Sensor Fusion-based Event Detection in Wireless Sensor Networks

Majid Bahrepour, Nirvana Meratnia, and Paul J. M. Havinga

(m.bahrepour, n.meratnia, p.j.m.havinga)@utwente.nl
Pervasive Systems Group, Twente University, the Netherlands

Abstract. Recently, Wireless Sensor Networks (WSN) community has witnessed an application focus shift. Although, monitoring was the initial application of wireless sensor networks, in-network data processing and (near) real-time actuation capability have made wireless sensor networks suitable candidate for event detection and alarming applications as well. Unreliability and dynamic (e.g. in terms of deployment area, network resources, and topology) are normal practices in the field of WSN. Therefore, effective and trustworthy event detection techniques for the WSN require robust and intelligent methods of mining hidden patterns in the sensor data, while supporting various kinds of dynamicity. Due to the fact that events are often functions of more than one attribute, data fusion and use of more features can help increasing event detection rate and reducing false alarm rate. In addition, sensor fusion can lead to more accurate and robust event detection by eliminating outliers and erroneous readings of individual sensor nodes and combining individual event detection decisions. In this paper, we propose a two-level sensor fusion-based event detection technique for the WSN. The first level of event detection in our proposed approach is conducted locally inside the sensor nodes, while the second level is carried out in a level higher (e.g., in a cluster head or gateway) and incorporates a fusion algorithm to reach a consensus among individual detection decisions made by sensor nodes. By considering fire as an event, we evaluate our approach through several experiments and illustrate impact of sensor fusion on achieving better results.

Keywords: Wireless sensor networks, event detection, data/sensor fusion

1 Introduction

A wireless sensor network (WSN) typically consists of a large number of small, low-cost sensor nodes distributed over a large area. The sensor nodes are integrated with sensing, processing and wireless communication capabilities. Each node is usually equipped with a wireless radio transceiver, a small microcontroller, a power source and multi-type sensors.

Although, WSN was originally considered as a monitoring platform, recent advances and achievements have made them suitable candidate also for event detection and actuation. In this end, event detection using the WSN has recently attracted much attention. Interesting applications vary from industrial safety and security to meteorological hazard, earthquake, and fire detection [1].

Resource constraints of the sensor nodes, dynamic and often volatile nature of the deployment area [2] and the network itself introduce unique challenges for researchers in the field. The designed event detection technique should therefore be light-weight because of limited computational capability of the sensor nodes. It should be distributed to reduce communication and transmission overhead as well as to increase the robustness by overcoming the problem of a single point failure.

Studying related work in the field of event detection reveals two major trends, i.e., centralized and decentralized. In centralized approaches, in which event detection is conducted in a base station, the focus has mostly been on data-aggregation and reducing communication overhead. In decentralized approaches, in which event detection is carried out inside the network and even inside every individual node, the

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

focus has often been on networking aspects and consensus. What is greatly missing in the field of event detection for the WSN, however, is the issue of online and distributed data mining, feature extraction, pattern recognition and matching, and data/sensor fusion.
In this paper, we consider fire as the event and propose a two-level sensor fusion-based approach for fire detection in WSN. The main idea behind our technique is to use computation capability of the sensor nodes and let them decide whether or not they detect any event. The decision of individual nodes will be fused in a higher level to detect whether eventually an event has occurred. This two-layer fusion-based technique provides extra support for dynamic nature of the network, physical failure of the nodes, and erroneous readings.
The rest of this paper is structured as follows. Section 2 presents the previous studies in the area of event detection. Section 3 describes our proposed event detection approach. Section 4 reports the empirical results and finally some conclusions are made in Section 5.

2 Literature Review

Only recently, pattern matching and online feature extraction are considered as the focus of event detection in the WSN community. It is a normal practice in the related literatures to assume some simple signatures for the events and then focus on solving the networking and communication related issues. To this end, existing work focus on issues such as reducing the number of required messages to be able to make a reliable decisions [3], minimizing the effects of message/packet losses [4], and support techniques for data corruption [5]. Here, however, we solely focus on studies related to other aspects of the event detection than purely communication.
A group of studies has focused on centralized event detection. Corenell Cougar [3], DIMENSIONS [4], Rutgers Dataman [5], SINA [2], SCADDS [6] and Smart-msgs [7] are examples of such techniques.
Martincic et al. proposed a grid based approach to perform in-network event detection in large scale sensor networks in a distributed manner [8]. Event detection was carried out by matching the sensor data against a pre-defined signature.
Escalation [9] is an adaptive system which can adapt to any environment and detect events. A general signature for any possible event is made and a distance vector calculates the distance between the sensory information and the event. The distance operator is the event detection technique for their approach.
D-FLER [10] uses a distributed fuzzy engine for event detection. It combines the individual sensor readings with data from their neighborhoods to produce a more accurate and robust classification result.
A fault-tolerant event detection scheme is proposed in [11]. Since the event-data and noises are different with normal-data, a distributed Bayesian algorithm discriminates between sensor-node failures (noises) and event.

3 Proposed Approach

The focus of this study is on online and distributed event detection in the WSN. Our approach is making use of data fusion as an enabling tool to increase resilience of the event detection technique against dynamic nature of the WSN itself and the deployment area. While designing our technique, we pay special attention to properties such as being computationally inexpensive, accurate, fault tolerant, energy efficient, and being distributed.
Our proposed approach has two-levels. In the first level, each sensor node will individually decide on detecting an event using a classifier. When an event occurs, we expect that event signature is propagated throughout the neighboring nodes and they will also detect the event. Therefore, we use a fusion technique in the second level to distinguish between outliers occurring at individual nodes and events that more nodes agree upon.
As we have previously shown in [12] , both feed forward neural network (FFNN) and Naïve Bayes classifiers have low computational complexity and also provide high classification accuracy. Thus, we will use these classifiers in our approach and first briefly introduce them in the next two subsections.
Since, we have a specific event, i.e., fire, in mid, we can do the learning offline. This implies that fire as an event has a rather known signature and by defining this signature, translating that into weights (in case of FFNN) and probability density function (in case of

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

Naïve Bayes), and program the sensor nodes using business rules [13], the node only need to evaluate a simple mathematic formula. This is perfectly in line with keeping the WSN requirement of keeping computational complexity low.

3.1 Feed Forward Neural Network (FFNN)

The artificial neural network (ANN) is a mathematical model based upon biological neural networks. It is composed of an interconnected group of artificial neurons and processes information using a connectionist approach for computation [14]. Feed forward neural network (FFNN) is a sort of the neural networks, in which each layer is fed by its back layer [15]. FFNN consists of one input layer, one or more hidden layers and one output layer. Fig. 1 shows the FFNN architecture.

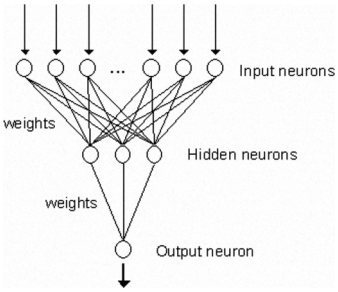


Fig. 1: Architecture of a Neural Network

To use FFNN as a classifier, the challenge is finding the weights. The process of finding the appropriate weights, which is called 'learning', can be carried out using algorithms such as gradient descent (GD).

3.2 Naïve Bayes Classifiers

A Naïve Bayes classifier uses Bayesian statistics and Bayes' theorem to find the probability of each instance belonging to a specific class. It is called Naïve because of emphasizing on independency of the assumptions. To find the probability of belongingness of each instant to a specific class, Eq. 1 can be used, which expresses the probability of an example $E = (x_1, x_2, ..., x_n)$ belonging to class c [16].

$$p(c \mid E) = \frac{p(E \mid c)p(c)}{p(E)} \qquad \text{(Eq. 1)}$$

3.3 Advantages of FFNN and Naïve Bayes Classifiers for WSNs

The main advantage of the FFNN is being easy to be programmed into a sensor node. Let us assume to have an FFNN with three neurons in input layer, two neurons in hidden layer and a neuron in output layer. The weights can be found by a gradient decent learning algorithm. After the weights have been found, we might have a network similar to Fig 2.

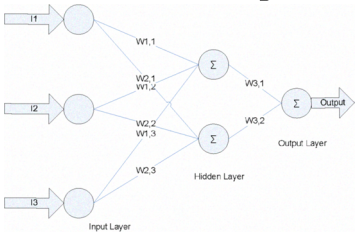


Fig. 2: An FFNN with three neurons in Input, two neurons in hidden layer, and one neuron in output layer along with their corresponding weights [12].

This network can be easily programmed into sensor nodes using Eq. 2., which formulates the network in a form of mathematical model. This should be noted that each neuron passes sum of the product (SOP) of the previous layer. In some networks SOP is given to a non-linear function, such as tangent and the transformation becomes nonlinear. In such a case Eq. 2 is changed slightly but, the general idea is the same.

$$Output = \left\lfloor W_{3,1} \times \sum_{j=1}^{3} (W_{1,j} \times I_j) \right\rfloor + \left\lfloor W_{3,2} \times \sum_{j=1}^{3} (W_{2,j} \times I_j) \right\rfloor \qquad \text{Eq. (2)}$$

Naïve Bayes classifier is also easy to implement. The most time-consuming part is how to compute $p(E \mid c)$ in Eq. 1. This probability can be

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

calculated once in an offline fashion and then programmed as a look-up table in sensor nodes [12].

3.4 The Proposed Approach

In our technique a classifier (either FFNN or Naïve Bayes) classifies the sensory data inside the sensor nodes and the output of the classifier, which indicates occurrence of an event, is fed to a higher level to be fused. This higher level can be a cluster head or gateway. Finding the optimal set of sensors is a topic by itself that in WSNs community was usually done intuitively by incorporating temperature and/or smoke sensors. However, looking into basic literature in traditional fire detection and investigating effects and impacts of individual sensors, reveals that a set of four-sensors, i.e., temperature, ionization, photoelectric and CO, are the most effective and have been proven to be the optimal sensor set for detecting fire [17].
Fig. 3 shows the general concept of the approach in a form of block-diagram.

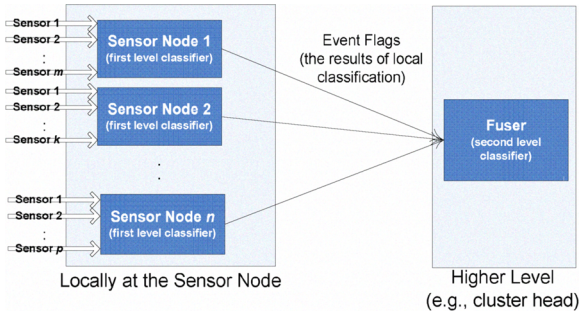


Fig. 3: Block Diagram of the Proposed Approach

Simply speaking, the first level performs a classification task in order to decide upon occurrence of an event. In doing so, it can use all its sensors or just a sub set (to accommodate for loss of a sensor or specific sensor reading). The second level learns how to deal with the decision results being provided by the previous stage and reach a consensus by fusing various event detection results. If it would be only one sensor node in the first level (*n=1*) the second level just learns the common errors in the first level; therefore, it would be a two-level classifier [18]. If there would be more that one sensor nodes in the first layer (*n>1*), the second layer becomes a fuser which fuses all the results in the previous level and reach a consensus whether or not an event has been occurred. Therefore, the first level is trained, and then, the results are given to the next stage. The second level is trained by the outputs of the first level. Consequently, the whole system is trained twice.

3.5 Computational Complexity Consideration

For each level we proposed to use either feed forward neural network (FFNN) or Naïve Bayes classifier. This implies that these two classifiers are not used in combination. Therefore, two classifiers use independent processing resources. Additionally, the training phase is conducted offline and only once, thus, its computational complexity is disregarded.
Complexity of FFNN is a function of m (number of features or number of nodes in the input layer), n (number of neurons in hidden layer), and p (number of neurons in output layer). Eq. 3 shows the time complexity of FFNN [12]:

$$O_{FFNN} = O(m \times n \times p) \quad \text{(Eq. 3)}$$

Complexity of Naïve Bayes is a function of m (number of features), i (number of classes), and j (number of partitions for distribution estimation). Eq. 4 shows the time complexity of Naïve Bayes classifier [12]:

$$O_{NaiveBayes} = O(m \times i \times j) \quad \text{(Eq. 4)}$$

4. Empirical Results

To evaluate our proposed approach, a data set was obtained and a number of experiments were conducted. We first describe our data set in Subsection 4.1, and then report on experimental results and comparison with existing approaches in Subsection 4.2.

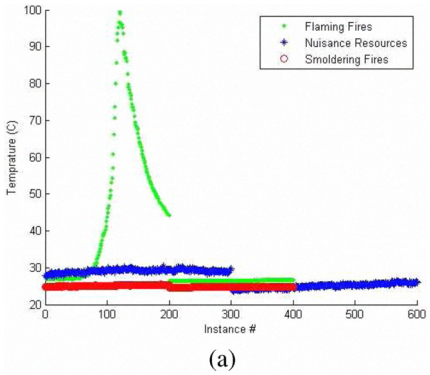
Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

4.1 Dataset

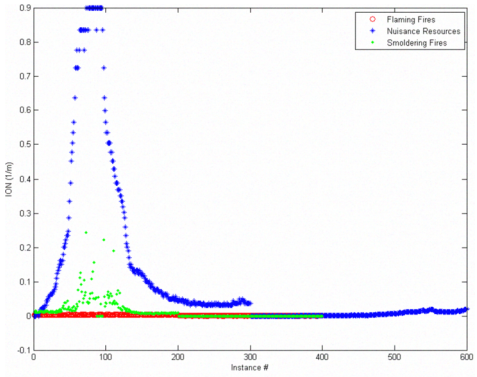
We use a data set from NIST (http://smokealarm.nist.gov/). In the original NIST data set, there are three separate fire-related data sets, (1) smoldering fire data, (2) flaming fire data, and (3) noise. To have a good balanced data set, therefore, we merge two smoldering fire dataset (SDC31, SDC40), two flaming fire dataset (SDC10, SDC14) and two nuisance dataset (MHN06, MHN16). A data set of 1400 data records was prepared. Fig. 4 displays the four features (i.e., temperature, ionization, photoelectric and CO) and their respective pattern in smoldering fire, flaming fire, and noise data sets.

The goal of the experiment is to make a classifier to correctly and fast classify each data measurement into its respective class, i.e., fires and noise.

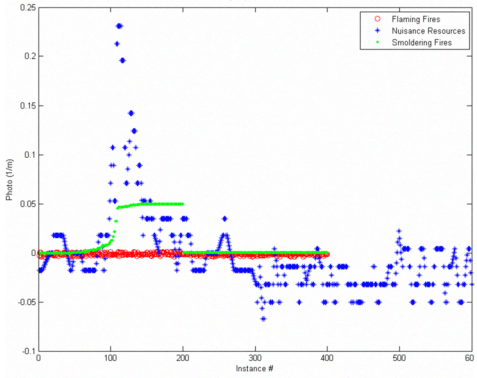
For the purpose of the experiments, the 1400 rows of data were split into 900 rows for training the first level classifier and a 500 row data for testing the first level classifier. Then, 500 row data (which is the output of the previous level classifier) is split to a 350 row data for training the second level classifier and a 150 row for testing it. The outputs of the first stage are in form of continuous values (decision about event occurrence) and are used for training the second level classifier.



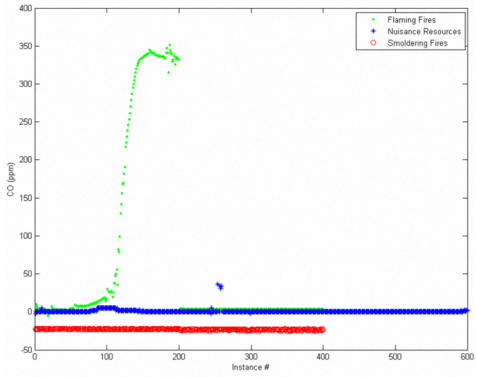
(a)



(b)



(c)



(d)

Fig. 4: (a) Pattern of Temperature data in smoldering fire, flaming fire, and noise data sets (b) Ionization data in smoldering fire, flaming fire, and noise data sets (c) Photoelectric data in smoldering fire, flaming

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

fire, and noise data sets (d) CO data in smoldering fire, flaming fire, and noise data sets

4.2 Experiments

We perform two experiments. In the first experiment, we only evaluate accuracy of the fire detection locally in the sensor nodes. We start with the assumption that each sensor node is equipped with all required sensors (in our case four). To test the robustness of our approach in presence of sensor failure or bad link quality, we then reduce number of sensors present on each node till there is only one sensor left. Another advantage in doing so is identifying contribution of each sensor set in detecting the fire. Table 1 and 2 report our empirical results of applying FFNN and Naïve Bayes classifiers only on the first level.

In the second experiment, we evaluate our complete two-level fusion based approach. After each node has individually and locally decided on occurrence of an event, it reports its decision to the higher level, where classification outputs of the first level are fused and cooperatively a consensus over occurrence of the event is made. Table 3 and 4 report our empirical results of the second experiments.

All the experiments were conducted 10 times and the average values are reported. In Tables 1-4, '✓' denotes the presence of a particular sensor, while '×' indicates the absence of that sensor.

Table 1. One Level Approach

FFNN				
TMP	ION	Photo	CO	Detection Accuracy
✓	✓	✓	✓	98.69
×	✓	✓	✓	98
✓	×	✓	✓	97.43
✓	✓	×	✓	98.53
✓	✓	✓	×	70
×	×	✓	✓	97.3
✓	×	×	✓	95
✓	✓	×	×	97
×	✓	✓	×	68.73
×	✓	×	✓	98
✓	×	✓	×	66.8
×	×	×	✓	94
×	×	✓	×	84.18
×	✓	×	×	45.68
✓	×	×	×	84.48

Table 2. One Level Approach

Naïve Bayes Classifier				
TMP	ION	Photo	CO	Detection Accuracy
✓	✓	✓	✓	71.58
×	✓	✓	✓	71.18
✓	×	✓	✓	70.27
✓	✓	×	✓	71.25
✓	✓	✓	×	63
×	×	✓	✓	70.2
✓	×	×	✓	71.78
✓	✓	×	×	60.05
×	✓	✓	×	62.8
×	✓	×	✓	71.13
✓	×	✓	×	63.28
×	×	×	✓	70.43
×	×	✓	×	61.05
×	✓	×	×	62.85
✓	×	×	×	62.5

The obtained results show that FFNN achieves better detection accuracy on one-level approach, while Naïve Bayes performs better as a whole. Another interesting finding is that contribution of CO sensor in fire detection is high; which means CO alone can result in an acceptable detection accuracy. Combining CO with ION, or ION with TMP can also improve the results. Moreover, the two-level approach is more precise and robust. This implies that in case of malfunctioning of any of the sensors or sensor nodes, the technique can perform with reasonable level of accuracy and reliability.

We also compare our best results with a recent study on fire event detection using the same dataset (called D-FLER [10]). D-FLER combines individual sensor inputs with neighborhood observation using a distributed fuzzy logic engine. The classification is then conducted based on two features (i.e., temperature and CO). Table 5 summarizes the comparison results.

Table 3. Two Level Approach

FFNN				
TMP	ION	Photo	CO	Detection Accuracy
✓	✓	✓	✓	98.93
×	✓	✓	✓	98
✓	×	✓	✓	98.73
✓	✓	×	✓	98.53
✓	✓	✓	×	95.66
×	×	✓	✓	97.4

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

✓	×	×	✓	98.83
✓	✓	×	×	97.4
×	✓	✓	×	90.33
×	✓	×	✓	98.11
✓	×	✓	×	96.86
×	×	×	✓	94.86
×	×	✓	×	87.8
×	✓	×	×	73
✓	×	×	×	88.6

Table 4. Two Level Approach

Naïve Bayes Classifier				
TMP	ION	Photo	CO	Detection Accuracy
✓	✓	✓	✓	99.33
×	✓	✓	✓	99.20
✓	×	✓	✓	99.73
✓	✓	×	✓	99.2
✓	✓	✓	×	97
×	×	✓	✓	99
✓	×	×	✓	98.66
✓	✓	×	×	93
×	✓	✓	×	99.53
×	✓	×	✓	99.13
✓	×	✓	×	94.27
×	×	×	✓	99.6
×	×	✓	×	90.4
×	✓	×	×	85.73
✓	×	×	×	90

Table 5. Comparing Empirical Results with D-FLER [10]

Best Result	Accuracy Rate	Computational Complexity
Naïve Bayes	99.73%	$O(m \times i \times j)$
FFNN	98.93%	$O(m \times n \times p)$
D-FLER [10]	98.67% [10]	$O(m \times k \times r \times o)$

It can be seen that both our approaches outperform D-FLER in terms of detection accuracy, while both have similar computational complexity. The complexity is based on the following variables: m is number of features, i is number of partitions (for probability density estimation) , j is number of classes (or outputs), n is number of neurons in hidden layer), p is number of neurons in output layer, k number of membership functions per input, r is the number of rules, o is the number of outputs (in the particular case of fire detection, o =1).

5 Conclusion

This paper presents the impact of sensor fusion on achieving a more robust and accurate event detection for the WSN. To comply with the requirement of detecting events locally and online, we propose an event detection technique, in which each sensor node is responsible for deciding whether an event has been occurred. To do so, it uses an FFNN or Naïve Bayes classifier. To support dynamic nature of the WSN, e.g., to cope with physical senor failure, bad link quality, or erroneous data, we extend our approach to a second level classification, in which the decision of individual nodes is fused and a consensus is reached. We have also considered the case when, for any reason, not all sensors required for event detection exist on one node. Comparing our best results with a recent study, in which a distributed fuzzy logic engine is used, proves superiority of our approach.
Our future work includes investigating applicability of other machine learning techniques for distributed and online event detection in WSN and defining a generic mechanism to detect more events.

Acknowledgement

This work is supported by the EU's Seventh Framework Programme, the SENSEI project.

Reference:

1. Vu, C.T., R.A. Beyah, and Y. Li. *Composite Event Detection in Wireless Sensor Networks.* in *Proc. of the IEEE International Performance, Computing, and Communications Conference* 2007.
2. Li, S., et al., *Event Detection Services Using Data Service Middleware in Distributed*

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056

Sensor Networks. Telecommunication Systems, 2004. **26**: p. 351-368.
3. Yao, Y. and J. Gehrke, *The Cougar Approach to In-network Query Processing in Sensor Networks.* ACM SIGMOD Record 2002. **31**(3): p. 9-18.
4. Ganesan, D., et al., *Multiresolution storage and search in sensor networks.* ACM Transactions on Storage (TOS) 2005. **1**(3): p. 277-315.
5. *Dataman Project*, www.cs.rutgers.edu/dataman.
6. *SCADDS: Scalable Coordination Architectures for Deeply Distributed Systems*, http://www3.isi.edu/home.
7. *Smart Messages Project*, discolab.rutgers.edu/sm.
8. Martincic, F. and L. Schwiebert. *Distributed Event Detection in Sensor Networks.* in *Systems and Networks Communications, 2006. ICSNC '06. International* 2006. Tahiti
9. Zoumboulakis, M. and G. Roussos, *Escalation: Complex Event Detection in Wireless Sensor Networks.* Smart Sensing and Context. 2007: Springer Berlin / Heidelberg.
10. Marin-Perianu, M. and P. Havinga, *D-FLER – A Distributed Fuzzy Logic Engine for Rule-Based Wireless Sensor Networks* Lecture Notes in Computer Science. Vol. 4836. 2008, Heidelberg: Springer Berlin.
11. Krishnamachari, B. and S. Iyengar, *Distributed Bayesian Algorithms for Fault-tolerant Event Region Detection in Wireless Sensor Networks.* Computers, IEEE Transactions on, 2004. **53**(3): p. 241- 250.
12. Bahrepour, M., N. Meratnia, and P.J.M. Havinga. *Use of AI Techniques for Residential Fire Detection in Wireless Sensor Networks.* in *Artificial Intelligence Applications in Environmental Protection Workshop, AIAEP 2009*. 2009. Thessaloniki, Greece.
13. Marin-Perianu, M., T.J. Hofmeijer, and P.J.M. Havinga. *Implementing Business Rules on Sensor Nodes.* in *Emerging Technologies and Factory Automation, 2006. ETFA '06. IEEE Conference on*. 2006.
14. Wikipedia, *Neural network*, Wikipedia.
15. Mehrotra, K., C.K. Mohan, and S. Ranka, *Elements of Artificial Neural Networks.* 1996, MIT Press.
16. Zhang, H. *The Optimality of Naive Bayes.* in *Seventeenth Florida Artificial Intelligence Research Society Conference*. 2004: AAAI Press.
17. Cestari, L.A., C. Worrell, and J.A. Milke, *Advanced Fire Detection Algorithms Using Data from the Home Smoke Detector Project.* Fire Safety Journal, 2005. **40**: p. 1-28.
18. Davarynejad, M. and M.-R. Akbarzadeh-T. *Time Series Prediction using Hierarchical Fuzzy Tools.* in *Proceedings of the Conference on Intelligent Systems (In Persian)*. 2005. Iran.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.7056
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.7056