Detecting Time-Related Changes in Wireless Sensor Networks Using Symbol Compression and Probabilistic Suffix Trees

YuanYuan Li, Michael Thomason, and Lynne E. Parker

Abstract— Our research focuses on anomaly detection problems in unknown environments using Wireless Sensor Networks (WSN). We are interested in detecting two types of abnormal events: sensory level anomalies (e.g., noise in an office without lights on) and time-related anomalies (e.g., freezing temperature in a mid-summer day). We present a novel, distributed, machine learning based anomaly detector that is able to detect time-related changes. It consists of three components. First, a Fuzzy Adaptive Resonance Theory (ART) neural network classifier is used to label multi-dimensional sensor data into discrete classes and detect sensory level anomalies. Over time, the labeled classes form a sequence of classes. Next, a symbol compressor is used to extract the semantic meaning of the temporal sequence. Finally, a Variable Memory Markov (VMM) model in the form of a Probabilistic Suffix Tree (PST) is used to model and detect time-related anomalies in the environment. To our knowledge, this is the first work that analyzes/models the temporal sensor data using a symbol compressor and PST in a WSN. Our proposed detection algorithm is distributed, "light-weight", and practical for resource constrained sensor nodes. We verify the proposed approach using a volcano monitoring dataset. Our results show that this approach yields the same performance as the traditional Markov models with much less cost.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) are widely used in environment monitoring applications. Each sensor in the network can monitor its local region and communicate via wireless communication with other sensor nodes to collaboratively produce a high-level representation of the environmental states. With limited power resources, it is important to reduce the amount of data to be transferred through the wireless network. In prior work, we have proposed a hierarchical, distributed, and modular anomaly detection system (see [4] and [5]) that does not require central decision making. The system uses a neural network to perform sensor fusion from multiple sensors and uses a heuristic discrete state time model to capture time relations among different states. In this paper, we extend our prior work by significantly enhancing the discrete state machine through the use of a more robust, high performance and memory efficient anomaly detection method. This method consists of symbol compression that extracts the semantic meaning from the raw data, along with a Probabilistic Suffix Tree that is a much more efficient method for anomaly detection.

Our main research objectives are twofold. First, we are interested in capturing the semantic meanings from multi-dimensional raw sensor signals (at each time instant) by classification. Second, we are interested in analyzing the underlying sensor event sequence information. An active volcano may have days or months of inactivity followed by an eruption lasting only minutes. A WSN that is designed to model time sequences should be able to model this process and be able to detect eruptions. However, it is infeasible to model the volcanic activity using a traditional high order Markov chain, since an L-th order Markov model requires $|\Sigma|^L$ states, where $|\Sigma|$ denotes the number of alphabet symbols and L is the length of past history/memory being modeled.

For a large variety of time-related sequential data, statistical correlations decrease rapidly with the distance between symbols in the sequence. If the statistical correlations are indeed decreasing, then there exists a "memory" length M such that the empirical probability changes very little if conditioned on subsequences longer than M. Ron et al. [8] proposed an elegant solution to this problem. The underlying observation in that work is that in many natural sequences, the memory length depends on the context and therefore is not fixed. Therefore, as in [8], we propose to use a Variable Memory Markov (VMM) model to preserve the minimal subsequences (of variable lengths) that are necessary for precise modeling of the given statistical source. This results in a more flexible and efficient sequence representation. It is particularly attractive in cases where we need to capture higher-order temporal dependencies in some parts of the behavior and lower-order dependencies elsewhere. The VMM model can be implemented in the form of Probabilistic Suffix Tree (PST) model. A PST is a tree whose nodes are organized such that the root node gives the probability of each symbol of the alphabet while nodes at subsequent levels give next-symbol probabilities conditioned on a combination of one or more symbols having been seen first.

The constructed PST model is essentially a symbolic predictive model: the underlying continuous time signals are first abstracted to a discrete space, analogous to a set of finite categories. In some cases, this has the advantage of being more immune to the problems of noise while still preserving the essential underlying patterns or dependencies that govern behavior in the observed domain. Arguably, it also produces a more understandable model since its components are higher level abstractions. All these advantages of the PST model make it suitable to model temporal sequences in WSNs.

One of the challenges of using a PST model on resource constrained sensor nodes is that the model complexity may grow exponentially with the memory depending on the data, which makes it impractical for resource constrained sensor

Y. Li, M. Thomason, and L.E. Parker are in the Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37996, USA, {yli,thomason,parker}@eecs.utk.edu.

nodes. For example, in the volcano monitoring dataset, it is common to have a period of inactivity that extends over several days, weeks or months. Using PST models directly to model the sensor data is computationally prohibitive. Thus, we use a data driven approach to automatically infer discrete and abstract representations (symbols) of primitive object interactions. These symbols are then used as an alphabet to infer the high level structures of typical interactive behaviors using PST models. This Markov model represents high level semantic notions. These environment feature descriptors are invariant of the sensor classes and time dependencies. They constitute the input to a statistical learning framework where discrete representations of interactive behaviors can be learned by modeling the probability distribution of the feature vectors within the interaction feature space.

Symbolic modeling and processing have several advantages over continuous measurements and models, including: 1) sensor data is often discrete (i.e., certain radar systems [12]); 2) environments that are modeled with discrete states that have clear physical interpretations are natural and easy for humans to interpret (e.g., eruption or no eruption vs. vibration measurements); and 3) data compression techniques, which we use to reduce the size of the observations, typically require discrete state representations.

The rest of this paper is organized as follows. We first review related work in Section II. We then present our proposed approach in Section III, which includes our network architecture. In Section IV, we test the approach on a volcano monitoring dataset. Finally, we summarize our findings in Section V.

II. RELATED WORK

Various regression models have been proposed for time-related analysis in WSNs (e.g., [11]). Regression models involve a complex parameter estimation process and may suffer from model mismatch and bias-variance trade-off problems. There has been some work on the use of probabilistic time-series models in WSNs, such as the Kalman Filters (e.g., [2]). In general, Kalman Filter-based models are sophisticated and require significant computation. The fixed length Markov model is another commonly used technique for time series analysis [12]. Examples of fixed order Markov models include the Markov Chain, Hidden Markov Model, etc. Due to the limited resources in WSNs, building high and fixed order Markov models is not feasible. The VMM is a data-driven Markov model that supports online learning and online detection. Note that in practice, the VMM is usually implemented in the form of a PST. Many researchers have proposed algorithms to build PST models in linear time and space (e.g., [1]). In addition, the method is automatic, and can be applied without assuming any preliminary information. PST models have been applied in WSNs for object tracking in (e.g., [10]). To the best of our knowledge, our work is the first work that makes use of a PST model to detect time-related anomalies in WSNs. Our results show that a PST has the same performance as the same order Markov model with less memory cost.

III. PROPOSED APPROACH

In this section, we first introduce the overall architecture, followed by the description of symbol compression method, PST model construction and a likelihood-ratio based anomaly detector.

A. Architecture for the WSNs

As addressed in our previous works [4] and [5], we employ a hierarchical learning/communication structure for our WSN. The sensor nodes in the WSN are divided into clusters, as shown in Fig. 1. Each cluster has a clusterhead and multiple cluster members. Each cluster covers a geographic region and is responsible for detecting the environmental changes in that region. Both cluster members and clusterheads run an identical detection system — a classifier, a symbol compressor, a time analysis model and a missing data estimator. Cluster members read in raw sensor readings, o_i, from the environment as input, and then perform sensor fusion and classify data into categories c_i. If an anomaly is detected by the classifier, an abnormal alert is raised. After the classification process, cluster members compress their category labels and send the compressed category labels s_i (higher level semantic symbols in temporal sequences) to their clusterheads. Note that in our previous system, the cluster member would send the class labels after each time step. By compressing the class label sequence into a shorter temporal sequence, we are able to save communication costs by removing the redundancies in the temporal sequence. The amount of redundancy removed depends on the compression ratio of the original temporal sequence. Then, the clusterhead analyzes the compressed sequences at a higher level. If an anomaly is detected by the time analysis module, an abnormal alert is raised. Clusterheads often cannot receive complete labels from all of their cluster members, due to unstable wireless communications. Ignoring missing data would result in too many false positives (as justified in our previous work [5]). Thus, the clusterheads first preprocess the collected category labels by identifying and estimating the missing values (using our technique described in [5]). The learning system has a hierarchical structure — clusterheads may have higher level clusterheads which classify their output (compressed) class labels. Finally, the root node obtains the final model of the environment. With this learning architecture, the system is able to detect both abnormal environmental changes and time-related changes.

We keep our design in a modular form because it gives human operators the flexibility of turning off the modules that are not needed. For example, the operator can turn off the time analysis module if analyzing time is not of interest. In addition, the WSN can be easily scaled to a large number of sensors. At the same time, this hierarchical approach reduces communication, which in turn saves energy in the WSN.

Fig. 2 demonstrates the flow of the sensor signals of an individual sensor node in our proposed system, which is a significant extension over our prior design in [4]. Each sensor node takes in multi-dimensional raw sensor vectors from the environment as input, and then performs sensor
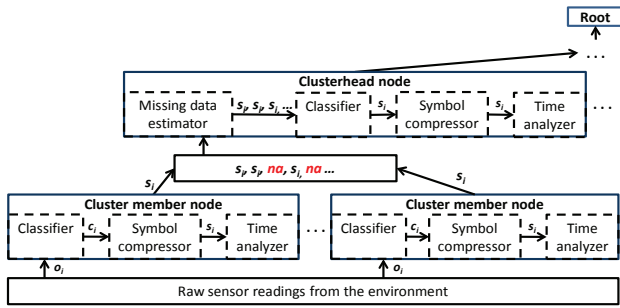
Fig. 1. The overall learning architecture for WSN. All sensor nodes run the same programming image. Depending on the role of the sensor node (clusterhead/cluster member), the assigned modules are activated.
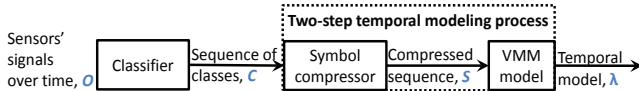

Fig. 2. The proposed architecture for an individual node.

fusion and classifies the raw sensor signals into categories using a classifier. If the new category does not match one of the existing normal categories, a sensor level anomaly is detected by the system. The classifier that we use is the Fuzzy Adaptive Resonance Theory (ART) neural network [3]. The details of the Fuzzy ART neural network algorithm can be found in [4]. After the sensor fusion and categorization process is finished, the system further checks whether there are time-related changes. The temporal sequence model is a two-step process. First, the sequence of classes/categories is first compressed with a symbol compressor (described in Section III-B). It operates efficiently by sampling the string statistics in a manner that allows a compressed representation and exact reconstruction of the original string. Depending on the implementation, learning and generating can be off-line or on-line, real-time or non-real-time. Then the compressed events are built into a PST model. In order to detect time-related changes, the system measures the likelihood of the current compressed sequence, compared to the PST model(s) learned during the training phase.

Note that we propose this two-step structure to make our system more flexible when capturing the temporal sequence information. For example, system designers may substitute another symbol compression method or Markov-based model as desired. When modeling the temporal sequence, if time duration within a state (class) is not of interest, the designers can simply remove all consecutive repeated categories in the symbol compression stage and let the PST model only the state sequences.

B. Identifying semantic symbols from temporal categories

One significant extension to our previous work in [4] is adding the ability to identify semantic symbols from temporal categories. Identifying semantic symbols is of particular interest since it allows reasoning to be extended from individual temporal categories to a higher semantic level. In this research we are interested in modeling the joint (combined) behavior of pairs of interacting semantic symbols. These semantic symbols can be constructed using compression algorithms. Compression algorithms can roughly be categorized into lossless or lossy compression [9]. Lempel-Ziv77/78 and Lempel-Ziv-Welch (LZW) are the most widely used dictionary-based compression techniques [9]. The main mechanism in both schemes is pattern matching: find string patterns that have occurred in the past and compress them by encoding a reference to the previous occurrence. Our temporal sequence compression process is defined as follows. The encoder/compression algorithm takes a sequence of category labels $C = \{c_1, \cdots, c_T\}$ and compresses it into another sequence denoted by S, which encodes higher-level semantic meaning. We have used the LZW algorithm to construct semantic symbols. LZW is an improved implementation of the Lempel-Ziv78 algorithm. The algorithm is designed to be fast to implement but is not usually optimal because it performs only limited analysis of the data. With the limited resources of a WSN, it is important to reduce the amount of processing. In a typical compression run, the algorithm takes a string as input, and processes the string using a dictionary of distinct substrings. The algorithm encodes the string (and builds the dictionary) by making a single left to right traversal of a sequence. Initially, the dictionary contains the alphabet with its corresponding encoding. Then it sequentially adds every new substring that differs by a single last character from the longest match that already exists in the dictionary. This repeats until the string is consumed. The idea is that, as the string is being processed, it populates the dictionary with longer strings, and allows encoding of larger blocks of the string at each replacement.

Note that with the proposed two-step temporal modeling process, it is not necessary to keep all entries in the dictionary. Especially with limited memory in the wireless sensor nodes, we wish to build a small dictionary. If a system can afford to build a PST with order up to M, dictionary entries with length shorter than M can be pruned, since they can be modeled by an M-th order PST. The length of each dictionary entry corresponds to a real-world event in discrete time. Specifically, the length of a single-alphabet entry in the dictionary denotes the time duration of the event's occurrence and the corresponding index carries that semantic meaning. Thus, the compressed temporal sequence is able to maintain the higher-level semantics of the initial temporal sequence while using a shorter length. In WSNs, transmitting a compressed sequence saves communication costs compared to transmitting uncompressed raw data.

C. Modeling semantic interactions using PSTs

Another significant extension to our prior work in [4] is modeling semantic interactions using PSTs. The PST model, which was originally designed for classification purposes, has the advantage of improved extraction of statistical information from sequences. The trade-off is that it deliberately throws away some of the original sub-sequences during the analysis process to maintain a compact representation. In resource constrained WSNs, compact data models save

energy in nodes by reducing the amount of data being transferred among the nodes. We are interested in building models of the environment that are able to support both interpretation and anomaly detection. We achieve this by using PSTs to efficiently encode the sequences of the categories corresponding to observed interactive behavior in the feature space. In the following, the definition of the PST is first given. Then, we explain how the PST is used to detect time-related anomalies in the WSN.

The PST is a stochastic model that uses a suffix tree as the index structure. This approach is based on the "memory" of natural sequences. That is, the root node of the PST gives the empirical probability of each symbol in the alphabet while each node at subsequent levels is associated with a vector that gives the next symbol given the label of the node as the preceding segment. For example, $P(s_{i+1}|s_0...s_i) = P(s_{i+1}|s_{i-M}...s_i)$, where $i > M$, gives the empirical probability distribution P of the next symbol s_{i+1} given the last M symbols in the preceding segment. Furthermore, a tree of order M has M levels beyond the root. Detailed PST inference and pruning procedures can be found in [6].

Let $S = \{s_u : u \in U\}$ denote a compressed temporal sequence of size U, where u denotes the discrete time unit after compression. To model the normal behavior using the Maximum Likelihood criterion, we find a model that maximizes the probability of a given sequence of observations. Given a PST λ, the total likelihood of the observations can be expressed mathematically as $L = P(S|\lambda)$. If the probability of the observation sequence given the model is below a threshold θ, then an anomaly is detected. A likelihood-ratio detector is addressed in detail in the following subsection.

The computation for this procedure is fast and inexpensive. The PST model has been shown to be implementable in linear time and space [1]. Let the length of the training sequence be n, the memory length of PST be M, the alphabet be Σ, and the length of a testing sequence be k. Apostolico and Bejerano's PST building algorithm takes $O(n|\Sigma|)$ time [1]. This procedure is a one-time overhead cost to the system during the initial period (unless the PST needs to be updated). To detect anomalies after the PST is constructed, the system has to calculate the likelihood that the testing sequence matches the built PST. The sequence matching procedure is a simple tree traversal, and the detection procedure takes $O(mk)$ time. Thus, it is practical for sensor nodes that have limited resources.

D. Likelihood ratio detection scheme

We are interested in detecting anomalies in an unknown environment. Therefore, we assume that the initial model that we have learned is the normal model of the environment; variations from the initial model are treated as anomalies. Let S denote the entire (compressed) sequence of environmental states $S = \{\tilde{S}, \hat{S}\}$, where $\tilde{S}$ denotes the training sequence, and $\hat{S}$ denotes the target testing sequence. Given a segment of observations/class labels $\tilde{S} = \{s_1, \cdots, s_v\}$, and a segment of observations/class labels $\hat{S} = \{s_{v+1}, \cdots, s_U\}$, the task of detecting anomalies in the sequence $\hat{S}$ is to determine

if $\hat{S}$ is the same as $\tilde{S}$. (Note that in some applications the types of anomalous events are known; in that case $\tilde{S}$ can represent abnormal observations, and an anomaly is detected when $\hat{S}$ matches $\tilde{S}$.) The likelihood-ratio detector is similar to that proposed by Reynolds, et al., in [7] for speaker verification problems. The anomaly detection problem is formulated as determining if sequence $\hat{S}$ is different from the normal sequence $\tilde{S}$. The anomaly detection task can be restated as a hypothesis test between:

$$H_0 : \hat{S} \text{ is normal} \quad \text{and} \quad H_1 : \hat{S} \text{ is abnormal.} \quad (1)$$

The likelihood ratio test to decide between these two hypotheses is given by:

$$\frac{p(\hat{S}|H_0)}{p(\hat{S}|H_1)} \begin{cases} \geq \theta & \text{accept } H_0 \\ < \theta & \text{reject } H_0 \end{cases} \quad (2)$$

where $p(\hat{S}|H_i), i = 0, 1$, is the probability density function for the hypothesis H_i evaluated for the observed sequence $\hat{S}$, also referred to as the likelihood of the hypothesis H_i given the sequence. The decision threshold for accepting or rejecting H_0 is θ. For our temporal anomaly detection, the null and alternative hypothesises use PST models λ. Hence, we denote the PST model for the null hypothesis as $p(\hat{S}|H_0; \lambda_0)$ and for the alternative hypothesis as $p(\hat{S}|H_1; \lambda_1)$. The likelihood ratio detector is $p(\hat{S}|H_0; \lambda_0)/p(\hat{S}|H_1; \lambda_1)$. Usually, the logarithm of this statistic is used giving the log-likelihood ratio,

$$\Lambda(\hat{S}) = p(\hat{S}|H_0; \lambda_0) - p(\hat{S}|H_1; \lambda_1) \quad (3)$$

During the training period, we may encounter various event sequences that are normal, abnormal and/or undetermined. The model for H_0 can be estimated using a normal event sequence. However, the model for H_1 is less well defined since it potentially must represent every abnormal situation possible. Since the environments that WSNs operate in are typically unknown, it is not possible to train H_1 with every abnormal situation. Therefore, we believe the universal background model (UBM) [7], which uses all hypothesized sequences of events, is more suitable. The UBM in our application is the entire set of training data, which may include normal, abnormal and undetermined event sequences.

IV. EXPERIMENTS

A. Volcano Reventador data

We use a volcano monitoring dataset to test the proposed anomaly detection system. The dataset was collected by Werner-Allen, et al., over a 24-hour time period at the Volcano Reventador [13]. The data used in our experiments is obtained from one of the seismic stations, which samples the environment at 120 *Hz*. Since the data is voluminous, we have analyzed one day's worth of data, which is approximately 200 megabytes. This data is from a single station (a single sensor node), which is not from a complete sensor network. The following experiments that we conduct do not involve clusters or clusterheads. However, the detection process for the entire WSN work is described in Section III. We have demonstrated in our prior work [4], [5] that the

previously proposed detection system works with multiple layers of clusters of sensor nodes that use multiple sensors in our lab environments. Our previous approach uses heuristic discrete states to model time; since the 2-step temporal modelling approach proposed in this paper also uses discrete state representations, we believe this should work with the existing system architecture when implemented on the physical nodes. The current approach is more robust when detecting multiple anomalies in the environment and takes less communication by transmitting compressed temporal sequences. We use the volcano dataset here as proof-of-concept to illustrate how our enhanced time modeling module works on a real world dataset. In future work, we will evaluate our system using a full WSN.

B. Preprocessing

The raw seismic data O is normalized and categorized by the Fuzzy ART neural network. The raw sensor data O is classified into $|\Sigma| = 13$ categories by setting the vigilance parameter ρ of the Fuzzy ART system to 0.93. Hence, the output temporal categorization sequence C has an alphabet size $|\Sigma|$ of 13. Note that the physical meaning of these 13 categories is unknown. In future work, if we have the ground truth of the seismic data, the Fuzzy ART neural network can assign more meaningful categories (e.g., magnitudes of eruptions) by adjusting its vigilance parameter ρ. The data is grouped into hour-long subsets. Based on visual inspection, there are no volcanic activities during the period between the 1st hour and the 15th hour; we regard this period as the normal period. There is an anomaly/eruption between the 15th hour and the 16th hour. Thus, we regard this hour as an abnormal period. We are unable to determine whether the periods following the eruption are "normal after eruption" or "abnormal". As a result, we currently discard the data after the 16th hour. In summary, we treat hours $1-15$ as a normal period and hour 16 as an abnormal period.

The following performance metrics are used to evaluate our system: compression ratio, True Positive Rate (*TPR*), True Negative Rate (*TNR*), False Positive Rate (*FPR*) and False Negative Rate (*FNR*).

C. PST vs. Fixed length Markov model

We first compare the performances between the PST model and the traditional fixed length Markov model on the volcano monitoring dataset. In this experiment, we use the abnormal period's category sequence (hour 16) as training data and the normal periods' (hours 1-15) category sequences as testing data. The performance is measured by the negative log-likelihood of the normal category sequence given the observation of the abnormal category sequence. Specifically, we construct both PST and fixed length Markov models from the training data with Markov orders 1, 2, 3, 4, 5, and 10. For each PST/fixed length Markov model, we calculate the negative log-likelihood $P(C|\lambda)$ of the testing sequence C given the PST/fixed length Markov model λ. The larger the negative log-likelihood value is, the more dissimilar are the compared sequences. We expect the dissimilarity between

the abnormal period and the normal period to grow as the memory order grows.

Our results are summarized in Table I. The empirical results indicate that the sizes of PST models are much smaller than the traditional fixed length Markov models as the order increases. For example, observe that the 10th order fixed length Markov model uses 981 states, while the order-10 PST model only uses 205 states. The negative log-likelihood is the same between a sequence given a PST model and a fixed length Markov model with the same order, since we eliminate nodes that have the same probabilities as their parent nodes when constructing the PST models. The model is lossless in terms of capturing the information of the training data. Therefore, we prefer a PST model over a fixed-length Markov model because it is purely data-driven, it is flexible, and most importantly, it takes less space. Note that the PST models can be pruned to remove some low probability nodes (see [8]); however, this will lead to information loss. In future work, we will evaluate the proposed detection system based on PST models with thresholds to remove low probability nodes. In addition, we will explore a systematic procedure for deciding the threshold values for PST models in the proposed detection system.

TABLE I
COMPARISON OF FIXED LENGTH MARKOV MODELS VS. PST MODELS

	Order	Num. of nodes	$-$ log-likelihood
Fixed length Markov	1	12	-0.0141
	2	45	-0.0112
	5	296	-0.0069
	10	**981**	-0.0046
PST	1	12	-0.0141
	2	41	-0.0112
	5	146	-0.0069
	10	**205**	-0.0046

D. PST model with compressed temporal sequence

As shown in the previous section, using the PST model directly on the volcano monitoring dataset is still not practical for the resource limited sensor nodes, i.e., it takes 208 states to build a PST model with a memory of 10 observations. An observation with 10 samples can hardly capture any meaningful sequences in the environment monitoring type of applications. In reality, daily life activities usually take more than 10 observations to capture. If modeling such a long sequence of actions is important to an application, the PST model will not be a practical solution if used directly on the raw samples. Our proposed compression technique, in which the PST is built from the higher level symbol representation rather than the original data, addresses this issue.

We applied the standard LZW symbol compression algorithm on the category sequences of the dataset. The average dictionary size for 16 hours (excluding the pre-built characters in the dictionary) is approximately 61 entries per hour. Based on the given data, we observe that there are long periods of inactivity, which are denoted by sequences of category "1". Then, we apply the compression algorithm to

all 16 hours of data independently. The average compression ratio for the 16 hours is approximately 33:1. The compressed representation of data saves on both processing power and reduces the amount of storage on the sensor nodes. Most importantly, when the local sensor nodes transmit the temporal models to the clusterheads, they are able to save the transmission power as well.

TABLE II
PERFORMANCES FOR PST ORDERS 5 AND 10 W. COMPRESSION

PST order	Threshold	*FNR*(MissRate)	*FPR*(FalseAlarm)
5	**0**	**17.5%**	**16.8%**
	0.5	0%	85.9%
	0.9	8.8%	34.9%
	1.1	22.2%	10.7%
	1.5	99%	1.8%
10	0	1.8%	6.8%
	0.5	0%	77%
	0.9	0%	17.6%
	1.1	**2.3%**	**3.4%**
	1.5	97.7%	0.9%

To detect anomalies in the temporal sequence, we employ a likelihood-ratio detection scheme as given in Eq.(2). The detection procedure works as follows: during the training period, we use semantics (compressed symbols) from hours 1-16 to obtain the alternative hypothesis $p(S|H_1)$, and hour 16 to obtain the null hypothesis $p(S|H_0)$. We have built two PST models for the dataset with orders 5 and 10. Table II shows the confusion matrix obtained when adjusting the threshold θ. Based on the detection performances, the *FPR* for the order-5 PST model is 16.8% when $\theta = 0$, which has the best tradeoff between *FPR* and *TPR*. The *FPR* for the order-10 PST model is 3.4% when $\theta = 1.1$, which has the best tradeoff between *FPR* and *TPR*. Therefore, we choose θ values of 0 and 1.1 for PSTs of orders 5 and 10, respectively. In addition, the miss rates for PST orders 5 and 10 PST models are 17.5% and 2.3%, respectively. The miss rates and false alarm rates are relatively low for both PST models. The tree sizes for the orders 5 and 10 PST models are 186 and 241 nodes, respectively. Note that the nodes on PST models that are built from compressed sequences represent higher semantic temporal meanings. Therefore, the nodes represent much longer observations compared to the nodes on the PSTs that are built from the uncompressed sequences, (i.e., the order 10 PST is modeling a sequence with approximately 330 observations compared to 10 observations). The detection results show that our proposed PST model with symbol compression method is able to detect anomalies with high performance. Additionally, the UWB-based likelihood ratio detector is robust and able to detect multiple anomalies in a time sequence with high performance. The robustness and the ability of detecting multiple anomalies are significant enhancements to our previous heuristic state machine.

V. CONCLUSIONS

Using resource constrained WSNs for environment monitoring of applications such as volcanic eruptions is challenging, because the event of interest is usually preceded by a long period of inactivity and the event itself lasts only for a short period of time. In this paper, we have proposed a two-step temporal modeling procedure to analyze and extract semantic symbols from a sequence of observations. The system detects time-related changes online by using a likelihood-ratio detection scheme. Our proposed temporal modeling technique is able to capture high-order temporal dependencies in some parts of the behavior and lower-order dependencies elsewhere. We have verified the proposed approach using a volcano monitoring dataset. Results show that our system yields a high performance. Our iterative temporal learning approach captures the temporal dependencies in data and removes redundancies, which translate into energy savings in the WSN. The algorithm is distributed, and supports a hierarchical learning structure, which we believe will scale to a large number of sensors and will be practical for resource constrained sensor nodes. In future work, we will implement and evaluate the time analysis module on our wireless sensor network (rather than a single sensor on a single node).

REFERENCES

[1] A. Apostolico and G. Bejerano. Optimal amnesic probabilistic automata or how to learn and classify proteins in linear time and space. *Journal of Computational Biology*, pages 25–32, 2000.

[2] A. Jain, E. Y. Chang, and Y. Wanf. Adaptive stream management using Kalman filters. In *The ACM International Conference on Management of Data*, 2004.

[3] A. Kulakov and D. Davcev. Tracking of unusual events in wireless sensor networks based on artificial neural-network algorithms. In *Information Technology: Coding and Computing*, 2005.

[4] Y. Li and L. E. Parker. Detecting and monitoring time-related abnormal events using a wireless sensor network and mobile robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[5] Y. Li and L. E. Parker. A spatial-temporal imputation technique for classification with missing data in a wireless sensor network. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[6] G. Mazeroff, J. Gregor, M. Thomason, and R. Ford. Probabilistic suffix models for API sequence analysis of windows XP applications. *Pattern Recognition*, 41:90–101, 2008.

[7] D. Reynolds, T. Quatieri, and R. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 2000.

[8] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25:117–149, 1996.

[9] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann, 2000.

[10] H. Tsai, D. Yang, W. Peng, and M. Chen. Exploring group moving pattern for an energy-constrained object tracking sensor network. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2007.

[11] D. Tulone and S. Madden. PAQ: Time series forecasting for approximate query answering in sensor networks. In *The 3rd European Workshop on Wireless Sensor Networks*, 2006.

[12] A. Wang and V. Krishnamurthy. Signal interpretation of multifunction radars: Modeling and statistical signal processing with stochastic context free grammar. *IEEE Transactions on Signal Processing*, 56:1106–1119, 2008.

[13] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. Fidelity and yield in a volcano monitoring sensor network. In *The 7th symposium on Operating systems design and implementation*, 2006.