AUBE '01

12TH INTERNATIONAL

CONFERENCE ON AUTOMATIC FIRE DETECTION

March 25 - 28, 2001
National Institute Of Standards and Technology
Gaithersburg, Maryland U.S.A.

PROCEEDINGS

Editors: Kellie Beall, William Grosshandler and Heinz Luck

NIST CENTENNIAL 1901-2001

EUSAS

NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

Strategies for the development of detection algorithms.

Dipl.-Ing. Rainer Siebel

Gerhard-Mercator-Universität-GH Duisburg
Fachbereich Elektrotechnik, Fachgebiet Nachrichtentechnik
Bismarckstrasse 81, 47048 Duisburg

Tel.: 0203/379-3366
Fax.: 0203/379-2902
e-mail: siebel@sent5.uni-duisburg.de

November 20, 2000

Dipl.-Ing. Rainer Siebel

Gerhard-Mercator-Universität, Duisburg, Germany

Strategies for the development of detection algorithms

1. Introduction

Multi-sensor/Multi-criteria based fire detectors require far more complicated detection algorithms as compared to simple threshold detectors, which are still used as single sensor based fire detectors. Cheap and powerful microprocessors are available to carry out this task. They may be implemented in each detector or alternatively as a central processing unit to carry out the fire or no-fire decision for all measured values being transmitted from various sensor-groups of a fire detection system.

A combination of smoke- and heat-sensors is at present frequently used in fire detection systems. The discussions not only on this congress show that a threefold sensor combination using smoke, heat and a gas sensors (for example CO) might be used in the foreseeable future.
The advantage of such a threefold combination is obvious for the simple reason that a more reliable alarm decision is possible if it is based on the observation of different physical phenomena associated with a genuine fire.

The combination of smoke- and heat-sensors is advantageous in case of open flame fires but the heat sensor does not help if the important class of smoldering fires is considered. A combination of smoke- and CO-sensor, however, would be beneficial in the latter case.

At present there are still some important restrictions to be taken into consideration for the design of new detection algorithms of higher complexity.

1. They possibly require much more computational power for each sensor combination. For hundreds of sensors per line this may quickly exceed the available computational power of a central processing unit since part of it is reserved for other administrative tasks (line protocols, handling of displays etc.).

2. Manufacturers of fire detection systems are for good reasons not too eager to replace their tried, tested and reliable central processing units by completely new devices with more powerful processors because a complete redesign is extremely expensive! For this reason the available processing power as well as the “Random Access Memory” (RAM) resources are usually rather limited.

3. The test authorities in Europe insist to understand at least the principle design rules of a detection algorithm. This is important as far as the selection of the detection method is concerned to avoid problems which may occur later in the certification process.

For these reasons it is recommendable to pursue a certain strategy and to keep in mind the given restrictions during the early design phase of a new detection algorithm.

2. Binary signal detection.

2.1 Principle solution of the detection problem.

Figure 1 shows a block diagram of a multi-sensor based detector and figure 2 illustrates the mode of operation of the decision-block from figure 1. H_0 denotes the so called

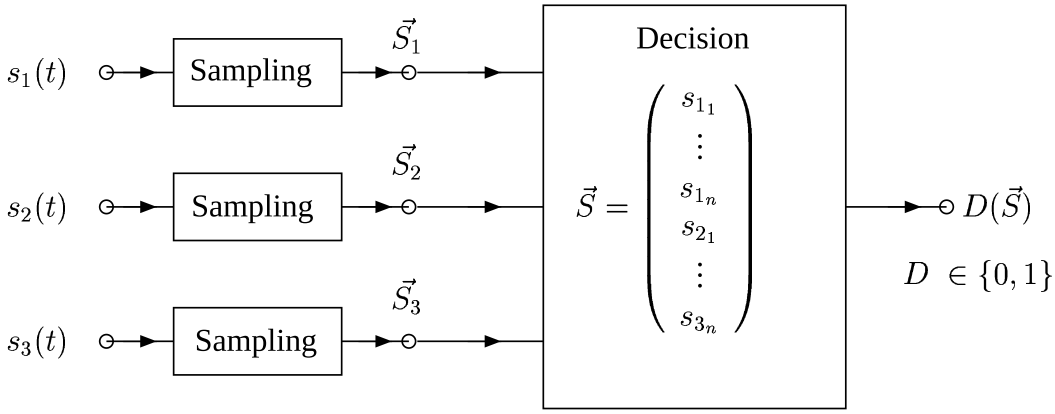

Figure 1: Block scheme of a multi-sensor based detector

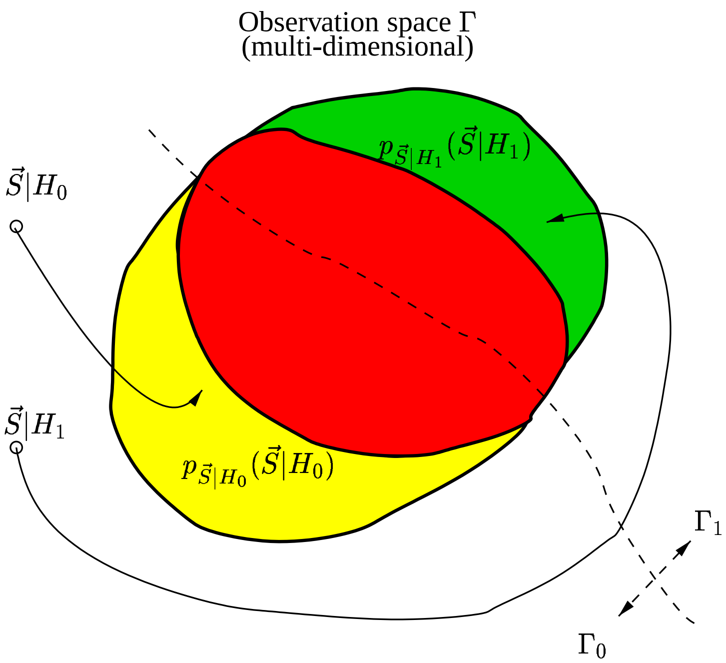

Figure 2: Observation space.

"zero-hypothesis", i.e. the signals $s_i(t)$ or signal vectors $\vec{S}|H_0$ originate from no-fire situations. Correspondingly, hypothesis H_1 denotes all signals $s_i(t)$ or signal vectors $\vec{S}|H_1$ originating from fire situations.

Each signal vector points to one specific location inside the multidimensional observation space, which is here schematically plotted as a two dimensional space.

In the aim to draw an unambiguous decision it is necessary to subdivide the observation space into two disjoint subspaces. This means:
if the signal vector points to the subspace $\Gamma_1 \Rightarrow$ the decision is: alarm.
if the signal vector points to the subspace $\Gamma_0 \Rightarrow$ the decision is: no-alarm.

Unfortunately, signal vectors $\vec{S}|H_0$ and $\vec{S}|H_1$ under either hypotheses may point to both subspaces Γ_0 and Γ_1. Hence, we have not only two types of correct decisions but moreover two possible types of wrong decisions - a false alarm or a missed alarm.

The most simple case: threshold detector for one sensor. (Here the observation space is a one dimensional straight line.)

Nearest more complicated case: Two subsequently measured values from one sensor or one measured value from each of two sensors. (Here the observation space is a two dimensional plane.)

and so forth: one measured value from each of three sensors (with a three dimensional observation space.).

n measured values from each of m sensors (requires a representation in a $m \cdot n$ dimensional hyper-space).

The well known decision theory clearly states, how to divide the observation space into two disjoint subspaces taking into account certain optimization criteria. The “Bayes”-decision rule is as follows (see for example [1], [2]):

$$\Lambda(\vec{S}) = \frac{p_{\vec{S}|H_1}(\vec{S}|H_1)}{p_{\vec{S}|H_0}(\vec{S}|H_0)} \quad \begin{matrix} H_1 \\ \geq \\ < \\ H_0 \end{matrix} \quad \frac{q_0 \cdot K_\alpha}{q_1 \cdot K_\beta} = K \tag{1}$$

From the formula above follows that we need to know the conditional multivariate probability densities $p_{\vec{\mathbf{S}}|\mathbf{H_i}}(\vec{S}|H_i) \quad \forall i \in \{0,1\}$. Unfortunately, these conditional probability densities are neither known nor determinable. Particularly, our knowledge concerning the signals of different sensors in no fire situations is very poor.

Intuitively, engineers would apply something similar to the above decision rule even if they have never heard of the “Bayes decision rule”.

Consider for example a decision, which is based on one measured value from each of two sensors for smoke and CO-concentration. A reasonable subdivision of the total observation space could be chosen as shown in figure 3 where we have plotted a sequence of measured sample values $s_1(t_i), s_2(ti)\ \forall i \in \{1,(1),n\}$ from a scattering light smoke sensor and a CO-measuring device which has been recorded during a TF2-smoldering-test-fire. Each pair of measured values is represented by one point in the $s_1,\ s_2$-plane.

Intuitively, an engineer would fix for example two threshold values $sw_1,\ sw_2$ in such a way that all points in the plane which are above the thresholds indicate more likely a genuine fire than a no-fire situation and vice versa - according to the assumption of the engineer. This reasoning corresponds to the decision rule in eqn.(1) but there however, the conditional probability densities are assumed to be known and thus, the borderline between the subspaces would look quite different.
The “Bayes decision rule” moreover takes into account the probabilities of occurrence

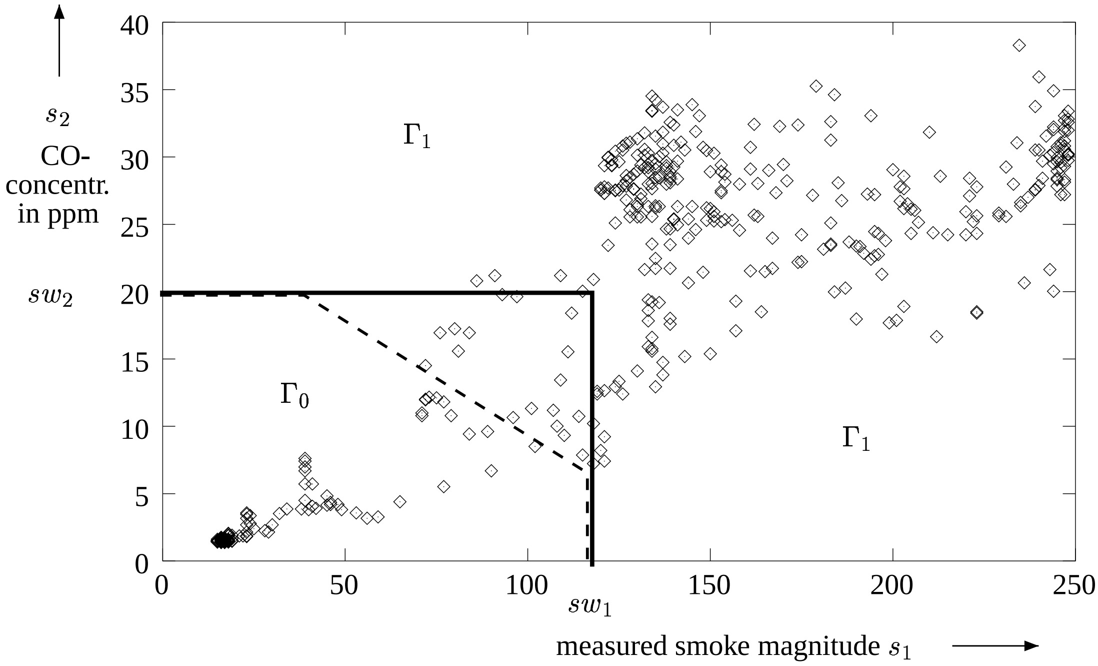

Figure 3: Intuitive selection of the subspaces Γ_0 and Γ_1.

for fire- and no-fire situations (q_0 and q_1) and cost- or risk factors (K_α and K_β) assigned to correct or incorrect decisions.

Thus, the observation space is subdivided into two disjoint subspaces Γ_0, Γ_1 as indicated by the solid line in the plot of figure 3. The corresponding decision rule in mathematical terms is:

$$\text{if } ((s_1(t_i) > sw_1) \text{ or } (s_2(t_i) > sw_2)) \Rightarrow \text{alarm}, \tag{2}$$

which is a simple twofold threshold detector.

Slightly more complicated is a subdivision of the observation space as indicated by the dashed line in figure 3. This approach reflects the fact, that simultaneously existing smoke and CO-concentration indicates more likely a fire than solely the presence of smoke or CO-concentration. In this case it makes sense to trigger an alarm at lower magnitudes of both smoke and CO-concentration.

The corresponding decision rule in this case is as follows:

$$\text{if } ((s_1(t_i) > sw_1) \text{ or } (s_2(t_i) > sw_2) \text{ or } (s_1(t_i) + s_2(t_i) > sw_3)) \Rightarrow \text{alarm} \tag{3}$$

The joint evaluation of one sample from each of three sensors (like for example smoke, heat and CO-concentration) is more complicated because in this case it is necessary to fix a border-plane in the 3-dimensional space.
If we intend to include not only actual samples but also the last m sampled values of each of the n sensor signals then the confusing task is to find an adequate hyper-plane in an $m \cdot n$ dimensional hyper-space. Thus, the task becomes more and more complicated with an increasing dimension of the observation vector and the question arises how to simplify the solution of the task.

2.2 Feature extraction methods

Feature extraction is a suitable method to simplify the decision problem considerably if we want to include the history of signals in our decision.

Feature extraction means to extract from each signal vector $\vec{S}_j$, consisting of n subsequently sampled values of the sensor signal, one (or a few) features $m_{j_k} \quad k \in \{1..\ell_j\}$ as shown schematically in figure 4.

Thus, we obtain a feature vector $\vec{M}$ of dimension $\sum_{j=1}^{m} \ell_j$, which is considerably smaller than the dimension $m \cdot n$ of the combined signal vectors. We can use the same decision method as previously described using the feature vector $\vec{M}$.

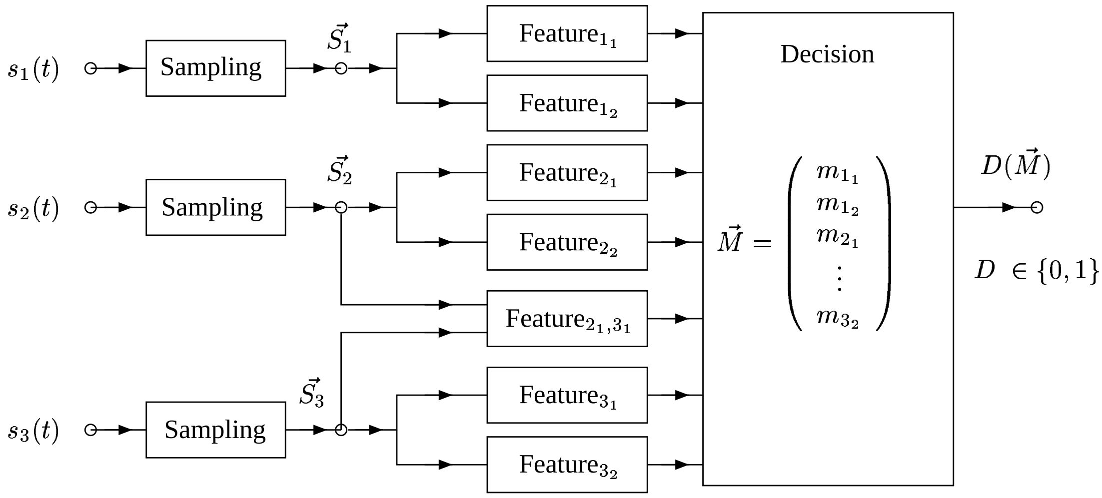

Figure 4: Schematic sketch of a detector with feature extraction.

Feature extraction is an important engineering task which requires to consider previously known individual characteristics of the sensors and their specific response in fire- and no-fire situations. Only a few examples are mentioned in the following:

- Determination of the average rate of rise.
- Suppression of the signal fluctuation which is considered to be irrelevant in no fire situations.
- Rate of rise limitation for sensor signals in those cases where step-like changes are unlikely to occur either physically or due to the features of the sensor (for example heat) but where step-like changes may occur due to data transmission errors or noise influence.
- Accumulation of signal amplitudes.
- Determination of significant spectral coefficients, if the signal fluctuation of sensors differs in fire- or no-fire situations.
- Joint evaluation of signal features from different sensors (correlation). e.t.c.

The few examples mentioned above show that feature extraction methods usually evaluate not only actual signal samples but include moreover the temporal development of signals. As an additional advantage we usually have a smaller variance of the features in comparison with the variance of the signals itself because features are calculated as some sort of average from several signal samples.

Feature extraction is applicable with more or less computational effort and storage-capacity efficient methods. One method to evaluate the temporal behaviour of signals is the so called "windowing"-method, i.e. feature extraction is based at each time instance with a new sample $s_i(k)$ on the last n-samples (where k denotes a discrete time variable):

$\vec{S}_i^T(k) = \{s_i(k), s_i(k-1), s_i(k-2), \ldots, s_i(k-n+1)\}$

If the next sample $s_i(k+1)$ is drawn the eldest previously used sample $s_i(k-n+1)$ is skipped and the new sample vector consists of the following components:

$\vec{S}_i^T(k+1) = \{s_i(k+1), s_i(k), s_i(k-1), \ldots, s_i(k-n+2)\}$

Usually, we can find a recursive calculation from this vector for whatever feature to be calculated in the following form:

$$\begin{aligned} m_{i_j}\left(\vec{S}_i^T(k+1)\right) &= m_{i_j}\left(\vec{S}_i^T(k)\right) + f[s_i(k+1), \overbrace{s_i(k), \ldots, s_i(k-n+2)}^{\text{common components}}] + \\ & \quad -f[\underbrace{s_i(k), \ldots, s_i(k-n+2)}_{\text{common components}}, s_i(k-n+1)] \end{aligned} \quad (4)$$

Recursive calculations reduce the computational effort significantly but not the required storage capacity.

Since features are usually calculated as some sort of average on account of the random nature of the signals, the following alternative method is not only computationally efficient but saves moreover storage capacity.

$$m_{i_j}(k) = a \cdot m_{i_j}(k-1) + (1-a) \cdot f[s_i(k),\ s_i(k-1); \{s_j(k)\}] \text{ and } 0.9 < a < 1 \quad (5)$$

In this formula $f(\cdot)$ denotes a so called "partial feature" which is calculated either from the actual sample $s_i(k)$, or from two subsequent samples of one sensor, or from one sample each of two sensors.
The updated feature $m_{i_j}(k)$ is calculated from its previous value $m_{i_j}(k-1)$ and the actually calculated partial feature $f(\cdot)$. This type of recursion is called "exponential windowing" and operates like a digital filter with RC-low-pass transfer-characteristic where the partial feature is the input signal and the output corresponds to the smoothed partial feature values. The denotation "exponential windowing" indicates, that the filter output corresponds to the sum of an exponentially weighted input sequence of the partial features with highest weight for the actual input and decreasing weights for previous partial features.
The constant a determines the 3dB-cut off frequency ω_g of the low-pass characteristic or the "memory-depth" respectively.

$\omega_g = -\dfrac{\ln(a)}{\Delta t}$ with Δt, the sampling period.

Apart from that, the use of integer or byte arithmetic saves storage capacity for random access variables as well as computation time because it is often possible to substitute multiplication- and division-operations by simple shift operations and integer or byte variables require less storage capacity by definition.

Although our electronic technology offers enormous processing power and giant storage capacity on one chip it seems to be important at present to regard the above mentioned aspects. The development of new control and indicating equipment with powerful central processors and large storage capacity requires time and enormous expenditures. But the use of tried and tested existing equipment with newly implemented detection algorithms sets hard limits with respect to storage capacity and computation time per detector.

A central processing unit in a fire detection system must carry out for example one decision/sec. based on feature extraction and preprocessing methods for hundredth of sensors or sensor combinations apart from various administrative tasks. For this reason only a few fractions of milliseconds remain as computation time for each sensor combination of a multi-sensor detector and the total amount of required RAM-storage capacity is proportional to the total number of detectors.

2.3 Automatic decision

The mode of operation of the decision-block in figure 1 or figure 4 we have already discussed in the previous section - i.e. the division of the observation- or feature-space into two disjoint subspaces. This task can be solved by two different approaches:

by a rule based decision **or** by a trained classifier.

For both of these methods there are various different realizations. Equations (2) and (3) represent without doubt simple rule based decisions. Clearly, the rule base becomes more complicated if more features are involved.

The advantage of a trained classifier is, that the designer does not need to think about various complicated rules for plenty of features because these rules are automatically determined during the training phase in a neural net for example. In other words the neural net automatically subdivides the multidimensional observation- or feature-space into the subspaces Γ_0 and Γ_1.

Some proposals using neural nets as fire detectors are already to be found in the literature. For this reason it seems to be necessary to investigate the chances and shortcomings of this method - particularly, because there are software tools available which configure and train neural nets without any requirement for the user to understand much about the background.

3. Detectors with neural nets.

3.1 Realization of a neural net.

Each neuron has $n+1$ continuous valued inputs where n denotes either the total number of vector components of the input feature vector $\vec{M}$ (or the signal vector $\vec{S}$) which is applied to the input nodes or the number of neurons in the previous layer. Each neuron

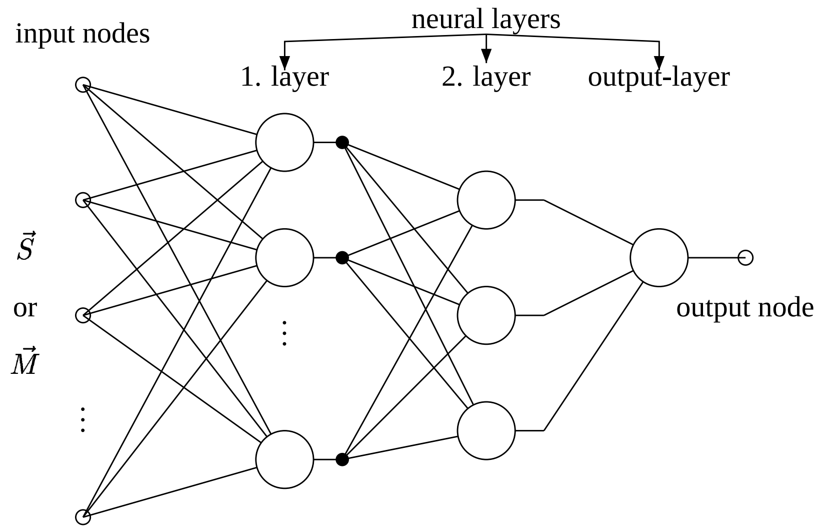

Figure 5: Sketch of a neural net.

computes a weighted sum of its input elements, subtracts a variable threshold and passes the result through a hard-limiting nonlinearity such that the output is between 0 and 1.

$$o_j(k) \;=\; \sigma\left(\sum_{i=0}^{n} w_{i,j} \cdot o_i(k-1)\right) \quad \text{with } o_0(k-1) = 1 \text{ and } o_i(0) = m_i \tag{6}$$

In the above formula k denotes the neural layer and the indices i, j the neurons in the $k-1$'st and k'th layer, respectively. The inputs of each neuron are weighted outputs (with weight $w_{i,j}$) of the neurons of the previous layer (or the input node values). The nonlinearity $\sigma(\ldots)$ is of a "sigmoid type"; two representative examples of which are shown in figure 6.

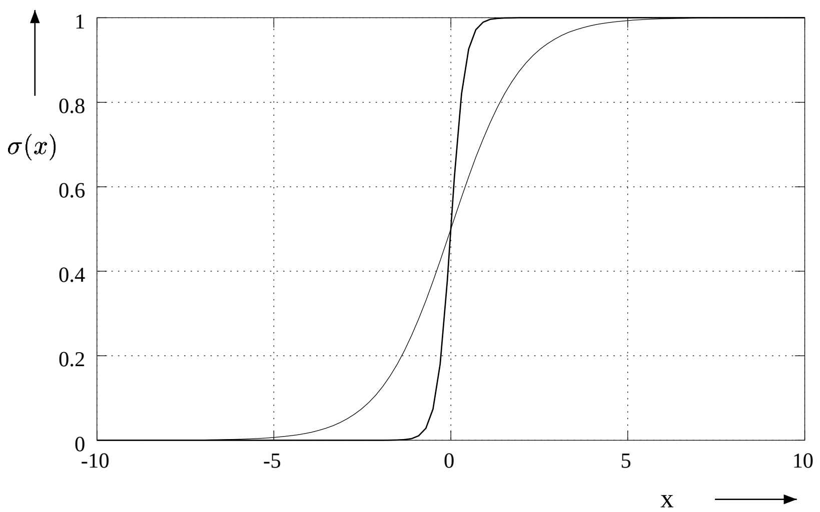

Figure 6: Hard-limiting nonlinearity of sigmoid type $\sigma(x)$

A net as shown in figure 5 transforms the set of all input vectors to a set of output values in the range 0..1 with a more or less steep transition between 0 and 1. This corresponds nearly with the task to subdivide the observation space into two subspaces, i.e. to assign a value of $0 \Rightarrow$ no-fire to all elements of a subset of the input vectors and a value of $1 \Rightarrow$ alarm to the complementary subset of the input vectors.

If a binary decision (fire or no-fire) is required one neuron is sufficient in the output layer with its output values in the range 0..1. If we interpret the output values such, that for output values >0.5 the probability for a fire is higher than for output values <0.5, we only need one additional threshold comparison to achieve an unambiguous decision.

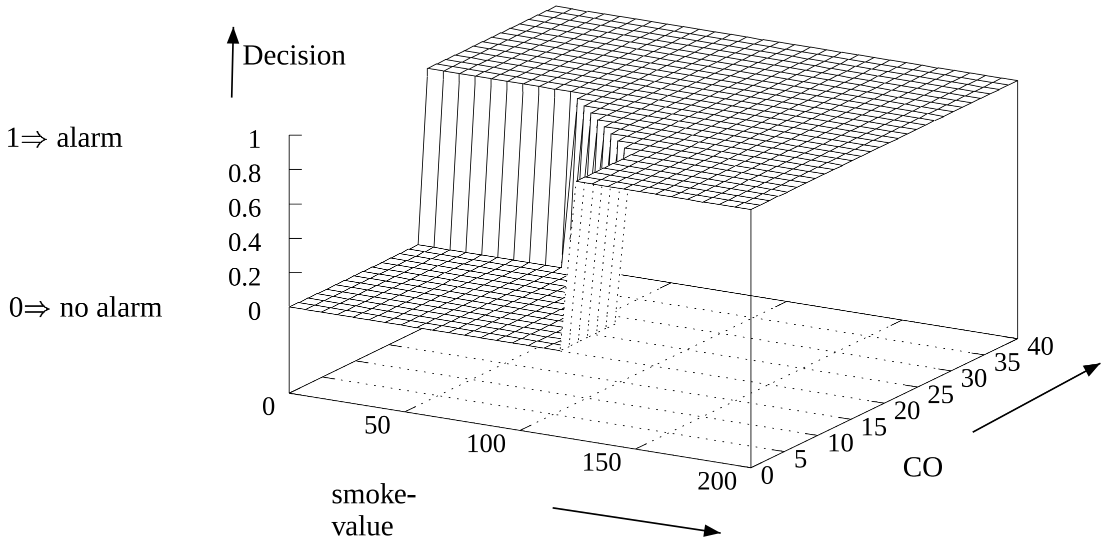

Figure 7: Decision function plotted above the observation space which corresponds to figure 3 or equation (3), respectively.

Thus, the output of a neural net defines an almost binary function above the observation space. For the simple example according to equation (3) and figure 3 the corresponding neural net output function is shown in figure 7. The precise modeling of this simple decision rule (equation (3)) requires 2 input nodes, 3 neurons in the first layer and 1 neuron in the second layer, which is the output layer in this case. Moreover, 9+4 weight factors are necessary which indicates the considerable effort. For this reason the rule-based solution is preferable in such simple cases.

3.2 Advantages and shortcomings of neural nets.

At first glance a neural net is attractive from the point of view that even in the case of observation- or feature-vectors of high dimension and perhaps confusing rules the neural net learns the underlying rules during a training phase. This requires to assign in advance the correct (or required) decision to any of the input vectors of the training set by a teacher.

All weights $w_{i,j}$ are initialized by small random values. During the training phase the input vectors are randomly selected from the complete set and each corresponding output is calculated. From the difference between the calculated output and the required output all weights in the net are updated (with the so called "error back-propagation algorithm" for example) such that **the mean square error** for all input vectors tends toward a minimum.

It has been shown that a trained neural net for certain assumptions classifies precisely according to the decision rule in equation (1) (see for example [3], p.14).

The disadvantages are:

1. the computational effort is enormous in comparison with other classifiers, not only during the training phase (which can be carried out off-line) but also during the classification phase,
2. the training procedure does not guarantee an optimal solution, i.e. the training phase must perhaps be repeated several times with different weight initializations,
3. only vague statements about the necessary number of neurons in different layers are possible,
4. the final classification performance depends very much on the correct selection of the observation vectors and even on the particular selection sequences during the training phase,
5. high partial errors may occur (while classifying single input vectors) although the mean square error indicates a good overall classification result. This might lead to false or missed alarms,
6. and last but not least - it is almost impossible to comprehend this classifying method in detail even if it works properly, because we do not know the classification rules after the training phase.

Some additional remarks concerning item 3 and the following:

Item 3.:
3.1: n-neurons in the first layer and 1 neuron in the output layer (Two-Layer-Perceptron):

Each neuron in the first layer subdivides the (2D)-observation space by a straight line, the (3D)-observation space by a plane and the (mD)-observation space by a hyper-plane, the inclination of which is determined by the corresponding weights $w_{i,j}$. With such a structure it is possible to separate open or simple enclosed regions in the observation space. The number of neurons in the first layer determine the details of the separated regions which are separated by polygons (see for example figure. 7 with 3 neurons in the first layer). The output neuron carries out some sort of logical OR operation which results in a final decision region that is the inclusion of all partial regions formed in the first layer.
This statement is true for sigmoid-nonlinearities with almost hard-limiting characteristic. For soft-limiting characteristics, the edges of the bounding polygons of the separated regions are smoothed.

3.2: Three neural layers with one neuron in the output layer (Three-Layer-Perceptron):

With a 3-layer structure it is possible to form arbitrary complex decision regions even with separated enclosed regions belonging to the same class.
Hence, more than 3 neuron layers are not necessary to solve an arbitrary complicated classification problem but it is not prohibited to use more than 3 layers.

Item 4.: The aggregate of observation vectors during the training phase.

One fundamental requirement to achieve a safe classification is, that the observation-vectors in the training set are representative for all possibly occuring fire situations and scattered over the whole observation-space during the learning phase, i.e. many **different** observation-vectors and their assigned class (fire or no-fire) must form the aggregate of the training set. Special attention must be paid to the subset of training vectors which cannot be assigned definitely to one or the other class or those vectors which point to approximately the same region in the observation space with contradictory meanings assigned by the teacher. The final classification of the trained network for contradictory training vectors is with higher probability in favour to those vectors, which appear more often in the training set with the same assigned meaning.
Contradictory training vectors can be avoided if the teacher observes the following rule in assigning the required class, which was already mentioned as part of the “Bayes”-decision rule in equation (1).

Is $q_0 \cdot K_\alpha \quad \substack{>\\<} \quad q_1 \cdot K_\beta$?

Here q_0 denotes the probability of occurrence for such a vector during a no-fire situation and $q_1 = 1 - q_0$ the probability of occurrence for a similar vector during a fire situation. K_α and K_β are the corresponding cost-factors for wrong decisions; i.e. K_α the cost-factor for a false alarm and K_β the cost-factor for a missed alarm.
The required class must be assigned to the hypothesis with the higher factor.

Moreover it is of importance to select the training-vectors at random and uniformly distributed from the complete training-vector aggregate.

Item 5.: Average classification-error and partial-errors.

The learning-progress during the training-phase is indicated by a decreasing mean square classification-error which is calculated according to the following sum:

$$\bar{F} \;=\; \frac{1}{N}\sum_{i=1}^{N}(o_{(i)} - t_{(i)})^2 \qquad (7)$$

or, more simply, by a low-pass filtered sequence of the squared partial errors $o_{(i)} - t_{(i)}$.
Here, N denotes the total number of training-vectors, $o_{(i)}$ the classification value of the output neuron for the i-th training vector and $t_{(i)}$ the corresponding correct or required output value assigned by the teacher.

$\bar{F} = 0$ can only be achieved after a sufficiently long training-phase if no conflicting vectors exist in the training-vector aggregate. Unfortunately, this does not hold for automatic

fire-detection problems otherwise false-alarms or missed alarms would not occur in practice. Consequently, the average classification error reduces during the training-phase and finally fluctuates around a minimum value >0 and the learning phase is stopped if no more learning progress can be achieved.
Even if the average classification-error is sufficiently small, some partial errors may be high and thus lead to wrong classifications of the trained network in certain situations.
Particularly, if the requirements mentioned in item 4 are neglected this may lead to missed alarms or unnecessary false alarms for 3-layer perceptrons with many neurons in the first and second layer.

For this reason a small mean square classification-error is not at all an adequate criterion. It is moreover urgently recommended to inspect closely all training-vectors with high partial-errors and their real classification after the training phase!

An adequate configuration of the neural net (number of layers and number of neurons per layer) may help to avoid such errors, though there is no other rule than: the least number of neurons and layers being sufficient and necessary to achieve a low mean square classification error.

Generally, neural nets are beyond doubt, powerful classifiers. Their use in automatic fire detection systems requires some caution, however, as shown by the remarks above - not to mention the problem of explaining with some plausibility their mode of operation to the authorities of a test institute for example.

4. Test of new designed detection-algorithms.

A performance test is an important part in the design process of a newly developed detection algorithm independent of the chosen type of detector. Personal computers in these days can be used as a powerful and rigorous test equipment. If the detection algorithm is software-implemented on the PC, it can be tested in time-lapse mode, using recorded data from fire-tests or, with artificially generated and randomly varied signal sequences for fire- and no-fire situations.

Though all manufacturers know the specific test procedures carried out by test institutes for certification, it is not recommendable to restrict the performance test to EN54-fires, for example, because in practice it is very unlikely that a fire develops according to EN54-conditions.

In our Institute, the sensor signals are recorded during the same test-fire from various sensor-heads, which are placed in different locations on the ceiling of the fire laboratory. Some of these locations are chosen arbitrarily unfavourable for detection purposes. Thus, the reproducibility conditions according to EN54 are usually not fulfilled for all sensor combinations but the recorded signals still represent fires to be detected.
Moreover, typical false alarm situations are either simulated or applied as recorded signals measured in more or less false alarm relevant environments.

We have already reported about a special test method which uses signal-models with stochastic variation of recorded signals (see for example: [4]). This method generates from a comparatively small set of recorded signals in fire- and no-fire situations arbitrarily many randomly varied test signals with similar characteristics (correlation features) than those of the recorded signals.

As a measure of quality, we compare the detection features of the newly designed algorithm either with a simple threshold detector or with the previously used detection algorithm of the manufacturer.

Summary

After some introductory remarks concerning the chances and still existing restrictions in the development of improved multi-sensor-/multi-criteria detectors, the principle solution of a detection problem has been discussed.
The task becomes more and more complicated and confusing as more sensors and/or criteria are to be taken into consideration. It was shown that feature extraction simplifies the problem and the use of recursive methods reduces the necessary computational effort.
Special attention was paid to neural nets as classifiers with the principal advantage of learning even complicated rules automatically during a training phase with the aid of a teacher. But also the shortcomings and risks associated with neural nets as classifiers have been discussed.
The complexity of multi-sensor-/multi-criteria detectors requires careful performance tests. A few experimental tests with EN54-test fires are insufficient for a rigorous test in the opinion of the author.
Test methods using computer simulations with randomly varied test sequences for fire- and no-fire situations are proposed as an alternative.

References

[1] Bayes, Th.; *An Essay Towards Solving a Problem in the Doctrine of Chances*; Phil. Trans. 53, 370-418 (1764)[1]

[2] van Trees, H.L.; *Detection, Estimation and Modulation Theory*; Part I; J. Wiley & Sons, New York 1968

[3] Lippmann, R. P.; *An Introduction to Computing with Neural Nets*; IEEE ASSP Magazine, April 1987

[4] Fischer, A., Müller, C.; *A simulation technique for the design of multi sensor based fire detection algorithms* ; Proceedings of the 10th International Conference on Automatic Fire Detection AUBE'95 in Duisburg, Germany; April 1995;
ISBN 3-930911-46-9

[1]This 236 years old work of an english clergyman, who pondered about minimizing the gambling risk is until today the basis of decision theory!