

# Neural network signal understanding for instrumentation

**Pau, L. F.; Johansen, F. S.**

*Published in:*
I E E E Transactions on Instrumentation and Measurement

*Link to article, DOI:*
10.1109/19.57233

*Publication date:*
1990

*Document Version*
Publisher's PDF, also known as Version of record

# Neural Network Signal Understanding for Instrumentation

L. F. PAU, FELLOW, IEEE, AND F. S. JOHANSEN, MEMBER, IEEE

*Abstract*—**This paper reports on the use of neural signal interpretation theory and techniques for the purpose of classifying the shapes of a set of instrumentation signals, in order to calibrate devices, diagnose anomalies, generate tuning/settings, and interpret the measurement results. Neural signal understanding research is surveyed, and the selected implementation is described with its performance in terms of correct classification rates and robustness to noise. Formal results on neural net training time and sensitivity to weights are given. A theory for neural control is given using functional link nets and an explanation technique is designed to help neural signal understanding. The results of this are compared to those of a knowledge-based signal interpretation system within the context of the same specific instrument and data.**

*Keywords*—**Neural understanding, calibration, signal understanding, control theory, neural control, training time, sensitivity to noise, explanation facilities, knowledge-based signal interpretation, instrumentation, analytical instrumentation.**

## I. Introduction

IN THE FOLLOWING, the term "neural signal understanding" is defined as using neural processing for signal understanding tasks, such as signal classification and interpretation. It does not cover neural signals as related to the neurons. For an introduction to neural processing, the reader is referred to introductory references such as [1]–[3].

This paper reports on a project involving neural processing for the purpose of classifying the shapes of a set of instrumentation signals $y_i(t)$, $i = 1, \cdots, N$, in order to calibrate a device, to diagnose anomalies or wrong settings, to generate appropriate tuning settings, and to interpret the measurement results; all together there are $c = 1, \cdots, C$ classes.

The sensors in the instrument are calibrated by exposing it to a standardized but controlled environment, with control parameters $U(t) = \{u_i(t), i = 1, \cdots, p\}$ (e.g., flow, gain, bandwidth, delay, etc.). The output of the calibration runs are $N$ 1-D signals $y_i(t)$; $i = 1, \cdots, N$ depicting the variations over time of a number of heterogeneous responses.

In instrument operations, likewise, signals $y_i(t)$ are recorded in the same way for given $U(t)$, except that the sensors, the environment, and the samples can be different or unknown.

Manuscript received June 22, 1989; revised March 24, 1990.

The authors are with the AI and Vision Group, Technical University of Denmark, DK 2800, Lyngby, Denmark.

IEEE Log Number 9036459.

Until recently, a skilled operator was needed to interpret the shapes of all $N$ signals jointly to relate them to changes to be made in the controls $U(t)$, and to past behavior/settings of the sensors and electronics. These signal shape changes were supplemented by a diagnosis (hypothesize-then-test) of the functional faults, which were either of the sensor and/or of the set-up.

Three approaches are possible [4]:

1) signal feature extraction, and neural processing thereof, with training by known calibration and test data, and later neural classification [9];
2) syntactic pattern recognition of the curve shapes, on the basis of formal symbolic grammars, and analysis of the parsed symbol strings by diagnostic knowledge based rules [5], [6];
3) identification of a dynamic model of the ($U(t)$, $y_i(t)$) relation, in a multi-model framework, with each model corresponding to a separate functional behavior of the whole instrument [7], [8]; one such approach is by hidden Markov models applicable to well segmented signal portions; two subclasses coexist here, one where the in-built control laws in the instrument are known, and one where they are not.

Approach 3) did not seem to work in general unless the signals could be segmented (as is the case in phoneme recognition, but different from the approach of Section III). Therefore, we will report mainly about approach 1) (Sections II–V), and compare it briefly with 2), which was also implemented for exactly the same practical case (see Section VI).

The paper is organized as follows. Neural signal understanding, as related to signal processing and the required neural network architectures, is surveyed in Section II. The main contribution of the paper is in Section III, where a real case of neural signal understanding for instrumentation is defined, results are given, and a formal result is presented about the training convergence speed of a feedforward neural network. Section IV shows how the signal classification results can serve for instrument actuator control, in relation to the functional link net architecture. Users of the system have required added explanation facilities akin to those of expert systems, and the implementation concept for this approach is presented in Section V. The same task, as that of Section III, is treated with an application specific expert system, called

0018-9456/90/0800-0558$01.00 © 1990 IEEE

KEMEX, and the comparative performances are discussed in Section VI. Conclusions are given in Section VII.

## II. Survey of Related Neural Signal Understanding Work

Neural signal understanding involves three major concerns:

1) signal knowledge representation or feature selection, to be presented to the inputs of the neural network;
2) selection of the neural network architecture;
3) adaptive control and classification considerations relating to the controls $U(t)$.

This section serves, not as much the purpose of covering neural signal understanding, but to highlight which alternative signal processing approaches are related to it and to exploit the signal representations they propose within a neural signal understanding framework.

### A. Neural Signal Representation

The four following approaches prevail (Fig. 1).

1) Signal feature selection, which means defining a battery of feature extraction functions $\varphi_j$ where $j = 1, \cdots, n$ applicable to signals:

$$\varphi_j: \left\{ \{ y_i(t), t \in [0, T] \}, i = 1, \cdots, N \right\} \rightarrow R.$$

Examples are minimum, maximum, morphomathematical signal features, first spectral component, Karhunen–Loeve expansion components, signal moments, polynomial coefficients, etc. [9], [10]. In most pattern recognition cases the features are highly application specific; there have been attempts in neural processing to identify some general feature extraction functions $\varphi_j$, e.g., for curvilinear line drawings by identifying high curvature/inflection points.

2) Serial order description [11], consisting in defining a state consisting of two elements:
   a) a vector of measurements ($Y(t) \triangleq \{ y_i(t), i = 1, \cdots, N \}$) for any given $t$.
   b) a plan $P$ for updating these vectors $Y(t)$, which are updated as

$$Y(t+1) = f(Y(t), P)$$

   where $P$ typically includes $U(t)$ but also provides scheduling choices ($P \ni U(t)$).

   In this case, there is no explicit representation of temporal order and no explicit representation of action sequences; the input nodes then receive the states ($Y(t)$, $P$). This approach is related to shape analysis of the signals.

3) An explicit signal representation, where there is one input node attached to each signal sample $y_i(t)$, leading to a total of $n = N \times T$ input nodes [12], [13].

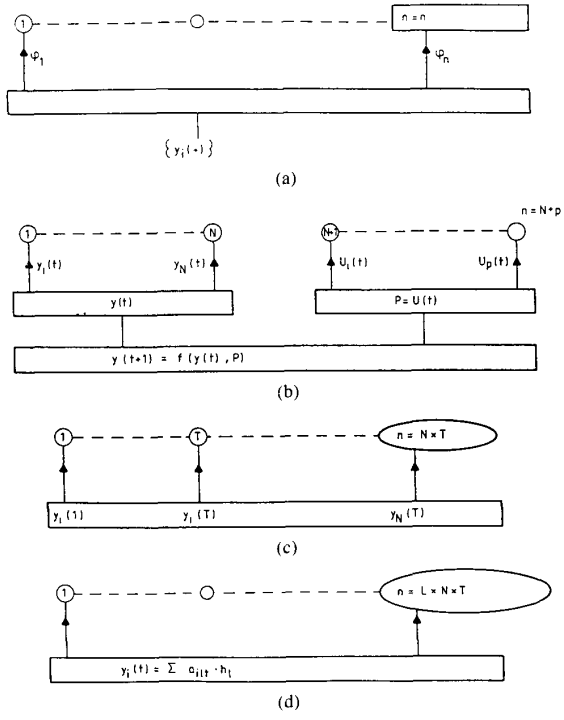


Fig. 1. Neural signal representation. (a) Signal feature selection. (b) Serial order description. (c) Explicit signal representation. (d) Functional signal expansion.

4) A functional signal expansion, where all signals $y_i(\cdot)$ are series expanded in terms of a basis of known kernel functions $h_\ell$ where $\ell = 1, \cdots, L$

$$y_i(t) = \sum_{\ell=1,L} a_{i\ell t} h_\ell.$$

Examples are: polynomial expansions (Legendre, Hermite), Walsh functions, sinusoidal functions, Fourier functions, etc. [1], [14]. The signal is then represented through $n = L \times N \times T$ coefficients $\{a_{i\ell t}\}$.

Later in the paper, we assume the default signal representation to be 3); changes can readily be made for further developments to accommodate other representations instead.

### B. Neural Signal Understanding Network Architecture

The selection of the network architecture is, together with the signal representation, paramount for its learning capabilities, convergence, etc.

Most work refers to the back-propagation algorithm [2], [3], [15] operating on a perceptron with hidden layers, $z_k$, $k = 1, \cdots, \lambda$ where the weights $\{w_{\ell m}\}$ control adaptation formulas inserted at each hidden layer node; $w_{\ell m}$ represents the weight applicable to the link between neurons $\ell$ and $m$. The input elements are usually kept as linear elements in order to provide for a good dynamic range, while sigmoid elements are used for hidden or output nodes. Sometimes the output nodes are also linear.

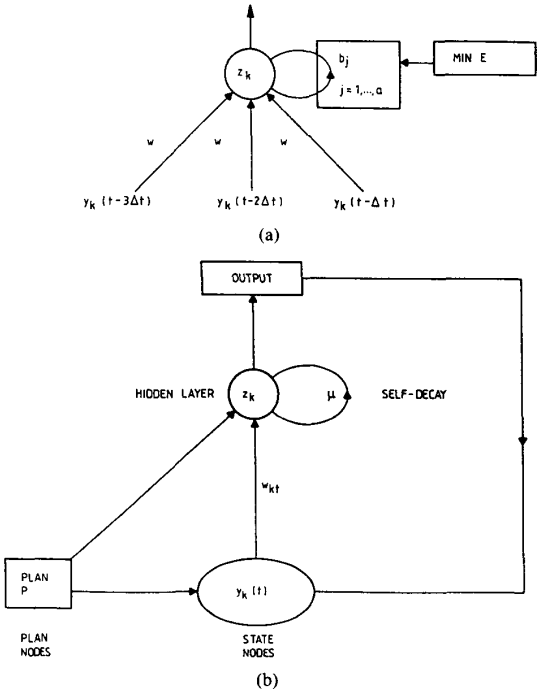


Fig. 2. Neural signal understanding network architectures. (a) Least mean square adaptation. (b) Recurrent network (delta rule) with self-decay.

Typical adaption formulas inside the hidden nodes (Figs. 2 and 3) are shown in the following.

1) Least mean squares/adaline [13]:

$$z_k = z_{k0} + \sum_{j=1}^{a(k)} w_{kt}\, y_k(t - j \cdot \Delta t)$$

where $z_{k0}$ is the activation threshold (the same value for all neurons of a given layer), and $w_{kt}$ are determined as to minimize the expected error in the least square sense:

$$E = \int \sum_k \left( y_k(t) - \sum_{j=1}^{a(k)} w_{kt}\, y_k \cdot (t - j \cdot \Delta t) - z_{k0} \right)^2 \cdot dt.$$

Here, $j$ designates the neurons in the successor/predecessor layer to layer $k$, and $a(k)$ is the number of such neurons. The time integral is for time averaging of the least square error, if appropriate.

2) Generalized delta rule with self-decay [11], e.g., with a logistic function

$$z_k = (\max - \min) \cdot f(y_k(t)) + \min$$

where min is the minimum value attained by the logistic function (and, therefore, the minimum value of activation that any unit can have), and max is the maximum value, or with an explicit decay weight $((\mu/w) < 1, \mu < 1)$:

$$z_k(t) = \mu z_k(t - 1) + w_{kt}\, y_k(t).$$

A variant is found in [16].

3) Functional link network [1], with the gradient minimization operating on the weights applicable to different time-slices $[T_1, T_2]$ and signals $y_j(\cdot)$, thus emphasizing some more or less according to the magnitude of the weights.

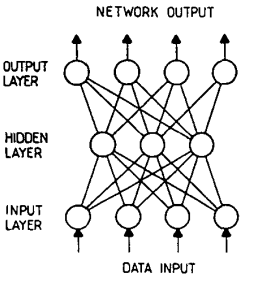


Fig. 3. A three-layer feedforward neural network.

## III. Neural Signal Understanding for Instrumentation

For the task defined in Section I, the following approach was selected in one specific instance of analytical instrumentation [17]. In the literature, cases of neural signal instrumentation are very rare, and those that do occur relate mostly to sensor design [20]. Further details are omitted for industrial confidentiality reasons.

### A. Signal Classes

The $C(C = 9)$ signal classes were taught by labelling observed evolutions $\{ y_i(t),\ i = 1, \cdots, N,\ t = 1, \cdots, T)$ by their class labels and $U(\cdot)$ plan/control labels. The issue of the selection of training examples was complicated by the fact that more than one label was often assigned to these samples, making further post-processing of the neural classifications necessary [18].

### B. Neural Signal Representation

A signal feature selection was determined as follows. (See Section II-B1)

1) The $N$ signals $y_i(t)$ were first passed through a second order derivative filter in order to find points of inflection and curvature changes.
2) The ordered lists $\varphi_j$ of such positions, and their number, were then entered into the network.

Additional similar evolved procedures for curve feature extraction involve acquiring the discrete tangent field by taking equidistant curve slices and estimating the tangents and curvature at the slice end points on the curves [17]. Orientation discontinuities appear as multiple tangents at a single position, with no curvature estimate. Splines can then be matched to these intersection points.

### C. Neural Signal Understanding Network Architecture

A layered feedforward network with the back propagation training rule [2], [3], [15] without decay (Section II.B.2)) was used and implemented in the self-developed NETSIM package (written in C) [12].

### D. Neural Classification Performance

Six series of experiments were performed:

1) calibrated signal data, $n_0 = 20$ input nodes, $n_1 = 20$ hidden nodes in one hidden layer $\lambda = 1$, and $C = 6$ output nodes;
2) same as 1), but with one additional hidden layer with $n_2 = 20$ nodes;
3) noncalibrated signal data, otherwise as in 1);
4) calibrated signal data, $n_0 = 15$ input nodes, one hidden layer $\lambda = 1$ of $n_1 = 15$ nodes, and $C = 6$ output nodes;
5) additive noise applied to the calibrated signal data 1):

$$y_j'(t) = y_j(t) + \beta \cdot n(t)$$

where $\beta$ is a constant, and $n(t)$ white noise in the interval $[-1, +1]$;
6) multiplicative noise applied to the calibrated signal data 1):

$$y_j''(t) = \left(1 + \alpha \cdot n(t)\right) y_j(t).$$

These results were obtained by leave-one-out testing of the entire training set. The results [12] yield the following percentages of correct classification results with the number of training steps (in parenthesis) for one specific training algorithm implementation:

1) 65% (127); 4) 40% (71);
2) 60% (93); 5) as 1) with $\beta < 0.0369$ (156);
3) 55% (248); 6) as 1) with $\alpha < 0.1777$ (311).

It also appears that calibrated training signals are less sensitive to multiplicative noise than noncalibrated ones, whereas the latter are less sensitive to additive noise.

The same experiments were carried out by varying the number of input nodes $n_0$, with hidden nodes in each layer ($n_1$ in 1), $n_2$ in 2)). The performances changed but the same overall comparative pattern as the one reported above prevailed, provided more than 5 nodes were used each time.

### E. Training Time and Network Sensitivity

A formal study of the convergence of the selected three-layer feedforward network was established [12], leading to the following result, to be justified in a subsequent paper.

*Hypotheses:*

$(\lambda + 1)$ number of layers,
$n_j$ number of neurons in layer $j$, where $j = 0$ is the input layer; when $j$ increases, $a(j) = n_{(j-1)}$,
$O_j$ output of neuron $j$,
$I_j$ input to neuron $j$, equal to $O_j$ in layer $j = 0$, otherwise equal to 0,
$w_{\ell k}$ weight applicable to the link between neurons $\ell$ and $k$,
$a_1 = z_{\ell 0} + \Sigma\, O_j * w_{j\ell}$, $j = 1, \cdots, n_\ell - 1$ weighted sum of the inputs to neuron $\ell$, having threshold value $z_{\ell 0}$, when this neuron is in layer $\ell$,
$f(x) = 1/(1 + \exp(-x))$ neuron response function being the logistic function (Fig. 4),
$\phi \triangleq 1/4$, $\delta \triangleq 4/27$ are given constants as related to bounds on the logistic neuron response function, and its derivative [12].

*Conclusions:*

1) The sensitivity of the network output $O_i$ in output layer $\lambda$, to changes in the input values $I_j$ in layer $k = 0$, is bounded by

$$\frac{\Delta O_i}{\Delta I_j} \leq \phi^{\lambda} * \sum_{k_n}^{N_\lambda - 1} \left( w_{k_n i} * \sum_{k_{n-1}}^{N_\lambda - 1} \left( w_{k_{n-1} l_n} * (\cdots) \right) \right).$$

2) The sensitivity of the network output $O_i$ in layer $\lambda$, to changes in the weight $w_{jk}$ between nodes at two successive layers $(\ell - 1)$ and $(\ell)$ is bounded by

$$\frac{\Delta O_i}{\Delta w_{jk}} \leq \phi^{(\lambda - l + 1)} * \left( \sum_{l_n}^{N_{\lambda - 1}} w_{l_n i} * \left( \sum_{l_{n-1}}^{N_{\lambda - 2}} \left( w_{l_{n-1}} * \cdots \right) \right) \right).$$

3) The training time $\tau$ is proportional to

$$\tau \sim 1/(\text{incr.}\ \upsilon^{1/2} \cdot D_{\min}^{1/\lambda})$$

where

$\upsilon \triangleq E(n_j)$ average number of neurons in a layer,
$D_{\min}$ minimum Euclidian distance between any two training points $y_j(t)$ belonging to two different classes among the $C$ possible,
incr. $\triangleq E(\Delta w)$ average weight change between two iterations.

Reference [12] shows a good correlation of this curve with real training time data.

*Interpretation:*

The two sensitivity results 1) and 2) indicate what changes can be expected on the network output neurons vs. changes in the input neurons and weights. Result 1) can be used to determine the number ($\lambda$) of layers to alleviate noise in the input neurons, while not affecting the output neurons by more than some threshold values preserving the classification results. Result 2) immediately gives a criterion to heuristically stop the training algo-

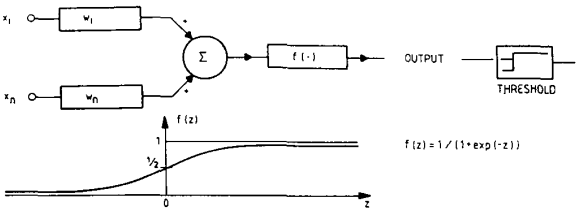


Fig. 4. Neural processing element.

rithm (thus speed up the results), as soon as the sensitivity of the output neurons to changes in the weights is less than the classification thresholds on their values. Result 3) allows, in the case of repetitive tests with one given architecture, to curtail the training or to extrapolate its value in case of changes, e.g., in the separability measure $D_{\min}$ between classes.

## IV. Neural Control by Functional Link Nets

### A. Introduction

As mentioned in Section I, the instrument operator will, after calibration, trim some controls $U(t)$ to achieve desired performances while also complying with specific sensor and electronics behavior. Examples of such controls $U(t)$ are gain, offset, and flow of titrant/solvent.

In this section we present the formal process of synthesizing these controls $U(t)$ by neural processing, assuming the net architecture to be the functional link net [1]. These results are original and of a general nature; therefore, the instrumentation application will not be emphasized in the remainder of this section.

As highlighted in Fig. 5, the back propagation net of Section III is used for classification, while all neural control functions require the functional net architecture. Thus there is no correspondence exploited here between them; on the contrary, different nets are shown here to be coexisting, and the classification net could be replaced by another classification approach.

### B. Functional Link Nets

Functional link nets [1] are essentially characterized by the following:

1) a single layer of input neurons;
2) a single aggregation node with a nonlinear response;
3) the expansion of the input layer by functional links which are kernel functions of selected components of the input vector $X(U, t)$; as a result the $N$ nodes of the input layer are the $N$ functional links $f_i(X(U, t)), I = 1, \cdots, N$ (including eventually $f_i = x_i(U, t)$, the $i$th component of the vector $X$);
4) the back propagation search rule which updates linear weights $w_i$, $i = 1, \cdots, N$.

Whereas functional nets allow for supervised learning, unsupervised learning, as well as classification, they provide a formal framework for the synthesis of controls while also encompassing key notions from traditional

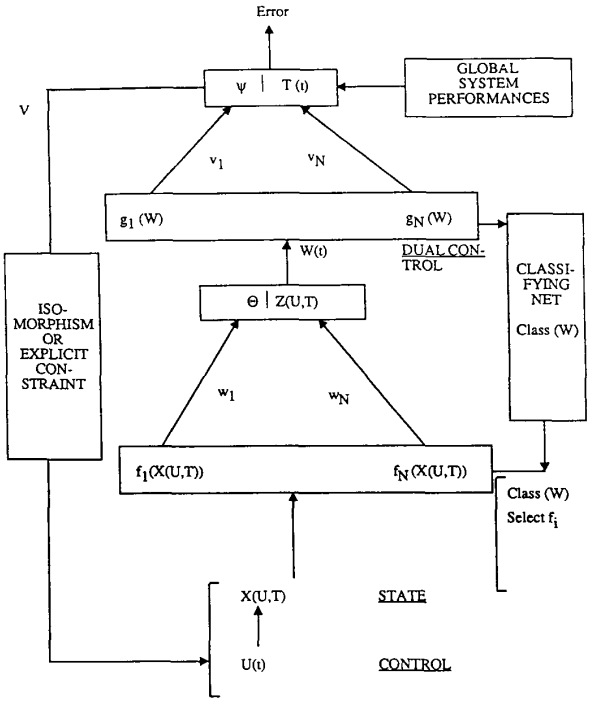


Fig. 5. Neural control by functional link net.

control theory. This gives the capability of reusing neural processing hardware and firmware, not only for perceptual or learning tasks, but also to replace programmable logic controllers and other control devices (proportional integrated derivative (PID) controls, etc.).

### C. Neural Control Theory

Consider a functional net, with inputs $X(U, t)$ subject to controls $U(t)$, and a flat net with $N$ functional links $f_i(X)$ (see Fig. 5).

The target $z(U, t)$ and threshold $\theta$ require that the linear weights $w_i$ in the net be such that:

$$w_1 f_1(X(U, t)) + \cdots + w_N f_N(X(U, t)) = z(U, t) - \theta.$$

This defines the vector $W(t) = [w_1 \cdots w_N\ 1]$ as orthogonal to a manifold which is subject to the controls $U(t)$. As a result, in line with identification theory, $W \triangleq [w_1 \cdots w_N\ 1]$ is a dual control $W(t)$, or point in the $w$-control space which is dual to the $U$-control space.

However, the neural learning schemes, which determine $W = [w_1 \cdots w_N\ 1]$, such as a gradient search or in some cases a recursive pseudo-inverse calculation, are of much wider applicability in practice than optimal control schemes, which also suffer from high numerical complexity.

The neural control functional net will in turn operate on functional links $g_1, \cdots, g_N$ applying to the input $W(t)$, with weights $v_1, \cdots, v_N$:

$$v_1 g_1(W(t)) + \cdots + v_N g_N(W(t)) = T(t) - \psi$$

where $T(t)$ is the target. Typically this target will deal with global performance criteria of the overall system, such as:

1) response time to meet the target $z(U, t)$;
2) energy level related to the control energy $\int U(t)\,dt$.

Being a dual to $W(t)$, the weights $V(t) = [v_1, \cdots, v_N 1]$ will belong to a space isomorphic with the original functional space $f(X(U, t))$, of which $V(t) = U(t)$ is a special case.

This formalism, opens up for three capabilities.

1) By classifying the vectors $W(t)$ over time, typically with fixed discriminants trained by a dedicated supervised net, makes it possible to *select* those functional link groups (i.e., subsets of $f_1, \cdots, f_N$) achieving the desired properties of the system that are chosen via the classification net training of the pairs ($W$, class ($W$)), where class ($W$) is the result of the classification.

2) The dual net $V$ will in fact enforce on the learning net $W$, long term properties over time, or measured value in terms of the error in the position $X(U, t)$ itself, by assigning the weights $V$ to belong to fixed domains in the $X$-space or $U$-space.

3) Temporal properties will be explicitly available by tracking the changes over time of $W(t)$.

## V. Explanation Facilities to Neural Signal Understanding

For the calibration, set-up, diagnostic and measurement interpretation tasks, the instrument operator almost invariably wants a support system to justify its advice by explanations such as why? how? why not? Unfortunately, such facilities do not exist in neural processing. We therefore describe the formal and practical solution developed to address this issue [19].

1) All inputs $X(U, t)$ are scaled to belong to the range $[-1, +1]$.

2) Our neural explanation facility involves asserting as Prolog language facts the weights $w_i$, $i = 1, \cdots, N$ (obtained after convergence), as well as the corresponding signal classes $N(X(U, t)) = c$, $c = 1, \cdots, C$.

3) A sorting utility ranks the weights $w_i$, $i = 1, \cdots, N$ by decreasing values of $|w_i|$, which are eventually signed.

4) A sorting utility ranks the classes $c$ by decreasing frequencies in both the learning and operational neural signal interpretation modes, for given weights $w_i$, $i = \cdots, N$.

5) Implement (in Prolog) standard back tracking or forward chaining explanation facilities operating on the sorted fact bases 2) and 3) of hypothesis and goals, respectively.

The result is to allow the selection of "why" explanations, which are those neural input nodes (meaning those inputs $x_i(U, (t))$ or functional links $f_i$), which have the largest weights and thus contributions to the output neuron responses. Likewise, the "how" explanations are those classes for which, given the current weights, the classification frequencies will be the highest. Of course, in both cases, text generation in Prolog allows us to relate text to these two explanations described. By selecting the number of items of highest ranks, the user can request rough, meaningful, or comprehensive explanations. Furthermore, a knowledge base may filter out the alternatives to eliminate nonsignificant explanations in causality chains and to generate the corresponding root explanations in such chains.

## VI. Comparison with Knowledge-Based Instrumentation Signal Interpretation

Fortunately, the neural signal understanding set-up described in Section III for a specific case could be compared with a knowledge-based signal interpretation system working with the same instrument and data [17].

The major differences between the two solutions, in the case of the knowledge-based system were:

1) a signal interpretation inference procedure, based on conceptual graphs, and implemented in Prolog;
2) an intelligent user interface implemented in Smalltalk language, involving logbooks, application notes, and sensor behavior monitoring;
3) a signal feature extraction procedure (similar to Section III-B), implemented in Pascal;
4) an explanation facility, implemented in Smalltalk and Prolog.

The knowledge-based signal classification performances, estimated on the same data as in Section III.D., was a minimum of 92% correct classification on calibrated or noncalibrated data.

The knowledge acquisition time was close to 6 man months, compared to a few days by the techniques in Section III-D.

The knowledge based signal interpretation sensitivity to additive or multiplicative noise (see Section III-D.) was:

$$e = 74\% \qquad f = 56\%$$

which indicates a better robustness of the neural signal interpretation procedure, for the available knowledge.

Regarding generating controls $U(t)$, of which two were stationary and a third was a linear time dependent function, the accuracy achieved by the approach of Section IV-C. was higher than that of the knowledge based approach which had a tendency to give only class-related settings (which were not measurement related).

In terms of explanation facilities, the knowledge-based system was clearly superior in terms of accuracy and depth of the justifications; however, the neural signal explanation facilities of Section V, executed with the first five ranked candidates, were quite sufficient for all practical purposes.

## VII. Conclusion

Neural signal understanding for instrumentation is a promising technology, especially when the sensors or

processes drift or have memory. Its performance is less than the knowledge-based signal interpretation approach, which however assumes an extensive knowledge acquisition process and is not always practical. The neural training time is still far too long using hardware which is usually implemented in or adjacent to instruments. Improved signal feature extraction procedures, possibly combining knowledge-based and neural processing and high-speed coprocessor boards, are key to better overall performances.

Depending on operations of the instrument, this may be a minor problem since the neural network only needs to be trained once for each operational setting.

## References


[1] Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*. Reading, MA: Addison Wesley, 1989.

[2] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, pp. 4–22, Apr. 1987.

[3] D. E. Rumelhart, J. L. McClelland, Eds., *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*. Cambridge, MA: MIT Press, 1986.

[4] L. F. Pau, "Intelligent man-machine interfaces in instrumentation," *Proc. 1988 Fin. Artif. Intel. Conf.*, Univ. Helsinki, vol. 1, pp. 12–19; also in *J. Jap. Artif. Intel. Soc.*, vol. 9, no. 4, pp. 8–24, Dec. 1989.

[5] K. S. Fu, *Syntactic Pattern Recognition*. New York: Academic, 1975.

[6] K. S. Fu, Ed., *Application of Pattern Recognition*. Boca Raton, FL: CRC Press, 1980, Ch. 6.

[7] B. Widrow and S. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1985.

[8] H. W. Sorensen, Ed., *Kalman Filtering*. New York: IEEE Press, 1985.

[9] L. F. Pau, *Failure Diagnosis and Performance Monitoring*. New York: Marcel Dekker, 1981.

[10] C. H. Chen, Ed., *Digital Waveform Processing and Recognition*. Boca Raton, FL: CRC Press, 1982.

[11] M. I. Jordan, "Serial order: a parallel distributed processing approach," *ICS Report 8604*, Inst. for Cognitive Science, Univ. California, San Diego, La Jolla, CA, May 1986.

[12] F. S. Johansen, "Neural signal tolkning (in Danish)," M.S. thesis, (EMI) Tech. Univ. Denmark, Aug. 1988.

[13] A. Lapedes and R. Farber, "Nonlinear signal processing using neural networks: Prediction and system modelling," T. R. Los Alamos Nat. Lab., Los Alamos, NM, July 1987.

[14] D. Gabor *et al.*, *Proc. Inst. Elec. Eng.*, vol. 1088, July 1960.

[15] W. P. Jones and J. Hoskins, "Back propagation: A generalized delta learning rule," *BYTE*, pp. 155–162, Oct. 1987.

[16] K. Fukushima, "A neural network for visual pattern recognition," *IEEE Comput.*, vol. 21, Mar. 1988.

[17] *KEMEX*, Technical report, Industry and Technology Agency, Copenhagen, Denmark, Nov. 1988.

[18] S. I. Gallant, "Connectionist expert systems," *Commun. Ass. Comput. Mach.*, vol. 31, no. 2, pp. 152–170, Feb. 1988.

[19] L. F. Pau and T. Goetzsche, "An explanation facility for neural networks," T. R. EMI R-416, Tech. Univ. Denmark, Lyngby, Dec. 1989.