Use of AI Techniques for Residential Fire Detection in Wireless Sensor Networks

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

(m.bahrepour, n.meratnia, p.j.m.havinga)@utwente.nl

Pervasive Systems Research Group, Twente University, the Netherlands

Abstract: Early residential fire detection is important for prompt extinguishing and reducing damages and life losses. To detect fire, one or a combination of sensors and a detection algorithm are needed. The sensors might be part of a wireless sensor network (WSN) or work independently. The previous research in the area of fire detection using WSN has paid little or no attention to investigate the optimal set of sensors as well as use of learning mechanisms and Artificial Intelligence (AI) techniques. They have only made some assumptions on what might be considered as appropriate sensor or an arbitrary AI technique has been used. By closing the gap between traditional fire detection techniques and modern wireless sensor network capabilities, in this paper we present a guideline on choosing the most optimal sensor combinations for accurate residential fire detection. Additionally, applicability of a feed forward neural network (FFNN) and Naïve Bayes Classifier is investigated and results in terms of detection rate and computational complexity are analyzed.

1 Introduction

Fires may take place in various environments, such as residential places, forests or open spaces. The easiest way to detect a fire at residential places is using the smoke detectors or any other similar sensors, which are usually sensitive to ionization or obscuration [1]. The problem with such detectors is that they are prone to false alarms. This means that in noisy conditions, such as smoking a cigarette or toasting a bread, a fire alarm may be generated wrongly [2, 3].

Generally, to reduce false alarms and perform fire detection accurately, two approaches are used [4]. The first approach uses one type of sensor and conducts the fire detection by a complex algorithm. An example of this approach is the work presented in [5], which uses a flame detection sensor and a fuzzy-wavelet classifier. In contrast, the second approach uses multiple sensors and performs the detection by a simple mathematical operation. The work presented in [2] is an example of the second approach, which uses CO and ionization (ION) sensors and a simple mathematic operation. Some researchers also tried to combine both approaches by using multiple sensors and an appropriate algorithm. The work presented in [6], which uses a feed forward neural network (FFNN) and four sensors, i.e., temperature, ION, CO and photoelectric, and their rising rates to discriminate fires from nuisance sources, is an example of the combined approach.

In recent studies, Wireless Sensor Networks (WSN) has also been proposed for fire detection [7-14]. In this type of research, fire detection in residential areas as well as forests and mines are considered as applications for WSN.

Although there are many achievements in the area of fire detection (in terms of selecting optimal sensors and algorithms) using individual sensors in general, these achievements often have not made their way into the WSN field. In this paper, we aim at bringing knowledge of already established fields of AI (because of their learning process, reasonable accuracy and computational cost) and fire detection into the WSN field.

The rest of this paper is structured as follows. Section 2 briefly reviews previous contributions for fire detection using WSN. In Section 3, our proposed fire detection technique is introduced. Section 4 reports the experimental results. Finally, some conclusions and future plans are given in Section 5.

2. Literature Review

In this section, contributions of WSN for fire detection are briefly surveyed. A more complete literature review on this matter can be found in our technical report [4].

Yu et al. [13] used the National Fire Danger Rating System (NFDRS) for forest fire detection. NFDRS inputs four sensory information (humidity, temperature, smoke and windy speed) and generates a fire-likelihood index. The contribution of this study is the function of a feed-forward neural network for data aggregation and reducing communication overhead.

Lu Zhiping et al. [14] proposed a forest fire detection approach using WSN. Their system is composed of some sensor nodes, gateway(s) and task manager(s). Each sensor node is equipped with temperature and humidity sensors. After obtaining sensory information at sensor nodes, the data is fused at the gateways and data analysis and decision making tasks are conducted by the task manager nodes.

In [7], the author incorporated Fire Weather Index (FWI) and a novel k-coverage algorithm to detect forest fires. K-coverage algorithm monitors each point by using *k* or more sensor nodes to improve fault tolerance. Therefore, some sensors can be put in standby mode to extend network lifetime. Although there are many algorithms to find the minimum number of sensors to be used, they are usually NP complete problems [12]. The proposed k-coverage solution proved that it can prolong the network life time.

Zervas et al. proposed a sensor network approach for early fire detection of open spaces such as jungles and urban areas [15]. They incorporated a temperature sensor and maximum likelihood algorithm to fuse sensory information. Their proposed system architecture is composed of (1) sensing subsystem, (2) computing subsystem, and (3) localized alerting subsystem. The author concluded the applicability of their approach for early fire detection.

A skyline approach for early forest fire detection is proposed in [10]. Skyline is built using greater values, i.e., those sensor readings with large temperature and high wind speed. Only data on skyline are sent to a sink to be used for fire detection. Sink processes the data according to the suggested algorithm and results in a fast and energy efficient forest fire detection.

Marin-Perianu et al. proposed a distributed fuzzy inference engine, called D-FLER, for event detection using WSN [9]. They considered fire as an event utilizing smoke and temperature sensors. D-FLER combines individual sensor inputs with neighborhood observation using a distributed fuzzy logic engine. The prototype of their work was implemented in practice using Ambient μNode 2.0 platform [16].

3. Proposed Fire Detection Approach

By looking at the previous work on fire detection using WSN, we can conclude that, use of WSN for fire detection can be improved in two directions. The first direction is to use more sensors in combination and conduct sensor fusion. This can lead to more accurate fire detection by incorporating more than one sensor [6]. The second direction is to use more intelligent detection algorithms such as AI approaches, as fires and nuisances have a distinct pattern.

In WSN research community, selection of sensors was often carried out randomly or assumption-basely. Although temperature sensors are probably the simplest and the most obvious sensors for fire detection, studying various sources in this field reveals that all researchers agree on the fact that it alone is not a suitable indicator for fires and gas concentration sensors result in a better fire detection and discriminating fire and noise sources [3,6]

In our approach, we adapt the optimal sensor set from [6] and use temperature, ionization, photoelectric and CO sensors. We assume that every node in the WSN contains all the required sensors. In this case, communication overhead between neighboring nodes is avoided and each sensor node can detect fire locally by itself.

To achieve this goal, sensor nodes need a computationally cheap, yet, efficient algorithm to conduct fire detection in a (near) real-time manner. For this reason, we propose to use FFNN and Naïve Bayes classifier. Subsections 3.1-3.3 provide information about these classifiers and the reasons why they are helpful for WSN.

3.1 Feed Forward Neural Network (FFNN)

The artificial neural network (ANN) is a mathematical model or computational 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 [17]. Feed forward neural network (FFNN) is a sort of the neural networks, in which each layer is fed by its back layer [18]. FFNN consists of one input layer, one or more hidden layers and one output layer. Fig 1 shows the FFNN's architecture.

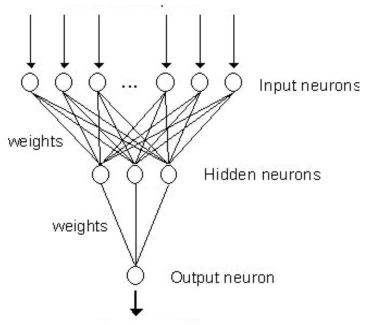


Fig. 1: Architecture of a Neural Network

The challenge of such networks is finding the weights. The process of finding the appropriate weights, which is called 'learning', can be carried out by some algorithms such as gradient descent (GD) approach.

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. 2 can be used. Eq. 2, expresses the probability of an example $E = (x_1, x_2, ..., x_n)$ belonging to class c [19].

$$p(c \mid E) = \frac{p(E \mid c)\, p(c)}{p(E)} \tag{1}$$

3.3 Advantages of the FFNN and Naïve Bayes Classifier for WSN

The main advantage of the FFNN for WSN is its ease 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 the GD learning algorithm. Then, we might have a network similar to Fig 2. Another advantage of the FFNN is its parallel capability, which means parameters used in Eq. 2 can be calculated independently and in parallel.

This network can be easily programmed into sensor nodes using Eq. 1. Evaluating this mathematical formula in form of a business rule is computationally very cheap and appropriate for resource constraint sensor nodes. This equation can be extended to more neurons and layers but the idea is the same. Eq. 2 formulates the network in a form of mathematical model. One should note that each neuron passes the sum of product (SOP) of the previous layer. In some networks SOP is given to a non-linear function such as tangent and transformation is a nonlinear one that makes Eq. 2 slightly different.

$$Output = \left\lfloor W_{3,1} \times \sum\nolimits_{j=1}^{3} (W_{1,j} \times I_j) \right\rfloor + \left\lfloor W_{3,2} \times \sum\nolimits_{j=1}^{3} (W_{2,j} \times I_j) \right\rfloor \tag{2}$$

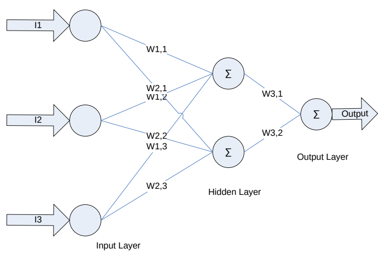


Fig. 2: An FFNN with three neurons in Input, two neurons in hidden layer, and one neuron in output layer along with their corresponding weights.

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 calculation is important to make the classifier more accurate. In basic literatures of pattern recognition or machine learning, it is proposed that this probability can be estimated by some standard data distribution such as Gaussian or Poisson [20].

To do a more accurate probability calculation, we can divide data into some intervals and count the data frequency within that interval. The new instances are also partitioned to the same intervals for finding the probability of each feature to be in that class.

To clarify the method, suppose we have the following data for ten samples in two classes A, B:

$A = [8,7,2,4,6,9,8,9,1,3]$

$B = [1,1,1,3,3,5,8,4,2,2]$

Then we divide these data into two intervals. Two intervals were chosen to simplify the example however the number of intervals are arbitrary. Therefore, those numbers less than five are allocated in the first interval, i.e., interval i_1, and the rest in the second interval, interval i_2.

Table 1. Classes and their Probability

	i_1 ($x < 5$)	i_2 ($x \geq 5$)
P_A	0.4	0.6
P_B	0.8	0.2

Now, let us assume to have an instance $x_1 = 3$ that should be classified into either Class *A* or Class *B*. It can easily be discovered that 3 belongs to the first interval, i_1, as it is less than 5. Then by looking at the probability table, Table 1, this can be seen that the probability of belongingness to class *B*, is higher ($P_B = 0.8 > P_A = 0.4$). Therefore we classify x_1 to class *B*.

This method of classification is also considerable for WSN because this is based upon a table which can be computed offline. Thereafter, this table is programmed into a sensor node and a simple algorithm inside the sensor nodes searches the table for the higher probable class.

In the next section the empirical results for both approaches is presented and also compared with a recent study.

4. Empirical Results

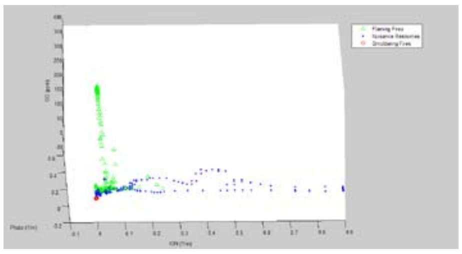
To evaluate the proposed approach, a set of data were obtained and a number of experiments were conducted. Subsection 4.1 describes the dataset, while Subsection 4.2 reports and compares the final results.

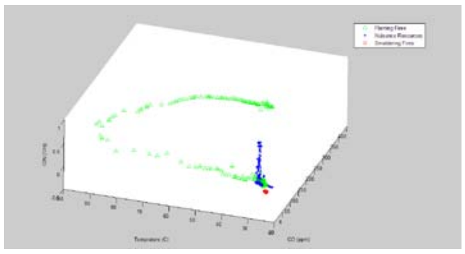
4.1 Dataset

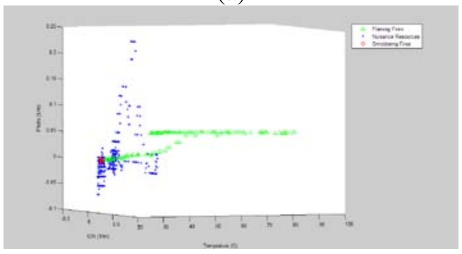
A set of data were obtained from NIST website (http://smokealarm.nist.gov/). To identify smoldering fire data, flaming fire data is combined with noise. Therefore, two smoldering fire dataset (SDC31, SDC40), two flaming fire dataset (SDC10, SDC14) and two nuisance resource dataset (MHN06, MHN16) were merged together. Totally 1400 data records were prepared, all having same units. Fig. 3 displays the data in 3D space. The goal is to make a classifier that can separate these data and classify them into their respective class, i.e., fire and noise.

4.2 Experimental results

The data were given to both classifiers and the results were obtained. To perform a cross validation, 1400 data records were divided to a 1000 training data and a 400 test data. All data were randomly mixed and given to the classifiers. Each test repeated 10 times and the average accuracy rate by changing the classifiers' parameters is reported in Tables 2-3. Table 4 provides a general comparison of our approach with a recent study, in which a distributed fuzzy system was proposed for residential fire detection using WSN [9]


(a)


(b)


(c)

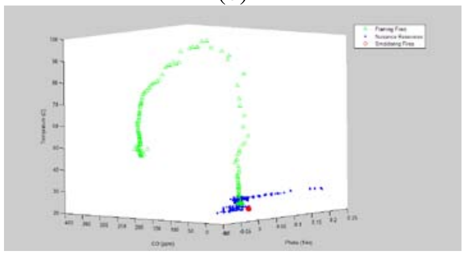

(d)

Fig. 3: Fire and noise data. (a) Ion, Photo and CO (b) Ion, temperature and CO (c) Temperature, Ion and Photo (d) Photo, CO and Temperature

For simulation of the proposed approaches Matlab$^{®}$ 7.1 was used. A two pass smoothing filter for a preprocessor was also applied that was adapted from [6].

Table 2. Empirical Results for Naïve Bayes Classifier

Number of Intervals	10	100	300	600	1000
Accuracy	32.15%	63.15%	96.425%	98.675%	100%

Table 3. Empirical Results for FFNN

Number of Neurons in the Hidden Layer	5	10	20	50
Accuracy Rate	97.495%	98.45%	93%	90.1%

Table 4. Comparing the Empirical Results with D-FLER [9]

Best Result	Naïve Bayes	Neural Network	D-FLER[9]
Accuracy Rate	100%	98.45%	98.67%

4.2 Computation Complexity Consideration

To compare these three approaches, not only the accuracy but also computation complexity is of significant importance, as they need to be implemented on tiny resource constraint sensor nodes.

4.2.1 FFNN's Computation Complexity

The most expensive part of the FFNN computationally is the training phase. Since we consider that FFNN is trained once and then is programmed into the sensor nodes, its computation complexity is negligible.

The computation complexity of a FFNN with m neurons in its input layer (number of features), n neurons in hidden layer, and p neurons in output layer is shown in Eq. (3).

$$O_{FFNN} = O(m \times n \times p) \tag{3}$$

In this calculation the multiplication operator is considered as the key for computation complexity calculation.

4.2.2 Naïve Bayes's Computation Complexity

The most expensive part of the Naïve Bayes classifier computationally is making the probability table. We assume that this probability table is made once and then is programmed into the sensor nodes. In this case computation complexity is calculated for search process only, which is more expensive. Computation complexity for the Naïve Bayes is calculated based on the Eq. (4), where m is number of features, i is number of classes, and j is number of intervals.

$$O_{NaiveBayes} = O(m \times i \times j) \tag{4}$$

4.2.3 D-FLER's Computation Complexity

Defining the fuzzy rules and membership functions represent the most complicated part of the fuzzy inference engine design. Assuming that these are programmed into the sensor nodes, the time complexity of the fuzzy inference engine is calculated based on the Eq. (5), where m is the number of membership functions per input, i is the number of inputs, r is the number of rules, o is the number of outputs (in the particular case of fire detection, $o=1$).

$$O_{D-FLER} = O(m \times i \times r \times o) \tag{5}$$

As shown in [9], the actual execution time can be greatly influenced by the specific defuzzification method chosen, to the extent that the number of outputs o can become the determinant factor.

4.2.4 Computation Complexity Comparison

Comparing computation complexity of the FFNN, the Naïve Bayes classifier, and fuzzy logic approaches shows that they are all product of three terms and if all variables have the same values it is a non-linear equation of power 3.

Table 5. Computation Complexity Comparison

	Naïve Bayes	Neural Network	D-FLER[9]
Computation Complexity	$O(i \times j \times m)$	$O(m \times n \times p)$	$O(m \times i \times r \times o)$

5. Conclusion

Wireless Sensor Networks may be deployed in many places thus they have different requirements. According to their scenarios each sensor node is either equipped with all the appropriate sensors or just a sub set of them. Fire in WSN is considered as an event; therefore event detection techniques are used for its detection. In this study, the optimal set of four sensors, i.e., temperature, ionization, photoelectric and CO, were adapted from [6] and two fire detection techniques based on the FFNN and the Naïve Bayes classifier were proposed to detect fire on each node locally. To carry out the detection task the sensory information is given to a classifier. The computation complexity and accuracy rate of each of these techniques and a comparison between them and a recent study, called D-FLER [9] based on fuzzy logic were presented. Results show that while all the three have similar computation complexity, the Naïve Bayes classifier can achieve a better accuracy and has a lower communication overhead (since it is centralized assuming all the sensors are present at the sensor node). However, in case just a sub set of sensors is present at each sensor node, D-FLER has the advantage. This is a good guideline to choose a proper technique for a particular scenario (centralized versus distributed) in mind.

Acknowledgment

Authors would like to thank Mihai Marin-Perianu for his contribution on complexity calculation of the D-FLER approach, and Thomas Cleary for his valuable input regarding unit conversions in preparing the data set. This work is supported by the EU's Seventh Framework Programme, Aware project and the SENSEI project.

Reference:

1. Brain, M. *How Smoke Detectors Work* 2000 [cited; Available from: http://home.howstuffworks.com/smoke1.htm.
2. Gottuk, D.T., et al., *Advanced fire detection using multi-signature alarm algorithms.* Fire Safety Journal, 2002. **37**(4): p. 381-394
3. Milke, J.A. *Using Multiple Sensors for Discriminating Fire Detection*. in *Fire Suppression and Detection Research Application Symposium*. 1999: National Fire Protection Research Foundation
4. Bahrepour, M., N. Meratnia, and P.J.M. Havinga, *Automatic Fire Detection: A Survey from Wireless Sensor Network Perspective.* . 2008, Centre for Telematics and Information Technology, University of Twente: Enschede.
5. Thuillard, M. *Application of Fuzzy Wavelets and Wavelets in Soft Computing Illustrated with the Example of Fire Detectors*. in *Wavelet Applications VII*. 2000.
6. 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.
7. Bagheri, M., *Efficient K-Coverage Algorithms for Wireless Sensor Networks and Their Applications to Early Detection of Forest Fires*, in *Computing Science*. 2007, SIMON FRASER UNIVERSITY. p. 75.
8. Bernardo, L., et al. *A Fire Monitoring Application for Scattered Wireless Sensor Networks: A peer-to-peer cross-layering approach* in *International Conference on Wireless Information Networks and Systems (WINSYS'07)*. 2007. Barcelona, Spain.
9. 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.
10. Pripužic, K., H. Belani, and M. Vukovic, *Early Forest Fire Detection with Sensor Networks: Sliding Window Skylines Approach*. 2008, White Paper: University of Zagreb, Faculty of Electrical Engineering and Computing, Department of Telecommunications.

11. Tan, W., et al. *Mine Fire Detection System Based on Wireless Sensor Network*. in *Information Acquisition, 2007. ICIA '07. International Conference on*. 2007.
12. Yang, S., et al., *On Connected Multiple Point Coverage in Wireless Sensor Networks* International Journal of Wireless Information Networks, 2006. **13**(4): p. 289-301.
13. Yu, L., N. Wang, and X. Meng. *Real-time forest fire detection with wireless sensor networks*. in *Wireless Communications, Networking and Mobile Computing*. 2005.
14. Zhiping, L., et al., *The Design of Wireless Sensor Networks for Forest Fire Monitoring System*. 2006, White Paper: School of Electronics and Information, Hangzhou Dianzi University.
15. Zervas, E., et al. *Fire Detection in the Urban Rural Interface through Fusion Techniques*. in *Mobile Adhoc and Sensor Systems (MASS 2007)*. 2007.
16. Hofmeijer, T., et al., *AmbientRT - Real Time System Software Support for Data Centric Sensor Networks.* Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2004: p. 61-66.
17. Wikipedia, *Neural network*, Wikipedia.
18. Mehrotra, K., C.K. Mohan, and S. Ranka, *Elements of Artificial Neural Networks*. 1996, MIT Press.
19. Zhang, H. *The Optimality of Naive Bayes*. in *Seventeenth Florida Artificial Intelligence Research Society Conference*. 2004: AAAI Press.
20. Alpaydin, E., *Introduction to Machine Learning* 2004: MIT Press.