An Error-free Data Collection Method Exploiting Hierarchical Physical Models of Wireless Sensor Networks

Lei Fang
School of Computer Science
University of St Andrews, UK
lf28@st-andrews.ac.uk

Simon Dobson
School of Computer Science
University of St Andrews, UK
simon.dobson@st-andrews.ac.uk

Danny Hughes
Depart. of Computer Science
KU Leuven, Belgium
danny.hughes@cs.kuleuven.be

ABSTRACT

Various studies have shown that a substantial portion of the data gathered in real-world sensing applications is faulty. Most existing fault-detection approaches are off-line, centralised, and rely heavily on expert domain knowledge which may not always be available. The stochastic nature of physical phenomenon means that expert knowledge or historical models that reflect the physical world at some point may become stale later and give rise to a large rate of false alarms. We propose a data collection method with in-network, hierarchical, Demand-based, Adaptive Fault Detectors (DAFD). By applying a two-tiered error detection technique, the approach adapts itself to the changing physical environment. Preliminary real world implementation was done to show its feasibility for resource-restricted sensors. We demonstrate good detection accuracy in simulation while keeping the false alarm rate low.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

Keywords

Stationarity of sensor data, Physical model, Adaptive design, Robust learning

1. INTRODUCTION

Wireless Sensor Network (WSN) applications are attracting growing interests from both academia and industry. One issue preventing the commercialisation of WSN technology is the frequently low reliability of data gathered by sensors. It has been found that a substantial portion of sensor data is actually faulty [17]: 51% of the data collected in [19] was faulty; 3-60 % of data collected in the Great Duck Island experiment was incorrect [6]; and many other data series [16, 5] have been found to be faulty. It is almost impossible to filter out faulty readings manually for WSNs. Therefore, automatic data fault detection becomes an imperative for further adoption.

PE-WASUN'13, November 3–8, 2013, Barcelona, Spain.

ACM 978-1-4503-2361-1/13/11
http://dx.doi.org/10.1145/2507248.2507255 ...$15.00.

Data faults occur when a node performs a sensing task in an erroneous way, resulting in faulty data which deviates from the true value [6]. Most existing work [17] adopts a centralised and off-line approach. However, obvious disadvantages are associated with this sort of method. Firstly, a centralised solution requires the transmission of the whole data set alongside extra control information, which consumes a great deal of energy. Secondly, a centralised solution often suffers from scalability problems. Thirdly, an off-line approach cannot check the errors in the data as they are being collected, which could render a whole data set useless. An on-line, in-network, but lightweight, solution is therefore desirable for WSNs applications.

A problem facing most existing WSN fault detectors, however, is the high false-positive rates. Jayant [4] warns that existing solutions are not resilient to changes in the physical environment. They may mis-classify interesting events – for example the changes in humidity and temperature resulting from a volcanic eruption – as faults, thereby discarding the most interesting measurements. The roots of this problem lie in the validation of data by constructing a historic model and assuming this model remains true for all the future readings (also called the *stationarity assumption*).

In this paper, we propose a data collection method with in-network, hierarchical, Demand-based, Adaptive Fault Detectors (DAFD) embedded to improve the reliability of WSNs. The solution detect faults by exploiting two-tiered hierarchical physical models. A local tier detector preliminarily filters out potential errors by making use of local physical model, while the second tier validates the potential error by exploiting the physical spatial correlations between nodes. At the spatial level, two exchangeable data validation methods are available for node to autonomously pick up based on their running expenses.

The contribution of this work lies in the following aspects. The work models and exploits physical models, like intra-node multi-modal data correlation and spatial correlation, for sensor applications. Secondly, to reduce false alarms, the stationarity of sensor data series is studied and economical remedies are given when the stationarity assumption breaks down. Thirdly, the solution is robust to errors in the training data; therefore, the common but impractical error-free learning data assumption made by most existing works [17, 6] is relaxed.

Section 2 briefly reviews recent work. Section 3 presents our proposed solution, which we evaluate in section 4. We

conclude the paper in section 5 with some pointers to future work.

2. RELATED WORK

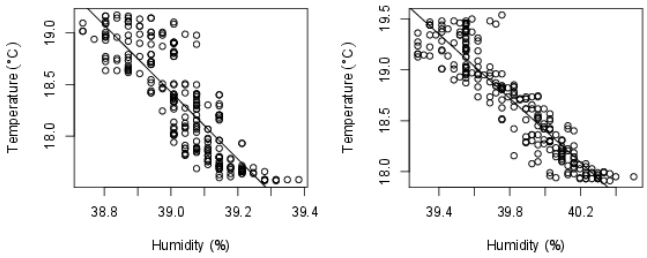
To the best of the authors' knowledge, DAFD is the first system that features multi-modal data validation as well as two-tiered fault detection exploiting both local and spatial correlation of sensor readings. In terms of exploiting correlation of multi-modal data at local level, Chu [1] creates optimised sensing schedules by modelling intra-node correlation between voltage and temperature. Instead of reporting temperature directly, each node reports a approximated temperature value calculated through voltage. Energy is saved because voltage is less expensive to retrieve comparing to a temperature sensor.

Ni [12] presents a detailed taxonomy of sensor data faults, and also presents a systematic approach to modelling these faults. However, no detailed detection method is proposed. Sharma [17] also studies data faults, as well as their possible causes. Four different data fault detection methods are compared: heuristic methods, estimation methods, time series analysis methods, and learning-based methods. The authors found that these four classes sit at different points on the detection accuracy spectrum: none are on-line detectors and none is adaptive to a changing physical environment. They depend heavily on domain/expert knowledge to set learning parameters beforehand rather than adjusting the learned model adaptively.

A packet-level attestation method to increase sensor data reliability is proposed by Kamal [6]. This work attaches an attestation bit to each observation to indicate its validity by exploiting one-hop spatial correlation. However, without a local filter, each observation – whether good or erroneous – will be sent to a neighbour for validation. Moreover, it ignores the possible breakdowns in the correlation between neighbouring sensor readings, which may result in wrong validations. Ni [11] proposes Hierarchical Bayesian Space-Time (HBST) modelling to find faulty data. Compared with linear autoregressive system, this work has a similar detection rate but lower false-positive rate. However, the system's computational complexity renders it inapplicable to resource-restricted sensors.

3. PROPOSED SOLUTION

In overview, the operation of DAFD consists of two phases, a learning phase and an operational phase. During the learning phase, statistical models of the data series are established in each node (local model) as well as between local nodes (spatial model). The local models make use of underlying physical relationships, for instance the linear correlation between temperature and humidity, at local level. The second tier model models the statistical distribution of spatial correlated data series. In the operational phase, each sampled data needs to go through the detectors at the local and, maybe further, the spatial level. Feedback from the second-tier test result will either be used to finalise the identity of faulty data or to update local model accordingly. Two validation methods at tier-two level are available to pick up by each node autonomously according to its the changing context so that extra communication is minimised. Appropriate remedial action can be taken when faulty data is found; for example, faulty data can be simply discarded locally or sent back with a flag.



(a) Node 1 (b) Node 2

Figure 1: Humidity versus Temperature with Regression Lines

3.1 Fault Detection: Tier-one (local) Model

3.1.1 Local Model

In reality, measurements of between different physical quantities are closely correlated, for example, temperature and humidity. In this paper, we harness this physical phenomenon by constructing a simple linear model between temperature and humidity, which is served as the local model:

$$T_i = \beta_0 + \beta_1 H_i + u_i \tag{1}$$

where T_i is the temperature; H_i is the humidity; u_i is the error term; and subscript i runs over all observations from 1 to n. We denote vector $\mathbf{T}^{'} = (T_1, T_2, ...)$ and $\mathbf{H}^{'} = (H_1, H_2, ...)$.

The model can be learnt by the ordinary least-squares (OLS) estimation. The OLS estimators have closed-form solutions, in matrix notation $\boldsymbol{\beta^*} = (\mathbf{X}^{'}\mathbf{X})^{-1}\mathbf{X}^{'}\mathbf{y}$, where $\mathbf{X} = (\mathbf{1}, \mathbf{H})$, $\mathbf{y} = \mathbf{T}$. Particularly, in this example, the estimators can be calculated according to Eqn. (2).

$$\begin{aligned} \hat{\beta_1} &= \frac{S_{HT}}{S_H} = \frac{\sum_{i=1}^{n}(H_i - \bar{H})(T_i - \bar{T})}{\sum_{i=1}^{n}(H_i - \bar{H})^2} \\ \hat{\beta_0} &= \bar{T} - \hat{\beta_1}\bar{H} \end{aligned} \tag{2}$$

The linear model is a data driven modelling approach; no prior knowledge about the system or environment is required. Models for two data series from Intel [5] are shown in Fig. 1. A clear negative linear correlation exist between temperature and humidity. Similar results are found in other data series.

According to the linear model, a prediction interval, which sets the boundary values for the value of interest, can be calculated formally at specific confidence interval:

$$\begin{aligned} T_{new} &\in \hat{\beta_0} + \hat{\beta_1} H_{new} \pm \varepsilon, \\ \varepsilon &= t_{n-2,\alpha}\hat{\sigma}(1 + \frac{1}{n} + \frac{(H_{new} - \bar{H})^2}{S_H})^{1/2} \end{aligned} \tag{3}$$

where $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(T_i - \hat{T_i})^2$ is the residual sum of squares; and $t_{n-2,\alpha}$ is the significance test coefficient obtained from a

t-table. When a new pair of observations, say (H_{new}, T_{new}), is sampled locally, the local prediction interval can be calculated according to Eqn. 3. Any data entry which is outside of the interval is marked as suspected data, and it will be checked by the spatial model for further validation. To reduce the validation workload for the local tier, we use user specified error band, $\hat{\varepsilon}$, instead of the regressor-specific error ε. Additionally, user-specified error band gives the flexibility to suit to different application specific requirements.

3.1.2 Data Efficient Local Learning

We introduce time-varying weights to the local learning method to make sure less data is required for local model construction. Since we are interested in applying the model to future data, the latter half of the learning data are more similar to the future series and therefore should be given a higher weight. A sigmoid function is used to give weights to each data entry in learning data set:

$$w_i(t) = \frac{1}{1 + e^{-k_s * (t-M)}} \tag{4}$$

In this equation M is the median epoch value of a learning data set, and k_s determines the shape of the sigmoid function. Data with epoch larger than M are given higher weights, and vice versa. As for the model, the model parameters can be estimated as

$$\boldsymbol{\beta}^* = (\mathbf{X}^{'}\mathbf{W}^{-1}\mathbf{X})^{-1}\mathbf{X}^{'}\mathbf{W}^{-1}\mathbf{y}, \tag{5}$$

where $\mathbf{W}^{'} = (w_1, w_2, ...)$.

3.1.3 Learning in Noisy Environments

To create reliable local models in noisy environments, we apply robust regression. The method assigns weights to each data entry according to its likelihood of being an outlier. To make the solution feasible for resource-constrained sensors, we employ a modified robust regression. We modified Huber's weight function, where k is the *tuning constant* whose value is set $2.0 \times \text{med}|e_i|$ $(i = 1, ..., n)$, med returns median value, and e is the residual [10].

$$w(e) = \begin{cases} 1, & \text{for } |e| \le k \\ k/|e|, & \text{for } k < |e| \le 2k \\ 0, & \text{for} |e| > 2k \end{cases} \tag{6}$$

The algorithm for robust learning is shown in Algorithm 2. Compared with traditional robust regression, we modified both weight function and iteration stopping criteria to make the learning process converge faster. The effect of robust regression is shown in section 4.1.4 .

3.2 Fault Detection: Tier-two (Spatial) Model

Spatial correlation is used to further check a suspect data entry reported by the tier-one model to make sure it is truly erroneous data rather than being a honest observation of a turbulent environment. Spatial correlation means data sampled by co-located sensors, which tend to observe the same phenomenon, is similar to each other. A initial learning phase is required to establish the spatial model. During the learning period, each node broadcasts its readings to its one-hop neighbours. We denote $\mathbf{d}_i$ as the local data source of node i, while data matrix $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n)$ is the learning data series from the source node's neighbours. We assume the data $(\mathbf{d}_i, \mathbf{S})$ is multivariate Gaussian distributed, which is a common assumption made in sensor data analysis [1], [3], [2]. The spatial modelling procedures are listed in Algorithm 1.

3.2.1 Verification nodes selection

The learning phase starts with the selection of verification node set for each source node. The reason for including this process is geometrically co-located sensors may not necessarily exhibit spatial correlation. For instance, when one of co-located nodes is in a separate dark-box enclosure, the readings reported may be quite different and independent from others. Pearson correlation coefficients, whose equation is shown as Eqn. (7), are used as the metric to decide verification nodes. The coefficient measures the strength of linear relationship between two data series, which is an indicator of data similarity [13].

We use a formal but lightweight statistical significance test to filter out irrelevant nodes. It can be proved that the sample correlation coefficient for N observations, when N is large $N > 25$, on two uncorrelated variable x and y is normally distributed with mean $\mu = 0$ and standard deviation $\sigma = 1/\sqrt{N-2}$ [13]. A simple one-tailed p-test at α-level then can be used to test whether the sample correlation is significantly larger than zero, indicating a strong positive correlation (see Algorithm. 1 for details). A primary verifier node with the largest sample correlation is then selected from the qualified node set. The primary verifier node is later used in single verifier validation method, see section 3.2.3.

$$r = \frac{\Sigma_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma_{i=1}^{n}(x_i - \bar{x})^2 \Sigma_{i=1}^{n}(y_i - \bar{y})^2}} \tag{7}$$

3.2.2 Group voting mechanism for spatial validation

During the learning phase, each node i also maintains a $2 \times |vrf(i)|$ spatial verification matrix, where $vrf(i)$ denotes the set of assigned verification nodes of i. The spatial matrix stores the estimators of expected absolute difference, $\hat{\mu}_{ij}^d$, between $\mathbf{d}_i$ and its verifiers $\mathbf{s}_j$ as well as the corresponding expected standard deviation, $\hat{\sigma}_{ij}^d$, for the absolute difference.

$$\hat{\mu}_{ij}^d = |med(\mathbf{d}_i) - med(\mathbf{s}_j)| \tag{8}$$

$$\hat{\sigma}_{ij}^d = 1.4826 \times MAD(\mathbf{d}_i - \mathbf{s}_j), \tag{9}$$

where med is the sample median, and MAD is the medium absolute deviation (MAD) of a data series, and 1.4826 is a scaling constant to adjust the MAD under normal assumption [8]. Note we use estimators, medium and MAD, instead of sample mean and standard deviation to reduce the effects of faulty learning data. The estimators are insensitive to outliers [8], which give good estimation even in noisy environments.

When node i receives a verification request from his neighbour, it will return a boolean result by consulting the matrix: if the on-request sample absolute difference is out of the boundary $[\hat{\mu}_{ij}^d - k_\sigma \hat{\sigma}_{ij}^d, \hat{\mu}_{ij}^d + k_\sigma \hat{\sigma}_{ij}^d]$, it is marked as faulty. We use $k_\sigma = 10$ in this work. Because according to Chebyshev's Inequality (Eqn. 10 [15]), when k_σ is 10, the probability of misclassification is only 1%. The proof of the theorem is beyond the scope of this article.

$$P(|X - \mu| \ge k\sigma) \le \frac{1}{k^2} \tag{10}$$

In terms of the data validation, the group voting method uses all the verification results from the source node's verifiers: if at least one of the verification results support the suspect data, the data is marked as non-faulty. The validation rule is stated as Eqn. (11), where $x_{i,t}$ denotes a data entry from sensor i at epoch t, and $bool_j(x_{i,t})$ is the verification result from node j.

$$faulty : \text{if } \vee_{j=1}^{vrf(i)} bool_j(x_{i,t}) == false \tag{11}$$

The reason we model the synchronized absolute difference, $\mathbf{z} = |\mathbf{d}_i - \mathbf{s}_j|$, to validate data is twofold. Firstly, $\mathbf{z}$ naturally reflects the discrepancy between two random variables. Secondly, and more importantly, after the difference, $\mathbf{z}$ usually becomes partially stationary or partially self-similar, i.e. the model learnt by historic data remains true for most future data series, which provides the legitimacy to use *group* verification without model update.

We use real world sensor data [5] to demonstrate the claims. Fig. 2 shows temperature data series from two correlated sensors. It is obvious that, comparing with the original data, the absolute difference is more self-similar. We use the first 150 data series as training data to learn our model parameters $\hat{\mu}_{ij}^d$, $\hat{\sigma}_{ij}^d$. It is found, with $k_\sigma = 10$, averagely 87% of all the future data series agree with the historic model. However, the model still breaks occasionally, rendering about 10-20% of future data rejected based on the historic model. To solve the problem, we use the group voting mechanism to share the risk of a breakdown in correlation. The rationale is while correlation between two specific nodes is likely to change, it is not likely that one node will be *totally* different from *all* its neighbours. By using group voting, the agreement rises from 87% to 99.9%.

3.2.3 Single verifier scenario for spatial validation

A different validation mechanism is needed when there exists only one neighbouring node qualified after the correlation test. As mentioned above, the single correlation model, which breaks from time to time, is not stable enough to serve as a validator alone. Therefore, the pair-wise spatial model needs to be updated periodically to adapt to the stochastic changes. To avoid the overhead of repeating the learning process, we only update the expected absolute difference, i.e. $\hat{\mu}_{ij}^d$. Note that only one parameter, the sample median $med(\mathbf{d_i})$ of the local learning data, needs to be sent for $\hat{\mu}_{ij}^d$ update. The variance, $\hat{\sigma}_{ij}^d$, is exempted from update because of its stationarity. The stationarity is valid from Fig. 2: the fluctuation pattern does not change over time. Formal analysis of the stationarity of $\hat{\sigma}_{ij}^d$ is presented in Appendix A.

The update frequency for $\hat{\mu}_{ij}^d$ is denoted as f_{update}. By updating $\hat{\mu}_{ij}^d$ with adequate frequency, say once every 50 epochs, the agreement arises from 87% to 98.2%.

3.2.4 Adaptive spatial validation method selection

The two different spatial validation methods incur different verification messages exchanges; and we give each node the freedom to choose one of the validation methods autonomously and adaptively so that smaller amount of validation messages need to be exchanged. The whole data collection period is divided into smaller slices p_t $(t = 1, 2, ...)$ with equal length of d_p. Each node decides its validation method, group or singular validation noted as G and S, at the beginning of every p_t. The number of message exchanges, n_{msg}, is based on the following parameters: the size of verifier set $|vrf|$, data collection window d_p, spatial verification demand v_d, and spatial model update frequency f_{update} for the singular verifier scenario. Among them, v_d, a stochastic variable measuring the number of verification requests issued in an epoch, needs to be monitored on-site. The variable depends largely on the condition of the hardware, the quality of local model and also the changing environment. The decision is made according to the rule stated in Eqn. (12). The derivation of the rule is listed in Appendix B. Note that the two methods can be exchanged easily by adding or removing corresponding verifiers as verification requests' recipients.

$$\text{validation method} = \begin{cases} G & 2 \times (|vrf| - 1)v_d \leq f_{update} \\ S & 2 \times (|vrf| - 1)v_d > f_{update} \end{cases} \tag{12}$$

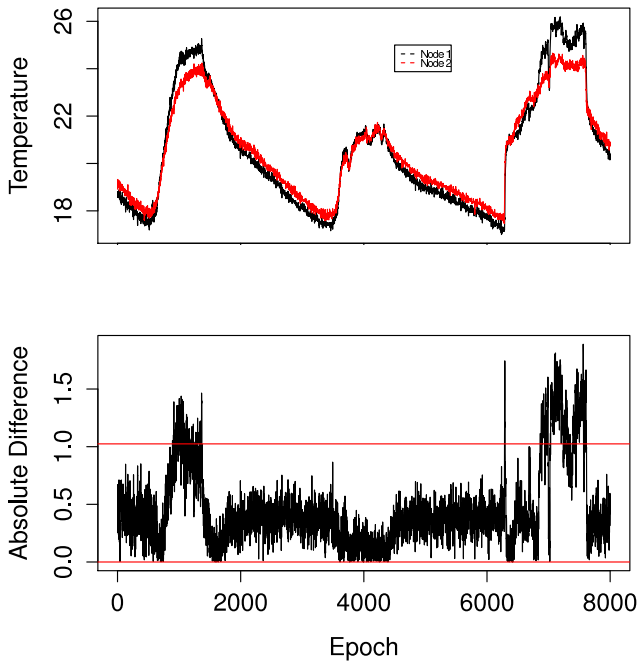


Figure 2: Stationarity of Real World Sensor Data. The top shows the temperature series from two co-located and correlated sensors; The bottom shows the absolute difference between the two series. The red lines are the boundaries learnt by the first 150 data entries.

3.3 Adaptive Local Model

The linear relationship between temperature and humidity usually changes as the physical world is changing. Therefore, updating the local model is essential. Instead of updating the local model at some pre-specified frequency – which could results in either infrequent or unnecessary update – we update the local model in an on-demand way by leveraging the feedbacks from the tier-two model.

When the feedback from spatial model is not faulty, it means the reported data, which is believed to be correct by its verifiers, does not agree with the current local model. In

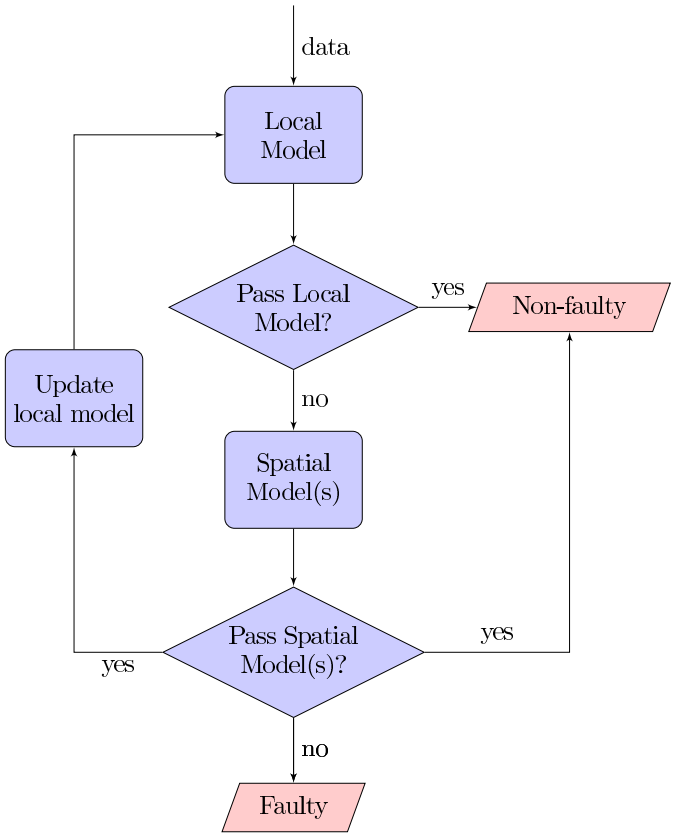


Figure 3: The detection procedure

this work we set an update threshold: when $n > thred_u$ consecutive false alarms from local model are witnessed, we force the node to learn a new local model. Fig. 3 show the detection procedure of the proposed solution.

4. EVALUATION

This section reports the results of an assessment of the performance of the proposed solution mainly in a numeric simulation. The solution is examined in various aspects by using a real world sensor data set. The solution has also been implemented. The footprint of the solution is reported.

4.1 Numerical simulation

To better understand the effectiveness of the solution, we use a real-world data set: the Intel Lab Data [5] to run numerical simulation. All the experimental results are obtained from simulations written in R [14] . The parameters used for the evaluation are listed in Table 1.

4.1.1 Fault Model

Injecting artificial faults into a real data set is a common approach to measuring the accuracy of a detector [6, 17, 4]. For our evaluation, four particular kinds of faults are considered: short, constant, noise and drift. Table 2 summarises the definitions, models and the parameters used for the different faults. The parameters are selected based on existing works [6, 17].

4.1.2 Detection Accuracy

Algorithm 1 Learning Spatial Model

Input: Neighbours Data Matrix: $\mathbf{S}$, LocalDataSeires: $\mathbf{d}_i$
Output: vrf_list, $spatial[][]$, vrf^*

initialise $spatial$ as a $|nbr()| \times 2$ matrix
$i = 0$
$r_{max} = 0$
$sigLevel = 1.96/\sqrt{length(\mathbf{d}_i) - 2}$
▷ one-tailed significance level at 2.5 level
for each s_j in S **do**
  calculate r_{d,s_j}
  if $r_{d,s_j} > sigLevel$ **then**
    $vrf\_list.add(NodeId(s_j))$
    if $r_{d,s_j} > r_{max}$ **then**
      $r_{max} = r_{d,s_j}$
      $vrf^* = NodeId(s_j)$
    end if
    ▷ calculate robust estimators according to Eqn. 8 and 9
    $spatial[i++][0] \leftarrow \hat{\mu}_{ij}^d$
    $spatial[i++][1] \leftarrow \hat{\sigma}_{ij}^d$
  end if
end for
truncate $spatial$ if needed

Algorithm 2 Robust Local Model Learning

Initialise estimates $\mathbf{b}^{(0)}$ as ordinary least square estimates
while t $<$ $thred_r$ & no convergence **do**
  calculate residuals $e_i^{(t-1)}$ and associated weights $w_i^{(t-1)} = w(e_i^{(t-1)})$
  Solve $\boldsymbol{b^{(t)}} = [\mathbf{X}'\mathbf{W}^{(\mathbf{t-1})}\mathbf{X}]^{-1}\mathbf{X}'\mathbf{W}^{\mathbf{t-1}}\mathbf{y}$
end while

Table 1: A Summary of Key Parameters

Name	Value Used	Description (Unit)
β^*	NA	Local model parameters
ε	1	Error band for local validation;(℃)
k_s	0.133	Parameter for sigmoid function, see Eq. 4
k_σ	10	Number of deviation from mean, used for spatial validation
L_{size}	150	Learning data size
$thred_u$	20	Update threshold for local model update
$thred_r$	4	Stopping criteria for robust model learning, see Algorithm. 2
d_p	100	Spatial method selection monitoring window;(epoch)
v_d	NA	Spatial verification request demand monitored; ($^{\text{pieces}}/_{\text{epoch}}$)
f_{update}	$\frac{1}{50}$	Update frequency for singular verifier; ($^{\text{times}}/_{\text{epoch}}$)

Table 2: Different Fault Models

Class	Definition	Model	Parameters
SHORT	Momentary deviation from the true reading	$S_s(x,t) = g(x,t) + f * g(x,t)$	random f, i.e. fault intensity, from $[0.1, 10]$ is assigned
CONSTANT	Sensor readings remain constant for an unexpected period	$S_c(x,t) = c$, where $t \in T$	random c from $[33, 999]$ is chosen
NOISE	Sensor readings exhibit large unexpected variation	$S_n(x,t) = g(x,t) + N(0, \sigma^2)$, where $t \in T$	random σ from $[3, 10]$ is assigned
DRIFT	Sensor readings deviate from true values by a time-varying offset	$S_d(x,t) = g(x,t) + f(t)$, where $t \in T$ and $f(t) = a^t$	random a from $[2, e]$ is assigned

Table 4: Comparing DAFD with Heuristic Methods

	True P.	False P.
DAFD	1.0	0.012
Heuristic (clean learning set)	0.99	0.74
Heuristic	0.16	0.427

The faults detected mainly can be categorised into the following four classes: data points correctly detected as faulty (true positive); data points correctly detected as non-faulty (true negatives); data points incorrectly detected as faulty (false positives); and data points incorrectly detected as non-faulty (false negatives). We compare both true positive rate and false positive rate, which provides a complete picture of the solution.

Three simulation scenarios are considered:

1. For each node, only inject errors in one category of sensor readings (for example, in temperature only)

2. For each node, inject errors in both categories of readings (temperature and humidity)

3. Inject errors in both a node and its neighbours.

Table 3 shows both true positive and false positive rates (listed in brackets). It is clear that DAFD achieves almost 100% accuracy under scenario one for all the faults except NOISE. DAFD achieves high accuracy even when errors may be present in learning data, which means the proposed solution is resilient to errors and can perform well in an noisy environment.

To better see the results of DAFD, we compare DAFD with a heuristic (rule-based) method presented in [4] when random SHORT faults are injected. The threshold for the heuristic rule is selected by a learning process involving the first 200 data entries. The results are shown in Table 4. Note that we assume the learning data for the heuristic method is error-free for the second result. It is clear that DAFD outperforms both the heuristic methods, especially in terms of false positives. It is also worth noting that the heuristic method is not reliable when errors are present in the learning data, which is a possibility that cannot typically be ruled out in real-world scenarios.

4.1.3 Data Usage Efficiency of DAFD

For methods involving a learning phase, one may also concern the efficiency of data utilisation. In other words, the learning set should be small so that more data can be validated. By applying time-varying weighted regression (see 3.1.2), it can be shown that the solution can achieve the same result while using a smaller total training data set, which is more efficient. Table 5 shows that DAFD for different nodes on average uses 930 data entries in total as learning data during the whole process. This saves about 60% of learning data compared to the method without time-varying weights.

Table 5: Data Usage Efficiency of DAFD

	DAFD	DAFD-NoWeighted
Total Learning Data Size	930	1552

One should also note that model adaptation is necessary and should not be omitted, which means that using learning data is justifiable. Without the local model update, a stale local model will produce an excessive amount of suspect faults and result in a large volume of validation requests. Results show that over three times (3.24) the number of suspect faults are generated in the local tier when a static model is employed, which means that energy efficiency is similarly reduced.

4.1.4 Learning in Noisy Environments

DAFD uses robust learning methods to downplay the effects of faults in learning data. Fig. 4 compares the accuracy of DAFD with and without robust learning. It can be seen from the figure that robust regression is resilient to errors. It automatically ignores those faulty data entries when learning a local model as well as the spatial model. Even for faults of high intensity it can still produce accurate models. It is interesting to note that when fault intensity gets smaller the two methods converge, as faulty data converge to the real data set.

4.1.5 Adaptive spatial validation method selection

We present the result that a local node autonomously select its validation method adaptively along the process in Fig. 5. As evident from the figure, the node chooses the two methods alternatively and the selection usually starts with a cluster of singular methods and followed by a longer series of group methods. The clustering feature means the hardware condition or local model is stable locally; therefore, the most recent monitored variable v_d can be used to predict one-step further situation. Similar results were found among other nodes. In terms of messages exchanges, the adaptive solution reduces 49% and 5% of the verification messages

Table 3: Simulation Results of True Positive and False Positive

	SHORT	CONSTANT	NOISE	DRIFT
CASE 1	1.0 (0.012)	1.0 (0.013)	0.96 (0.012)	1.0 (0.012)
CASE 2	1.0 (0.012)	1.0 (0.013)	0.928 (0.01)	0.997 (0.012)
CASE 3	0.99 (0.014)	1.0 (0.014)	0.926 (0.012)	0.989 (0.014)

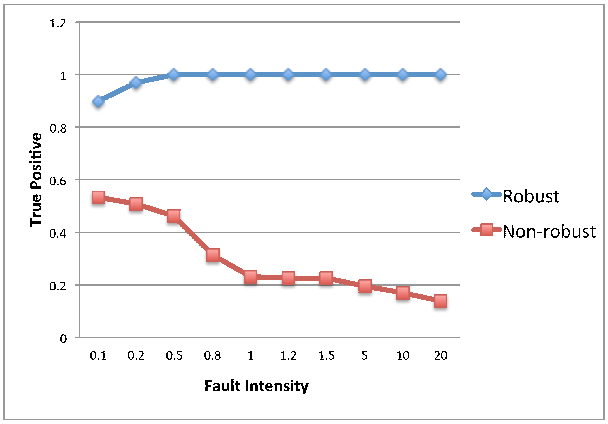


Figure 4: The effects of robust model learning

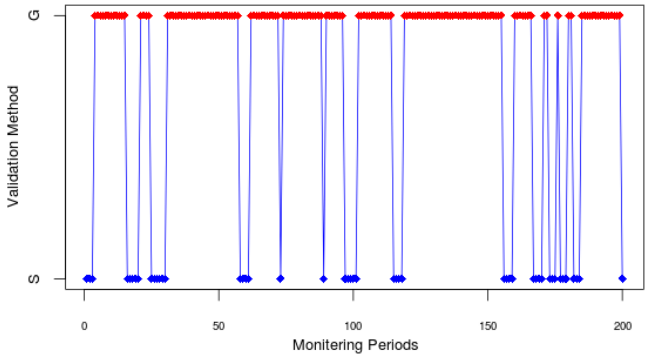


Figure 5: Adaptive validation method selection

comparing to pure group method and singular method respectively.

4.2 Implementation

We have evaluated DAFD using IEEE 802.15.4 complaint T-mote Sky mote. It consists of an processor running at maximum 8MHz and RAM of 10 KB [9]. The relative small RAM size becomes a major hindrance for the system and application program. We implemented the solution in nesC that runs on TinyOS 2.1.0 [7]. For the entire learning and operating process, the solution uses a number of local array of floating point numbers to temporarily hold the learning data and learnt model parameters. The array size mainly depends on L_{size}. We observe that, with $L_{size} = 150$, the footprint of RAM in average is only 5100 bytes, which is only 51% of the total available memory for a typical T-mote Sky node.

5. CONCLUSION AND FUTURE WORK

In this paper we have proposed a two-tiered data validation framework for WSN data collection application. The proposed solution maintains high rates of faulty-data detection and accuracy. By exploiting spatial correlation, we also manage to reduce the false alarm rate. To make the solution adaptive to the changing physical phenomenon, we form a feedback loop to make the local models update in response to changing environments. Early experimental results seem promising, and we believe that the use of structured statistical methods has the potential to dramatically improve the quality of data collected.

Some parts of the solution still need further investigation. For instance, due to time limits, we only implemented the core part of the solution and no comprehensive evaluation on the real world deployment has been carried out. We plan to finish the evaluation by checking the overheads of the proposed solution as well as its scalability. Moreover, currently we use fixed periodical update to adjust spatial model for the singular verifier scenario. However, we believe a better solution would be on-demand update. Additionally, some of the parameters, like k_σ, are quite sensitive. We currently use a global fixed value for the whole deployment. An open research question is how to select the parameters according to the specific context. Other statistical methods may also be investigated and compared. Another interesting extension would be the integration of efficient data collection with data validation. We intend to investigate how to integrate the validation framework into an efficient data acquisition method so that overall overheads are minimised by using statistical methods.

Acknowledgements

Lei Fang is supported by a studentship from the Scottish Informatics and Computer Science Alliance (SICSA).

APPENDIX

A. THE STATIONARITY OF $\hat{\sigma}$

We analyse the stationarity of $\hat{\sigma}^d_{ij}$ by firstly carrying out a formal statistical test and then a error tolerance analysis.

We apply Augmented Dickey-Fuller (ADF) test, a widely used test for examining the stationarity of a time series [18], to different variance series with a sliding window size of 150. The variance series is a time series with each entry being the variance of a moving window of data. The test results show that the $\hat{\sigma}^d_{ij}$ series is strongly stationary (the hypothesis cannot be rejected at 99.99% with average Dickey-Fuller coefficient: -8.24).

Secondly, if an error tolerance band, ϵ, is known, i.e. any measurement within this error band is acceptable for this application, we find that most of the future $\hat{\sigma}^d_{ij}$ reside within a $\frac{\epsilon}{k_\sigma}$ band of the first sample variance. For example, 100%

of the test variances is within 0.25 band of the first variance learnt from the first 150 data when $\epsilon = 1$ and $k_\sigma = 10$. The result implies that, even without variance update, 100% of the measurements still can meet the measurement error requirement; i.e. the temperature only deviates 1 Celsius degree.

B. DERIVATION OF THE RULE

Group Verifiers

For method G, each spatial verification requires $|vrf| + |vrf|$ of message exchanges. Therefore, the number of message exchanges expected for the next monitoring period p_t (of duration d_p) based on the most recent observed verification demand v_d is

$$n^G_{msg} = (|vrf| + |vrf|) \times d_p \times v_d;$$

Singular Verifier

While for method S, the counterpart is

$$n^S_{msg} = (1 + 1) \times d_p \times v_d + d_p \times f_{update}.$$

Note method S requires periodical update which incurs extra $d_p \times f_{update}$ message exchanges.

By differencing, $n^G_{msg} - n^S_{msg}$, the rule (Eqn. 12) can be obtained. The method with lower message exchanges is preferred.

C. REFERENCES

[1] D. Chu, A. Deshpande, J. Hellerstein, and W. Hong. Approximate data collection in sensor networks using probabilistic models. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*, pages 48–48, 2006.

[2] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, VLDB '04, pages 588–599. VLDB Endowment, 2004.

[3] B. Gedik, L. Liu, and P. Yu. Asap: An adaptive sampling approach to data collection in sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 18(12):1766–1783, 2007.

[4] J. Gupchup, A. Sharma, A. Terzis, A. Burns, and A. Szalay. The perils of detecting measurement faults in environmental monitoring networks. In *Proceedings of DCOSS*, 2008.

[5] INTEL. Intel lab sensor dataset 2004. http://db.csail.mit.edu/labdata/labdata.html, 2004.

[6] A. R. M. Kamal, C. Bleakley, and S. Dobson. Packet-level attestation (pla): A framework for in-network sensor data reliability. *ACM Trans. Sen. Netw.*, 9(2):19:1–19:28, Apr. 2013.

[7] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. TinyOS: An Operating System for Sensor Networks. In W. Weber, J. Rabaey, and E. Aarts, editors, *Ambient Intelligence*, chapter 7, pages 115–148. Springer Berlin Heidelberg, Berlin/Heidelberg, 2005.

[8] R. A. Maronna, R. D. Martin, and V. J. Yohai. *Robust statistics*. J. Wiley, 2006.

[9] Moteiv. *Tmote Sky Datasheet http://www.sentilla.com/pdf/eol/tmote-sky-datasheet.pdf*, 2006.

[10] R. Myers. *Classical and modern regression with applications*, volume 2. Duxbury Press Belmont, CA, 1990.

[11] K. Ni and G. J. Pottie. Sensor network data fault detection with maximum a posteriori selection and bayesian modeling. *ACM Transations on Sensor Networks*, 8(3), 2012.

[12] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, E. Kohler, G. Pottie, M. Hansen, and M. Srivastava. Sensor network data fault types. *ACM Transactions on Sensor Networks*, 5(3):25:1–25:29, 2009.

[13] H. A. Panofsky and G. W. Brier. *Some applications of statistics to meteorology*. Mineral Industries Extension Services, College of Mineral Industries, Pennsylvania State University, 1958.

[14] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. ISBN 3-900051-07-0.

[15] S. M. Ross. *Introduction to probability models*. Academic Press, 2006.

[16] SensorScope. EPFL SensorScope Project. http://sensorscope.epfl.ch, 2008.

[17] A. Sharma, L. Golubchik, and R. Govindan. Sensor faults: Detection methods and prevalence in real-world datasets. *ACM Transactions on Sensor Networks*, 6(3):23, 2010.

[18] R. H. Shumway and D. S. Stoffer. *Time series analysis and its applications*. Springer Science+ Business Media, 2010.

[19] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, et al. A macroscope in the redwoods. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 51–63, 2005.