Robust Event Boundary Detection in Sensor Networks — A Mixture Model Based Approach

Min Ding
Department of Computer Science
The George Washington University
Washington DC 20052, USA
Email: minding@gwu.edu

Xiuzhen Cheng
Department of Computer Science
The George Washington University
Washington DC 20052, USA
Email: cheng@gwu.edu

***Abstract*—Detecting event frontline or boundary sensors in a complex sensor network environment is one of the critical problems for sensor network applications. In this paper, we propose a novel algorithm for event frontline sensor detection based on statistical mixture methods with model selection [1], [11], [5]. A *Boundary* sensor is considered as being associated with a *multimodal* local neighborhood of (univariate or multivariate) sensing readings, and each *Non-Boundary* sensor is treated as being with a *unimodal* sensor reading neighborhood. Furthermore, the set of sensor readings within each sensor's spatial neighborhood is formulated using *Gaussian Mixture Model* [9], [5]. Two classes of *Boundary* and *Non-Boundary* sensors can be effectively classified using the model selection techniques for finite mixture models. Our extensive experimental results demonstrate that our algorithm effectively detects the event boundary with a high accuracy under moderate noise levels.**

***Index Terms*—Event boundary/frontline detection; Gaussian Mixture Model; Expectation-Maximimation**

I. Introduction

Sensor networking and methods on information processing of sensoring data have emerged as an important area of many real world applications and research activities. In general, information processing in sensor networks is facing three main challenges: (i) the scale of a sensor network can be tremendous, (ii) sensor readings are prone to noise in real environments and unreliable inter-sensor communications can cause further information loss, and (iii) each sensor has very limited computational power, memory storage and energy supply. These challenges motivate us to design robust, efficient, distributed and in-network information extraction algorithms.

In this paper, we propose a new, more accurate and robust, statistic based algorithm for detecting event *Boundary* (or frontline) sensors then previous work [4], [2], [7], [6]. *Median* based approach [4] works with scalar sensor inputs and only handles single channel information such as temperatures, or humidities over a geographical region. [2], [7] only take 0/1 binary predicates. Thus it remains to be a problem of how to process multivariate data resources of multi-modality sensor readings for event reporting. As a contrast, our new statistical *Gaussian* mixture model [5] based method in this paper is capable of fusing multivariate real-valued sensor inputs to detect boundaries of events, in a mathematically principled manner. Our new methods also naturally work for detecting boundaries of multiple event intersections, without constraining simplistic individual event shapes[1].

Our algorithm of event frontline detection is derived from statistical *Gaussian* mixture models [5] with explicit model selection schemes [1], [11]. The basic idea is based on the observation that a *Boundary* sensor is considered as residing within a local sensor neighborhood with a *multimodal* distribution of (univariate or multivariate) reading inputs, while each *Non-Boundary* sensor is surrounded with a neighborhood of *unimodal* sensor readings. More precisely, the distribution of sensor readings within each sensor's spatial neighborhood is mathematically formulated using *finite Gaussian Mixture Model* [9]. The model selection techniques [1], [11], [5] can then effectively identify the correct number of modes Γ for finite mixture models, of which *Gaussian* mixture model is the most popular. Thus *Boundary* and *Non-Boundary* sensors can be consequently distinguished from its neighboring sensor data distributions of $\Gamma > 1$ or $\Gamma = 1$.

Extensive simulation results demonstrate that our proposed algorithm can accurately detect the event *boundary* sensors with high robustness regarding to various noise levels and different experimental settings. Though designed for sensor networks, our algorithm can be applied to general regional data analysis in spatial data mining [10] and network traffic mining [8].

II. Related Work

[2] proposes three different schemes as statistical approach, image processing approach and classifier-based approach, all of which can only take inputs of the 0/1 decision predicates from neighboring sensors. [6] presents a noise-tolerant algorithm named *NED* for event and event boundary detection. In *NED*, the moving mean of the readings of the neighboring node set is used as the estimate for a certain sensor node. [4] propose *Median*-based approaches for outlying classification and event frontline detection. *Median* is a useful robust estimator which works directly with continuous numbers, rather than binary 0/1 readings. All previous works discussed above are designed for sensor networks in single modality. To the best of our knowledge, this paper is the first work that presents

[1]In [4], methods are designed to detect only simplified ellipse curves or straight lines as event boundaries/frontlines for the validity of their heuristic based random bi-section or tri-section *Median* schemes.

a principled methodology integrating multimodality sensor readings.

III. Motivation and Mixture Models

The mixture model has been widely used in many areas [10], [8]. Our major contribution is exploring the statistical property of the finite mixture model [3], [5], especially the Gaussian mixture model, and adopting it into the scenario of distributive (sensor network) sensing data process and mining.

A. Finite Mixture Models and Gaussian Mixture Models

Given a collection of data samples $\mathbb{X} = \{x_1, x_2, ..., x_m\}$ with each x_i representing a D-dimensional random (column) vector, assume that $\mathbb{X}$ follows a k-component (or mode) finite mixture distribution as

$$p(x_i \mid \theta) = \sum_{j=1}^{k} \alpha_j p(x_i \mid \theta_j), j = 1, 2, .., k; i = 1, 2, .., m \qquad \text{subject to} : \sum_{j=1}^{k} \alpha_j = 1 \tag{1}$$

where α_j is the mixing weight and θ_j is the set of parameters of the j-th mixture component $p(x_i \mid \theta_j)$. We denote $\theta = \{\alpha_1, \theta_1, \alpha_2, \theta_2, ..., \alpha_k, \theta_k\}$ as the complete set of parameters defining a specific finite mixture model. The objective function of estimating θ from $\mathbb{X}$ is to maximize the log-likelihood criterion

$$logp(\mathbb{X} \mid \theta) = \sum_{i=1}^{m} log \sum_{j=1}^{k} \alpha_j p(x_i \mid \theta_j). \tag{2}$$

Then the maximum likelihood estimator of θ is

$$\hat{\theta}_{ML} = \arg\max_{\theta} \{logp(\mathbb{X} \mid \theta)\}. \tag{3}$$

It is well known [9], [5], [3] that $\hat{\theta}_{ML}$ can not be computed analytically from equation 3. Instead, *expectation maximization* (*EM*) algorithm [3] is applied as its general solver to iteratively find the maximum likelihood solution $\hat{\theta}_{ML}$.

The *Gaussian mixture model* (GMM) is the most important class of *finite mixture densities* [5]. *GMM* is formulated by using a *Gaussian* density $\mathcal{G}(x_i \mid \mu_j, \Sigma_j)$ with its mean vector μ_j and covariance matrix Σ_j to replace the general probability density function $p(x_i \mid \theta_j)$ in the finite mixture model

$$p(x_i \mid \theta) = \sum_{j=1}^{k} \alpha_j \mathcal{G}(x_i \mid \mu_j, \Sigma_j), \tag{4}$$

where a D-dimensional multivariate *Gaussian* distribution is defined as

$$\mathcal{G}(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu) \right\}.$$

B. Expectation Maximization Algorithm

EM algorithm is an iterative method on estimating the free parameters of finite/Gaussian mixture models, more specifically the mean vectors μ_j, covariance matrices Σ_j and prior weights α_j in *GMMs*. The EM algorithm for *GMM* includes iterations of E-step and M-step as follows.

E-step: Estimate the posterior probability γ_{ij}^t that the j-th component generated x_i using the estimates of the parameters from the last M-step

$$\gamma_{ij}^t = \frac{\alpha_j^{t-1} \mathcal{G}(x_i \mid \mu_j^{t-1}, \Sigma_j^{t-1})}{\sum_{j=1}^{k} \alpha_j^{t-1} \mathcal{G}(x_i \mid \mu_j^{t-1}, \Sigma_j^{t-1})} \tag{5}$$

M-step: Compute the parameters using γ_{ij}^t

$$\alpha_j^t = \frac{\sum_{i=1}^{m} \gamma_{ij}^t}{\sum_{j=1}^{k} \sum_{i=1}^{m} \gamma_{ij}^t} \tag{6}$$

$$\mu_j^t = \frac{\sum_{i=1}^{m} \gamma_{ij}^t x_i}{\sum_{i=1}^{m} \gamma_{ij}^t} \tag{7}$$

$$\Sigma_j^t = \frac{\sum_{i=1}^{m} \gamma_{ij}^t (x_i - \mu_j^t)(x_i - \mu_j^t)'}{\sum_{i=1}^{m} \gamma_{ij}^t}. \tag{8}$$

The *EM* algorithm generates a sequence of model parameter estimates $\hat{\theta}_{ML}^t, t = 1, 2, ..$, until convergence, by alternatively executing *E-step* and *M-step* from an initialization of θ^0. The most common initialization is to set $\alpha_j = 1/k$, $\Sigma_j = I(D)$ where $I(D)$ is a D-dimensional identity matrix, and $\mu_j = random(\{x_1, x_2, \cdots, x_m\})$ for any mixture model component j. Convergence of the *EM* algorithm is guaranteed since the objective function is monotonically increasing for each incremental iteration and the global optimum is bounded [9].

C. Model Selection

The *EM* algorithm does not provide any information regarding selection of the number of mixtures from data. Clearly, such a selection is an important and unavoidable computational issue for *GMM* and *EM*.

Akaike's information criterion (AIC) [1] and Bayesian information criterion (BIC) [11] are the two most popular model selection criteria based on penalty terms of model complexity. *AIC* and *BIC* intend to compute the model that best represents the data (i.e., data likelihood term), with a minimum number of free parameters to be estimated (i.e., model complexity term). Nevertheless, *AIC* has the tendency of obtaining an estimated model with a negative bias towards the true data distribution when the data sample number m is in the same magnitude as $\mathbb{K}$. Here $\mathbb{K}$ is the total number of parameters to be estimated in *GMM*. Thus in this paper, we use the following *BIC* [11]

$$BIC(\theta) = -2log(p(\mathbb{X}|\theta)) + \mathbb{K}log(m) \tag{9}$$

for *GMM* model selection. To use *BIC* for model selection, we simply choose the model that leads to the smallest *BIC* over the set of possible models.

IV. Algorithm for Event Boundary Sensor Classification

In this paper, we provide a solution for classifying event boundary sensors using *EM* based model fitting and model selection techniques.

A. Algorithm

Given a sensor network $\{S_i\}$, we assume that sensors are moderately dense-deployed in the spatial domain and the spatial distribution of sensing signals is smooth within each event region. From a mathematical perspective, sensor readings provide a dense, but discrete samplings of the underlying continuous distribution. Furthermore, x_i, the reading of S_i, is considered as a D-dimensional random vector[2].

To detect or classify the event boundary sensors, the associated neighborhood sensor set $\mathcal{N}(S_i)$ of each sensor S_i needs to be first constructed as follows,

$$\begin{aligned} \mathcal{N}(S_i) = \{S_n\} \\ \text{subject to}: \ dist(S_i, S_n) <= D \end{aligned} \tag{10}$$

where $dist(S_i, S_n)$ represents either the Euclidean distance or one-hop, or multi-hop distance between sensor S_i and S_n, and D is the distance constraint.

To check whether or not S_i is a sensor lying on the boundary of an event, we input the data $\{x_n\}$ from readings of the sensors in $\mathcal{N}(S_i)$ and then build our best *GMM* based on $\{x_n\}$. Here the *EM* algorithm in Section III-B is applied for parameter estimation, and the *BIC* in Section III-C is used to select the final model. More details on this follow below. We first set the upper bound of the mixture component number to be K. Then for each $J = 1, 2, ..., \mathrm{K}$, the data set $\{x_n\}$ is feeded into the *EM* algorithm for estimation of $\theta(J)$. Correspondingly, we obtain *BIC* scores $\{BIC(\theta(J))\}_{J=1,2,..,\mathrm{K}}$ in light of equation 9. Let CL denote the number of mixture components of our final model (the best model). We select $CL = k$ where $BIC(\theta(k)) = \min_{J=1}^{\mathrm{K}} BIC(\theta(J))$. Therefore our final is $\theta(CL)$, or $\{\mu_j, \Sigma_j, \alpha_j\}_{j=1,...,CL}$. Optionally, a sensitivity test can be performed by removing from $\theta(CL)$ any under-presented mixture components with weights $\alpha_j < \zeta_1$. Then we obtain an updated model $\theta'(CL)$ with $CL = CL - k_0$ where k_0 is the number of removed components from the original $\theta(CL)$. Now it is straightforward to check the status of S_i. S_i is treated as an event boundary sensor if and only if $CL >= 2$. To classify if S_i is an outlying sensor, the conditional probability for x_i given model $\theta'(CL)$ is computed

$$p(x_i|\theta'(CL)) = \sum_{j=1}^{CL} \mathrm{w}_j \mathcal{G}(x_i \mid \mu_j, \Sigma_j) \tag{11}$$

then if $p(x_i|\theta'(CL)) < \zeta_2$, S_i is classified as an outlying sensor and $\mathcal{F}(S_i) = 1$ is set; otherwise $\mathcal{F}(S_i) = 0$. Note that ζ_1 is a threshold indicating an insignificant value of overall mixture component weight, and ζ_2 is used as another threshold to measure outlying sensors which have significantly low probalility density values given the final model $\theta'(CL)$. The above is summarized in algorithm 1. Our algorithm is purely distributed and computed for each sensor data neighborhood set $\{x_n\}_{n=1,2,...m}$ of $\mathcal{N}(S_i)$ at S_i or x_i. The complexity of the *EM* algorithm has the same order of magnitude as the *expected complexity* of *Median* based algorithms in [4].

V. Experiments

In this section, we describe our experimental settings and report our evaluation results on the task of *Boundary/Non-Boundary* sensor classification.

[2]In our approach, the pairwise correlations among multi-modality sensor readings, such as temperature, humidity and others, are naturally formulated by the covariance matrices of finite/*Gaussian* mixture models.

Algorithm 1 *Event Boundary Sensor Classification Based on GMM.*

inputs: Multivariate sensor readings $\{x_n\}$ of sensors from the neighborhood $\mathcal{N}(S_i)$.
outputs: Labels $\mathcal{E}(S_i)$, $\mathcal{F}(S_i)$ of sensor S_i. If S_i is an event boundary sensor, set $\mathcal{E}(S_i) = 1$; otherwise $\mathcal{E}(S_i) = 0$. If S_i is an outlying sensor, set $\mathcal{F}(S_i) = 1$; otherwise $\mathcal{F}(S_i) = 0$.

1) Input $\{x_n\}$ to build *GMM* using *EM* algorithm (equations 4, 5, 6, 7 and 8) for parameter estimation and *BIC* (equation 9) for model selection.
2) *Mixture Component Number Determination:* Set the upper bound of the mixture component number to be K. From the available *GMMs* $\{\theta(J)\}_{J=1,2,..,\mathrm{K}} = \{\{\mu_j, \Sigma_j, \alpha_j\}_{j=1,..,J}\}_{J=1,2,..,\mathrm{K}}$ and their associated *BIC* scores $\{BIC(\theta(J))\}_{J=1,2,..,\mathrm{K}}$, we select $CL = k$ where $BIC(\theta(k)) = \min_{J=1}^{\mathrm{K}} BIC(\theta(J))$. Our final model is then $\theta(CL) = \{\mu_j, \Sigma_j, \alpha_j\}_{j=1,...,CL}$.
3) *Sensitivity Testing (optional):* Denote by k_0 the number of mixture components with $\alpha_j < \zeta_1$. Removing these k_0 under-presented mixture components from $\theta(CL)$ yields the new model $\theta'(CL)$ with $CL = CL - k_0$.
4) *Classify Boundary Sensor:* If $CL >= 2$, S_i is an event boundary sensor and set $\mathcal{E}(S_i) = 1$; otherwise $\mathcal{E}(S_i) = 0$.
5) *Classify Outlying Sensor by density value:* Compute the conditional probability for x_i of sensor S_i: $p(x_i|\theta'(CL))$ using equation 11. Then if $p(x_i|\theta'(CL)) < \zeta_2$, S_i is an outlying sensor and set $\mathcal{F}(S_i) = 1$; otherwise $\mathcal{F}(S_i) = 0$.

A. Experiment Setup

Our experiments are performed within a map consisting of 128×128 simulated sensors with each randomly deployed in one of the 128×128 grids, restricting one sensor per grid. By default, our experiments are performed on sensors with three sensing channels. The extension to multivariate variables with dimension > 3 is straightforward.

In our **Matlab**-based experiments, we rigorously test the following signal models under different *Normal* distributions. The background sensing function is set as *Gaussian* noise $\mathcal{G}(\mu_0, \Sigma_0)$, where $\mu_0 = randn(3,1) \times 8 + [25, 25, 25]'$ and Σ_0 is a symmetric, positive semi-definite matrix with diagonal elements as σ and other elements as $randn(1,1) \times \sigma \times 0.3$ under $\Sigma_0(i,j) = \Sigma_0(j,i)$. Function $randn(m,n)$ returns a $m \times n$ matrix, of which each matrix element satisfies the standard *Normal* distribution $\in \mathcal{N}(0,1)$. Diagonal elements of the covariance matrix represent self-correlation and off-diagonal elements indicate cross-correlation between pairwise multi-variate variables. In the experiments, we set $\sigma = 5$ by default. For synthesized sensor events, *Normal* distributions are also assumed. For a particular example of the ith event $\mathcal{G}(\mu_i, \Sigma_i)$, each variable in μ_i is randomly sampled from a 10-component vector $[25, 50, 75, 100, 125, 150, 175, 200, 225, 250]$ plus an additive *Gaussian* disturbance $\in \mathcal{N}(0,8)$, to preserve the separatability among different sensing events including the background noise. Σ_i is kept as the same of Σ_0 (ie.,$\Sigma_i = \Sigma_0$) for simplicity.

There are three different shapes and five total configurations in our experiments. We randomly generate q ellipses or rectangle bars at arbitrary orientations, or $q + 1$ radially divided zones in a star-graph in the simulated sensor map with q event regions. In each event area i, sensor values are sampled from a specific multivariate *Gaussian* distribution $\mathcal{G}(\mu_i, \Sigma_i)$ as discussed above. The lengths of ellipse axes are sampled uniformly from $[5, 20]$ independently. The center and orientation of each ellipse are randomly placed. For rectangle bar, we first produce a random line with length > 20 and then label all sensors as event sensors which are within a certain distance (randomly from $[5, 10]$) from the line. Due to the spatial randomness, an event i may intersect with another event j. In this case, the simulated sensor readings of an overlapping area can be either selected from one of the two distributions according to an arbitrary order (i.e., one event distribution is overwritten by the other), or as the overlaid sum[3]. These two different ways of handling sensor readings in an overlapping region are denoted as EL_OW or EL_OL for the ellipse event shapes, and LB_OW or LB_OL for the event shape of rectangle bars, respectively. The star-graph originates at a random point in the center 32×32 region of the grid map, and its zones span some random angles from the range of $2\pi/(q+1) \pm \pi/6$. Unless otherwise stated, we set $q = 3$ for experimental evaluations.

To evaluate the performance of algorithm 1 for boundary sensor classification, we use four classification accuracy metrics: event boundary rate (EB Rate), error rate (ER Rate), true positive rate (TP Rate), and false positive rate (FP Rate). EB Rate is the ratio of the number of *Boundary* sensors to the number of *all* sensors; ER Rate is the ratio of the number of *incorrectly* classified sensors to the number of *all* sensors; TP Rate is the ratio of the number of *correctly* classified *Boundary* sensors to the number of *Boundary* sensors; and FP Rate is the ratio of the number of *incorrectly* classified *Boundary* sensors to the number of *Non-Boundary* sensors. Furthermore, the outlying detection rate (OR Rate) is used to evaluate the accuracy of outlying sensor detection of algorithm 1. OR Rate is the ratio of the number of *correctly* classified *outlying* sensors to the number of *all* true outlying sensors.

B. Event *Boundary* and *Non-Boundary* Sensor Classification

First, we evaluate the performance of *Boundary* sensor detection under the condition of no outlying sensors. For space reason we only report the results for EL_OL, LB_OL, and the star-graph event shapes. The simulation results for EL_OW and LB_OW are similar. The sensitivity parameter in algorithm 1 is set as $\zeta_1 = 0.25$. The upper bound of the component number is set as $\mathrm{K} = 5$. The size of the neighborhood sensor set $\mathcal{N}(S_i)$ is 196. These parameters keep as default in section V unless otherwise stated.

Boundary sensors are considered as lying on the boundary of multiple (two or more including background noise) sensing events, while *Non-Boundary* sensors existing inside the region of any single event. Refer Table I for the performance of our algorithm in the average of 100 runs.

[3]This is equivalent to sampling from $\mathcal{G}(\mu_i, \Sigma_i) + \mathcal{G}(\mu_j, \Sigma_j)$, which is still a *Gaussian* component for the multivariate Gaussian distributions. Thus our proposed algorithm is still applicable.

Comparison of Different Algorithms under Five Event Shapes					
Shape	Algorithm	EB Rate	ER Rate	TP Rate	FP Rate
EL_OW	EM_BIC	0.0906	0.0019	0.9902	0.0012
	Median	0.0179	0.034	0.6438	0.0211
LB_OW	EM_BIC	0.1712	0.0232	0.9632	0.0042
	Median	0.0304	0.0692	0.6354	0.0436
EL_OL	EM_BIC	0.0844	0.0123	0.9722	0.0084
	Median	0.0215	0.035	0.7651	0.0247
LB_OL	EM_BIC	0.1527	0.0204	0.9646	0.009
	Median	0.0243	0.0698	0.6108	0.0461
Star	EM_BIC	0.118	0.0066	0.9713	0.0036
	Median	0.02	0.0519	0.2462	0.0283

TABLE I
Comparison of Median *based method [4] and algorithm 1.*

Comparison with *Median* based method [4]: We compare our new algorithm on *Boundary* sensor detection with the *Median* based method [4] using bi-section scheme (which is claimed better than tri-section. The distribution-wise assumption for *Boundary* sensor in [4] is defined as having two mixture components with each weighting 0.5 ($\mathrm{w}_j \approx 0.5$). Here we set $\zeta_1 = 0.45$ to mark the ground truth of the *Median* based method for statistical stability where 0.5 is too strict to have sufficient numbers of *Median* based *Boundary* sensors. Therefore the EB rates are noticeably less than the above case of algorithm 1 with $\zeta_1 = 0.25$, but ER, TP and FP Rates are still meaningful metrics to be compared. The other parameter settings are the same as default.

Table I shows numerical error comparison of different algorithms with the average of 100 runs under five event shape configurations when $q = 3$. Our algorithm 1 (denoted as EM_BIC) outperforms the previous *Median* based methods (denoted as Median) [4] in all five event configurations, with a statistically significant margin of higher TP Rates but lower ER and FP Rates.

Robustness against outlying sensors: To test the robustness of algorithm 1 against outlying sensors, we randomly choose a portion ϖ (eg., 5%, 10%, ...) of sensors and add a noise offset as $randn(3, 1) \times 30$ to each of their original readings. The readings of other sensors and the parameter settings are kept same as before. Table II shows the performance evaluation of our algorithm 1 under different outlying sensor ratios ϖ in EL_OL. All results are the averages of 100 trials with all other default settings. Performance of algorithm 1 drops gracefully with increasing ϖs. The TP Rates of *Boundary* sensors remain at the same level, but more false alarms (FP Rates) appear. We also test the robustness of *Median* based algorithms for comparison purpose. It degrades about 2.13 times faster than our *mixture* model approach in FP Rate. Due to the space limitation, we omit numerical details. Results of other four cases are *similar*. Importantly, the OR Rates of *outlying* sensor detection by using algorithm 1 are also reported in Table II. This result shows slightly worse performance but comparable stability than *boundary* sensor detection as ϖ increases. Our *mixture* model based framework is capable to provide an unified approach for simultaneous *boundary* and *outlying* sensor detection.

Robustness Evaluation on the outlying sensor ratios ϖ of Algorithm 1					
Metric	$\varpi = 0\%$	$\varpi = 5\%$	$\varpi = 10\%$	$\varpi = 15\%$	$\varpi = 20\%$
EB Rate	0.0844	0.0859	0.0864	0.0834	0.0858
ER Rate	0.0123	0.0148	0.0159	0.0193	0.0264
TP Rate	0.9722	0.9691	0.9519	0.9583	0.9562
FP Rate	0.0084	0.0108	0.0122	0.0179	0.0231
OR Rate	——	0.9481	0.9238	0.9029	0.9185

TABLE II
Robustness evaluation on the outlying *sensor ratio ϖ of algorithm 1.*

Sensitivity and Flexibility Testing on ζ_1 of Algorithm 1				
Metric	$\zeta_1 = 0.15$	$\zeta_1 = 0.25$	$\zeta_1 = 0.35$	$\zeta_1 = 0.45$
EB Rate	0.1707	0.0844	0.0678	0.0262
ER Rate	0.0081	0.0123	0.0109	0.0082
TP Rate	0.9671	0.9722	0.9695	0.9794
FP Rate	0.0033	0.0084	0.0054	0.0046

TABLE III
Performance evaluation on the different ζ_1 settings of algorithm 1.

C. Performance Sensitivity Testing

In the following, we evaluate the performance of algorithm 1 on different parameter settings, i.e., threshold ζ_1, covariance scale σ, and neighborhood size ϕ.For space limitations, we take EL_OL as an illustrative case for error analysis. Results of other four cases are *similar*.

1) Performance Sensitivity and Flexibility Testing on ζ_1: Table III shows the performance evaluation of algorithm 1 on different settings of parameter ζ_1. As shown in the table, the *Boundary* and *Non-Boundary* sensor classification accuracy is insensitive to ζ_1. When ζ_1 increases, for example from 0.15, 0.25, 0.35 to 0.45, smaller EB Rates and thinner *Boundaries* are obtained, which is as expected. However at the meanwhile, the classification accuracy metrics of ER Rates, TP Rates and FP Rates remain at the same level as in algorithm 1 for EL_OL, which shows the good stability over different ζ_1s.

2) Performance Sensitivity on Covariance Scales: In our experiments, all *Gaussian* signal components (including background white noises) share the same covariance matrix Σ_0, which is defined in section V-A and σ is a key factor controlling the noise levels. Thus we test the performance sensitivity of algorithm 1 with different σ settings in Table IV. Using EL_OL as an example, we conclude that our algorithm 1 is very *insensitive* to σ changes. No statistically significant performance downgradings are observed when σ varies from 2 to $5, 10$ and 15.

3) Performance Sensitivity on Neighborhood Size: We evaluate the performance sensitivity on sensor neighborhood size ϕ of algorithm 1 in EL_OL. From Table V, by varying ϕ from $36, 100, 196$ to 400, the *EB Rate* stays at the same level of $\phi = 36, 100$, and raises slightly when $\phi = 196$ and noticeably under $\phi = 400$. At the meantime, the *ER Rate* is stable when ϕ changes from 36 to 196, but increases obviously when $\phi = 400$. It indicates that the impact of the changes of sensor neighborhood sizes ϕ on algorithm performance is stable within a moderate range of $[36, 196]$. When ϕ is extremely small, not enough sensors are available for *EM* based statistical *GMM* algorithm. On the other hand, *EM* optimization in algorithm 1 tends to output more *Boundary* sensors, under much wider distributional variations when $\phi \geq 400$.

Performance Sensitivity Testing on σ of Algorithm 1				
Metric	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 15$
EB Rate	0.0823	0.0844	0.0850	0.0838
ER Rate	0.0120	0.0123	0.0127	0.0125
TP Rate	0.9768	0.9722	0.9699	0.9703
FP Rate	0.0077	0.0084	0.0101	0.0092

TABLE IV
Performance sensitivity testing on covariance scale σ of algorithm 1.

Performance Sensitivity Testing on ϕ of Algorithm 1					
Metric	$\phi = 36$	$\phi = 100$	$\phi = 196$	$\phi = 400$	$\phi = 900$
EB Rate	0.0630	0.0646	0.0844	0.1270	0.1871
ER Rate	0.0126	0.0124	0.0123	0.0209	0.0494
TP Rate	0.9706	0.9796	0.9722	0.9743	0.9820
FP Rate	0.0086	0.0084	0.0084	0.0187	0.0519

TABLE V
Performance sensitivity testing on neighborhood size ϕ of algorithm 1.

VI. Conclusion

In this paper, we first propose a novel, unified solution based on *Gaussian* mixture modeling with explicit model selection to detect *Boundary* sensors, which achieves accurate, robust experimental performance by leveraging its probabilistic nature. We plan to study how to recognize sensor events from different resources (e.g., a certain type of vehicle) and track the scale changes (besides location) of a moving event region in our future research.

Acknowledgments

This research was partially supported by the US National Science Foundation under grants CNS-0347674, CNS-0721669, and CNS-0831852.

References

[1] H. Akaike, Information theory and an extension of the maximum likelihood principle, *Second International Symposium on Information Theory*, 267-281, 1973.

[2] K. Chintalapudi and R. Govindan, Localized Edge Detection in Sensor Fields, *IEEE Ad Hoc Networks Journal*, pp. 59-70, 2003.

[3] A.P. Dempster, N.M. Laird, and D. Rubin, Maximum likelihood estimation from incomplete data via the EM algorithm, *Journal of The Royal Statistical Society*, 39(1):138, 1977.

[4] M. Ding, D. Chen, K. Xing, and X. Cheng, Localized Fault-Tolerant Event Boundary Detection in Sensor Networks, *IEEE INFOCOM*, March, 2005.

[5] M. Figueiredo and A.K. Jain, Unsupervised learning of finite mixture models, *IEEE Trans on Pattern Analysis and Machine Intelligence*, 24(3):381-396, 2002.

[6] G. Jin and S. Nittel, NED: An Efficient Noise-Tolerant Event and Event Boundary Detection Algorithm in Wireless Sensor Networks, *Proceedings of the 7th Int. Conf. on Mobile Data Management*, 2006.

[7] B. Krishnamachari, S. Iyengar, Distributed Bayesian Algorithms for Fault-Tolerant Event Region Detection in Wireless Sensor Networks, *IEEE Trans. Computers*, vol. 53, no. 3, pp. 241- 250, Mar. 2004.

[8] X. Meng, G. Jiang, H. Zhang, H. Chen and K. Yoshihira, Automatic Profiling of Network Event Sequences: Algorithm and Applications, *IEEE INFOCOM*, April 2008.

[9] G. McLachlan and D. Peel, *Finite Mixture Models*. New York: John wiley & Sons, 2000.

[10] R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. *VLDB Conference*, 1994.

[11] G. Schwarz, Estimating the dimension of a model, *Annals of Statistics*, Vol. 6, pp. 461-464, 1978.