), is really of great redundancy; however, relatively high feature correlations (take the correlation coefficient score from 0.75 to 0.95 for example) cannot be easily identified as redundancy. It may be of great complement to anomaly detection. Table 2 shows the score of correlated features in a cycle of packet reception. Here, we use $\{f_1, f_2, \ldots, f_7\}$ to denote seven statistical features of node i, and the detail description of each feature is shown in Table 1.

3.1 How does correlation influence feature redundancy?

Table 2 shows that the correlation coefficient score between `TransmitNoACKRetrans` (f_3) and `RetransmitCounter` (f_5) is 1, since the `TransmitNoACKRetrans` (f_3) is a subset of `RetransmitCounter` (f_5). So we ignore this score, as they do not satisfy the statistical independent assumption. Let us take a look at correlation coefficients between feature pairs (f_2, f_4) and (f_2, f_6), of which the scores are both 0.8894, exhibiting relative high correlations. It can be found that with more packet transmissions, the number of dropped packets and duplicate packets increases. This matches our basic observations, if we simply throw any one feature, we will

Table 1: Description of Each Feature

Features	Description
`ReceiveCounter` (f_1)	Number of packets received
`TransmitCounter` (f_2)	Number of packets transmitted
`TransmitNoACKRetrans` (f_3)	Number of No ACKed packets retransmitted (transmission failed but retransmission does not exceed the threshold)
`TransmitNoACKDrop` (f_4)	Number of No ACKed packets dropped (transmission failed and retransmission exceed the threshold)
`RetransmitCounter` (f_5)	Number of packets retransmission
`DuplicateCounter` (f_6)	Number of duplicate packets
`ParentChange` (f_7)	Number of parent node changes

Table 2: Feature Correlations in GreenOrbs Under Relative Low Traffic Load

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
f_1	1	–	–	–	–	–	–
f_2	.9996	1	–	–	–	–	–
f_3	.2939	.3142	1	–	–	–	–
f_4	.0658	.8894	.8140	1	–	–	–
f_5	.2926	.3129	1	.8165	1	–	–
f_6	.8927	.8894	.2781	.0701	.2770	1	–
f_7	.1804	.2021	.9774	.7401	.9769	.1670	1

miss some state changes. Real redundancy exists between the feature pair (f_1, f_2), where the correlation coefficient is 0.9996. In fact, the number of packet receptions and number of transmissions are redundant for us to detect anomalies in WSNs. These observations enable us to choose representative features and thus eliminate redundant information.

3.2 Can a feature which is useless by itself be useful with others?

For features f_1, f_2, f_6, if we focus on anyone of them, we get mere information. Those features, however, if taken as a whole, can provide valuable evidences to us. It holds true for the other features in the example(f_3, f_4, f_5, f_7). Furthermore, we need to find features which can indicate the state of WSNs. Through the changes of combined representative features, we can detect anomalies. So the useless one feature may be useful when combined with other features in the representative feature subset.

4. DESIGN OF RFS

4.1 Main Idea of RFS

Firstly, we introduce the main idea of Ranking-based Feature Selection (RFS), and an overview of RFS is described in Figure 1. RFS contains three steps to process the received data packets. For information-intensive WSNs, information as a whole can be used to detect anomalies; however, most of the information-intensive diagnostic approaches may meet

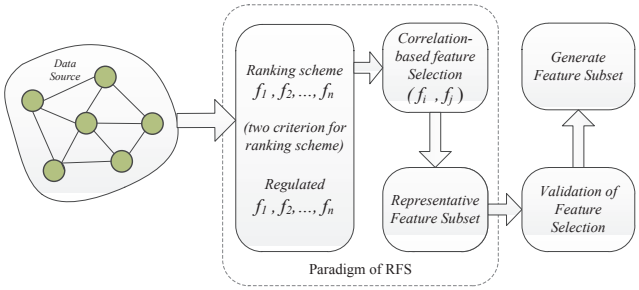

Figure 1: Workflow of RFS

disasters of high data dimensions, especially in large-scale sensor networks.

In the following sections, a packet is represented by a vector of p feature scores. S denote a set of instances of size n. F is the given feature set $\{f_1, f_2, \ldots, f_p\}$. An instance X is denoted by a p-dimensional vector $(x_1, x_2, \ldots, x_p)$, where x_j stands for the score of feature f_j of X.

4.2 Phase I — Feature Ranking

Many feature selection algorithms include feature ranking as a principal or auxiliary selection scheme since its simplicity, scalability, and good empirical success [18]. In this section we consider using correlation criteria to rank features. The feature ranking can be beneficial, since we could find informative features, or these features may be of redundancy.

For the feature f_i with score x_i and the feature f_j with score y_j, we define the canonical correlation coefficient as:

$$\rho(f_i, f_j) = \frac{cov(f_i, f_j)}{\sqrt{var(f_i)var(f_j)}} \quad (1)$$

Where cov denotes the covariance and var stands for the variance. The estimate of is given by:

$$\rho(f_i, f_j) = \frac{\sum_i (x_i - \overline{x})(y_j - \overline{y})}{\sum_i (x_i - \overline{x})^2 \sum_j (y_j - \overline{y})^2} \quad (2)$$

Where the bar notation stands for an average over the index p. This coefficient is also the cosine between vectors f_i and f_j, some of features may be correlated positively, and some negatively. If x and y are completely correlated, the score of $\rho(f_i,f_j)$ is 1 or -1, if x and y are thoroughly uncorrelated, ρ (f_i,f_j) is 0. We rank different features based on absolute score of correlation coefficient.

According to this definition, we could rank the features into classes for the features of which hold same number of high correlated features. Therefore, we rank the revealing example in Section 3, and find that f_2, f_3, f_4 and f_5 can be categories into a class, and f_1, f_6, f_7 can be classified into another class. Hence, we need to rank them in decreasing correlation coefficient order according to the below two criteria. The criteria are for further ranking of features through all existing features.

Criterion 1. For feature (f_i) that has more correlated features rank higher.

We identify two features as correlated features if the absolute score of their correlation coefficient is greater than a defined threshold. For instance, in the motivating example, there exists 7 features and each feature has correlated with

another one. We calculate the number of correlated features for each feature, f_1 , f_2, ..., f_7 respectively and the results are 3, 4, 4, 4, 4, 3, 3. As a result, for the features from f_2 to f_5, they rank higher than f_1, f_6 and f_7. This criterion demonstrates that for the features of which contain more redundant information rank higher.

Criterion 2. If two or more features get the same number of the correlated features, for features which have greater average correlation coefficient score rank higher.

According to Criterion 1, we cannot tell the ranking difference from f_2 to f_5, however, we can calculate their mean score with any other features in this class, such as for f_1, f_6 and f_7, and the mean score for each feature is 0.9641, 0.9274, and 0.9848 respectively. Then for the three features, they rank f_7, f_1, f_6 with decreasing order of the calculations. It shows that in this class of features, f_7 contains more redundant information that the left two features can denote the current status of data collection.

Why do we regard f_7 as the representative feature among `ReceiveCounter`(f_1), `DuplicateCounter`(f_6) and `ParentChange` (f_7)? In running status, with `ParentChange` (f_7) grows, `TransmitNoACKRetrans` (f_3) and `DuplicateCounter` (f_6) both increase in response. That means parent change could influence packet receive state, and the receive packet numbers is simultaneously influenced by the duplicate packets. However, if there is a perfect correlation between two features, we could certainly eliminate this feature, as the feature must be redundant in statistics. According to criterion 2, we can rank the features of different classes in a decreasing order.

4.3 Phase II — Correlation-based Feature Selection

When all the features are in a statistical decreasing order, we need to select features with enough effective information which can be representative features to indicate the current system status. Can we simply choose the features with high rankings, eliminating the perfect correlated features, to be the representative features to denote the current running status? No, we cannot just choose the high ranked features. Simply choose them will lead to misunderstanding of the correlated features, since high ranked features may correlated with each other.

We select the high ranked features in the same class that contain redundant information with high probability. Hence, we illustrate this process into two steps, firstly we choose the features in the same class with maximized correlation coefficient, and secondly we aim to choose minimized correlation coefficient of features between different classes [18]. Therefore, the features are in statistically with maximized effective information.

If a group of k features has already been selected, correlation coefficients may be used to estimate correlation between this group and the class, including inter-correlations between the features. Relevance of a group of features grows with the correlation between features and classes, and decreases with growing inter-correlation. Denoting the average correlation coefficient between these features and output variables as $r_{kf} = \overline{\rho}(f_k, F)$ and the average between different features as $r_{kk} = \overline{\rho}(f_k, f_k)$ the group correlation coefficient measuring the relevance of the feature subset can be defined as:

$$R(f_k, F) = \frac{k r_{kf}}{\sqrt{k + k(k-1) r_{kk}}} \qquad (3)$$

This formula is obtained from linear correlation coefficient of Pearson with all features standardized. We can use it in the correlation-based feature selection for adding or removing features at a time. Correlation-based feature selection is a simple filter approach that ranks feature subsets according to a correlation based heuristic function. The bias of the function is toward subsets that contain features which are highly correlated with the phase one selected classes and uncorrelated with each other.

Irrelevant features should be ignored because they will have low correlation with the class. Moreover, with best first search strategies [5], the feature subset can be finally chosen as the representative feature of the current running status. The two phases are described in Algorithm 1.

Algorithm 1: Ranking-based Feature Selection

Input: $F = \{f_1, f_2, \ldots, f_n\}$, θ (defined according to various applications).

Output: Representative feature subset $F_r = \{f_1, f_2, \ldots, f_m\}$, where $m \leq n$.

for f_i, f_j $(i \neq j)$ **do**
  Calculate $\rho(f_i, f_j) = \frac{\mathrm{cov}(f_i, f_j)}{\sqrt{\mathrm{var}(f_i)\mathrm{var}(f_j)}}$;
  Ranking feature according to Criterion 1 and 2;
end
$F \leftarrow \{f_i, \ldots, f_j$, where $1 \leq i < j \leq n$, $k = j - i + 1\}$;
Calculate $r_{kf} = \overline{\rho}(f_k, F)$ and $r_{kk} = \overline{\rho}(f_k, f_k)$;
Then, $R(f_k, F) = \frac{k r_{kf}}{\sqrt{k + k(k-1) r_{kk}}}$;
if $R(f_k, F) \leq \theta$ **then**
  f_k is eliminated from the subset;
else
  $F_r \leftarrow f_k$;
end
return stop;
end

The time complexity of correlation-based feature selection is relatively low. It just requires $p((n^2 - n)/2)$ operations for computing the pairwise feature selection matrix, where p is the number of instances and n is the number of features through the overall data retrieval cycle.

For the procedure of RFS, we can select feature subsets through calculating the $R(f_k, F)$ between features and classes. For each feature, we can calculate the $R(f_k, F)$ with phase one decided classes. Equation 3 forms the core of this phase and imposes a ranking on feature subsets in the search space of all possible feature subsets. Exhaustive enumeration of all possible feature subsets is prohibitive due to time and complexity limits, the organization of the search, start point, and stopping condition should be addressed as well. We here adopt backward elimination search strategy [5] which begins with the full feature set in a class and greedily removes one feature at a time as long as the evaluation function does not degrade. And the backward elimination is efficient than forward selection and best first strategy.

Firstly, we calculate $R(f_k, F)$ among a group of selected features and ranked class of features to obtain the score of $R(f_k, F)$. Secondly, we choose subset of features which has minimized relevance with other features among different classes, and maximized relevance with the features in their class. The procedure of the feature selection is described in

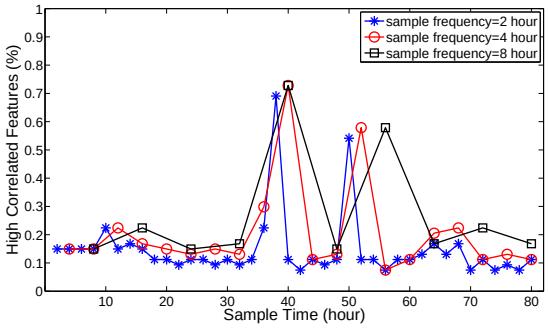

Figure 2: Percentage of high correlated features over all features

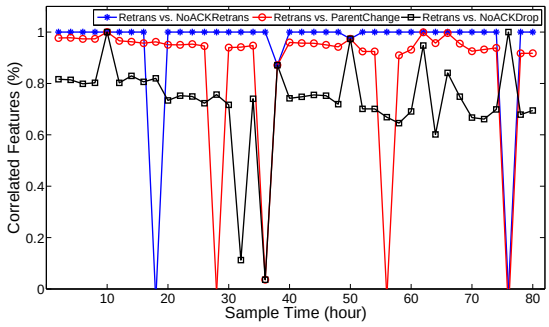

Figure 3: Correlation coefficients between redundant features and representative features

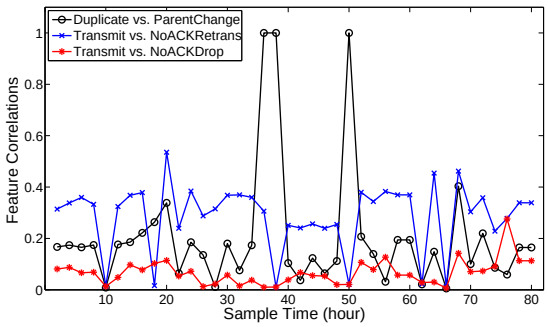

Figure 4: Correlation coefficients between representative features

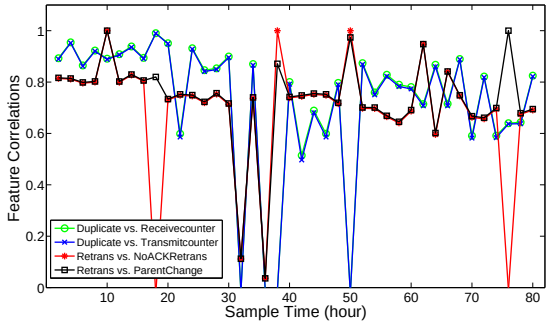

Figure 5: Correlation coefficients between features with similar changes

Figure 1. For instance, we still choose ranked features based on stage one from the motivating example, and calculate the feature correlation through equation 3 to get the subset of features.

4.4 Phase III — Validation of Feature Selection

From two phases of RFS approach, the representative features of the motivating example are `TransmitCounter` (f_2), `TransmitNoACKRetrans` (f_3), `TransmitNoACKDrop` (f_4), `DuplicateCounter` (f_6) and `ParentChange` (f_7), respectively. To validate the accuracy of feature selection, we take *leave-one-out* cross validations for accuracy validation. It is a special case of k-fold cross validation where k is set to the number of initial tuples. That is, only one sample is "left out" at a time for the test subset.

Figure 2 shows the percentage of high correlated features over all correlated features with sample time changes. As we can see from Figure 2, most of the times, the high correlated features are of a small portion around 15% of all correlated features; however, for the sample point from 35 to 60, the high correlated features are up to 70%. That makes a strong hint that with redundant features increasing, the diagnostic efficiency can degrade. As a result, with the change of representative features the current network effective information can be traced.

As Figure 3 shows, the correlation coefficient between retransmissions and No_ACK_Retransmissions always at 1 since the No_ACK_Retransmissions are a subset of retransmissions. So the retransmission packets are definitely a redundant feature as we illustrated in Section 3. Notice that the correlation at 18 hour is a statistical error since no anomaly happens at this time, so our approach may face such a false alarm, however, statistical error merely happens during the sample period.

As a contrast, Figure 4 describes the representative features that have chosen from the example, the feature correlations stay low during the same sample period compared to Figure 2 and Figure 3, only the correlation coefficient between duplicate packet numbers and parent change times experiencing a sharp increase within three sample periods. That illustrate the representative features we choose are of low correlation can contain less redundant information, the spikes can be used to indicate representative feature subset changes to identify occurrence of anomalies.

Figure 5 denotes the features contain redundant information; we need to eliminate features with redundancy. We can find that duplicate packets as a representative feature has a similar correlated pattern with `TransmitCounter` and `ReceiveCounter`. Through RFS, we eliminate `ReceiveCounter` as redundancy which shows the decision in RFS is of effectiveness. So does the `RetransmitCounter` for the other two correlations.

5. PERFORMANCE EVALUATION

Our experiments are conducted with data traces collected from GreenOrbs system during the period January - July 2010. The data traces correspond to two consecutive operational periods and contain 4,495,769 packets. In order to conduct comprehensive feature correlations, during the first

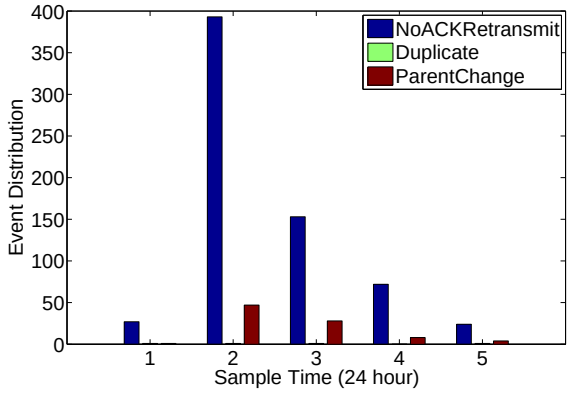

Figure 6: Event distribution under low traffic load

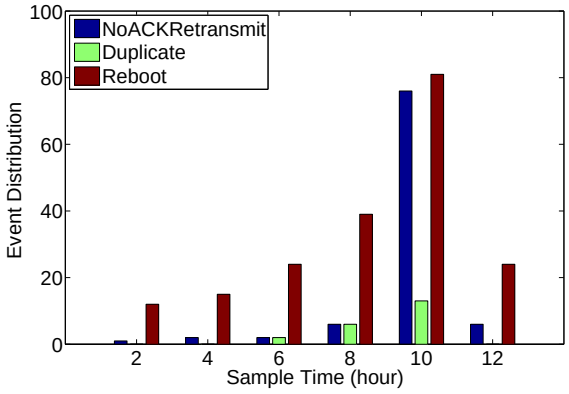

Figure 7: Event distribution under high traffic load

period we regulate the nodes to work with different parameter settings, such as transmission power, routing protocol, and duty cycle, etc. We evaluate RFS respectively under relatively low traffic load and high traffic load. Through the experiments, we target validate whether the dynamic representative feature subset really work as an indicator of the running status of a WSN. Meanwhile, our real system-based evaluation may help one to better understand features in a WSN, which can be used for anomaly detection.

5.1 Data Set of GreenOrbs

We denote the back end data set, namely the entire data set collected at the sink, by D_{sink}. D_{sink} is made up of three categories of traces. We mainly use one category of the traces for evaluation, which include the nodes' statistical data.

Data traces of a node's statistics are denoted by T_{stats}. T_{stats} includes the following metrics: the cumulative time of radio power-on, the cumulative number of received packets, the cumulative number of transmitted packets, the cumulative number of packet drops (due to receive queue overflow, transmit queue overflow, and transmit timeout), the cumulative number of transmissions that are not ACKed, the cumulative number of retransmissions, the cumulative number of received duplicate packets, and the cumulative number of parent changes in CTP-based multi-hop routing.

5.2 Methodology

We have implemented RFS approach with GreenOrbs' diagnostic information. GreenOrbs employs up to 500 TelosB motes with a MSP430 processor and CC2420 transceiver. We modify a subset of TinyOS 2.1 components to embed the RFS-related diagnostic functions into the software on sensor nodes.

We evaluate RFS in three aspects. First, we demonstrate the efficacy of selected features to represent the system's running status and present our explanations based the concept of feature correlations. Second, we examine the changes of representative features over time and illustrate that anomaly can be detected when there is a change of such features. Third, we share our insights on anomaly detection. Such insights are obtained from the implementation experience of GreenOrbs.

5.3 Anomalies in GreenOrbs

Here are some basic observations that indicate anomalies are likely to take place, if network configurations are changed. Figures 6 and 7 plot the events detected under low traffic load and high traffic load, respectively. As we can see, the number of duplicate packets is always small. The count of No_ACK_Retransmission shows a similar trend with the count of reboots under high traffic load. This indicates frequent retransmits may be one of the main reasons that causes node to reboot. Parent changes might cause NO_ACK_Retransmission is another result we may infer from Figure 6.

5.4 Results of RFS with GreenOrbs

We implement RFS with our diagnostic tools for GreenOrbs and evaluate its performance and overhead during anomalies detection. We use two metrics for measuring the diagnostic performance of RFS, namely *detection ratio* and *false alarm ratio*. The detection ratio is defined as the ratio of the number of anomalies detected to the number of all the anomalies that happened. A high detection ratio efficiently helps our diagnostic tool to recover the WSN system from anomalies. The false alarm ratio is the ratio of false alarms to all the anomalies that happened. A low false alarm ratio indicates that the diagnostic tool has high accuracy.

Figures **??** and **??** show the results of anomaly detection under low traffic load and high traffic load, respectively. According to Figure **??**, RFS successfully identifies over 80% of all the anomalies. The false alarm ratio is lower than 10%. Figure **??** describes the detection ratio and false alarm ratio under high traffic load. Compared to Figure **??**, the detection ratio stays stable, but the false alarm ratio slightly goes up. That indicates RFS has a bit lower accuracy when the traffic load is high. This is because RFS relies on rule-based diagnostic tools for anomaly detection. When traffic load increases, both the network data set and the representative feature subset become more complicated. As a result, more unpredictable anomalies happen.

Now we evaluate the overhead of RFS. We use the following metric to measure the overhead: the ratio of traffic amount incurred by anomaly detection to the amount of all the network traffic. The results are shown in Figures **??** and **??**. According to Figure **??**, RFS uses less than 5% of all the network traffic under low traffic load, while the overhead increase to nearly 7% in the last sampling period under high traffic load. We find that under low traffic load, the over-

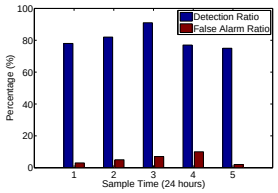

Figure 8: Detection ratio and false alarm ratio under low traffic load

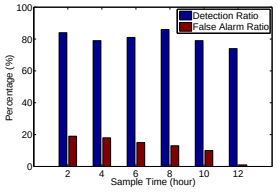

Figure 9: Detection ratio and false alarm ratio under high traffic load

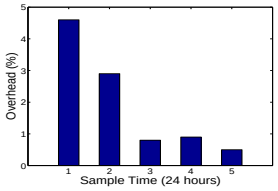

Figure 10: Overhead under low traffic load

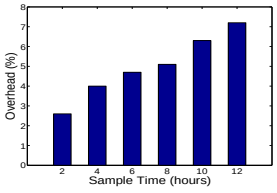

Figure 11: Overhead under high traffic load

head of RFS indicates large variations of feature sets over time.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose RFS, a novel three-stage ranking-based feature selection approach for anomaly detection in WSNs. Due to the lack of techniques for managing information in most of diagnostic tools, diagnosing WSNs is often information-intensive and laborious work. We in this work disclose a significant fact: changes of the representative features are important indicators of WSN anomalies. Based on this fact, we propose RFS to solve the above mentioned problem, using a feature selection approach. Although feature selection method is not new in the fields of statistics and data mining, it is effective in information reduction in WSNs. Our experiments demonstrate that RFS is a practical and efficient approach for anomaly detection.

For future works, we plan to study the issue of using representative feature correlations for failure troubleshooting. Moreover, the statistical correlations of sensor data can be better used as node health indicator and network performance indicator.

Acknowledgements

We would like to thank all members in GreenOrbs project (http://www.greenorbs.org) for their contributions to this work. This work is supported in part by China NSFC under Grants No.61170213, No.61103187 and No.60873262, and China 973 Program under Grants No.2011CB302705.

7. REFERENCES

[1] Q. Cao, T. Abdelzaher, J. Stankovic, K. Whitehouse, and L. Luo. Declarative tracepoints: A programmable and application independent debugging system for wireless sensor networks. In *ACM SenSys*, 2008.

[2] O. Chipara, C. Lu, T. Bailey, and G. Roman. Reliable clinical monitoring using wireless sensor networks: experiences in a step-down hospital unit. In *ACM SenSys*, 2010.

[3] W. Dong, Y. Liu, X. Wu, L. Gu, and C. Chen. Elon: enabling efficient and long-term reprogramming for wireless sensor networks. In *ACM SIGMETRICS*, pages 49–60. ACM, 2010.

[4] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. Emstar: A software environment for developing and deploying wireless sensor networks. In *USENIX ATC*, 2004.

[5] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh. Feature extraction: Foundations and applications. 2006.

[6] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1):389–422, 2002.

[7] E. Jovanov, A. Milenkovic, C. Otto, and P. De Groen. A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation. *Journal of NeuroEngineering and rehabilitation*, 2005.

[8] M. M. H. Khan, H. K. Le, H. Ahmadi, T. F. Abdelzaher, and J. Han. Dustminer: Troubleshooting interactive complexity bugs in sensor networks. In *ACM SenSys*, 2008.

[9] M. M. H. Khan, L. Luo, C. Huang, and T. Abdelzaher. Snts: Sensor network troubleshooting suite. In *IEEE/ACM DCOSS*, 2007.

[10] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: Accurate and scalable simulation of entire tinyos applications. In *ACM SenSys*, 2003.

[12] M. Li and Y. Liu. Underground coal mine monitoring with wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 5(2):10, 2009.

[13] P. Li and J. Regehr. T-check: Bug finding for sensor networks. In *IEEE/ACM IPSN*, 2010.

[14] K. Liu, M. Li, Y. Liu, M. Li, Z. Guo, and F. Hong. Passive diagnosis for wireless sensor networks. In *ACM SenSys*, 2008.

[15] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X.-Y. Li, and G. Dai. Canopy closure estimates with greenorbs: Sustainable sensing in the forest. In *ACM SenSys*, 2009.

[16] N. Ramanathan, K. Chang, L. Girod, R. Kapur, E. Kohler, and D. Estrin. Sympathy for the sensor network debugger. In *ACM SenSys*, 2005.

[17] T. Sookoor, T. Hnat, P. Hooimeijer, W. Weimer, and K. Whitehouse. Macrodebugging: Global views of distributed program execution. In *ACM SenSys*, 2009.

[18] H. Wei and S. Billings. Feature subset selection and ranking for data dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 162–166, 2007.

[19] J. Yang, M. L. Soffa, L. Selavo, and K. Whitehouse. Clairvoyant: A comprehensive source-level debugger for wireless sensor networks. In *ACM SenSys*, 2007.

[20] Z. Zhu, Y. Ong, and M. Dash. Wrapper-filter feature selection algorithm using a memetic framework. *IEEE Transactions on Systems Man and Cybernetics*, 2007.