A General Method For The Discovery And Use Of Rules Induced By Feedforward Artificial Neural Networks

Robert F Harrison and Zhe Ma

Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK.

R Lee Kennedy

Department of Medicine, The University of Edinburgh, Edinburgh, UK.

10 November 1995

Research Report No. 607

Key Words: Rule Extraction, Hybrid Knowledge-based Systems, Neural Networks, Rule Validation.

Running Header: Rule extraction from neural networks.

Abstract

GR2 is a hybrid connectionist / knowledge-based system consisting of a Multi-layer Perceptron and a rule-based system for knowledge representation and reasoning.

Knowledge embedded in a trained Multi-layer Perceptron is extracted in the form of general (production) rules—a natural format for abstract knowledge representation. The rule extraction method integrates black-box and white-box techniques on the MLP, obtaining feature salient and statistical properties of the training pattern set. This is achieved via a heuristic based on the static connection strengths from input to output.

The extracted general rules are quantified and selected in a rule validation process. Multiple inference modalities such as categorical reasoning, probabilistic reasoning and exceptional reasoning can be performed in GR2. In addition, quantitative indications of a rule's validity within the domain and the importance of any antecedent within a rule can be calculated.

Experiments are conducted in artificial (simple logic) domains and using data from emergency medicine. The predictive performance of the underlying neural networks is seen to be maintained whilst a valid set of rules is extracted. For the medical problem, favourable comparison is drawn with the C4.5 technique, an extension of the celebrated ID3 algorithm.

The methodology can be applied to any feedforward neural network via straightforward extensions to the basic ideas and avoids the need for specialised architectures found in some other methods.

1. Introduction

For an "intelligent" decision support system to gain acceptance by a target user community it must be

- useful, i.e. there should be some perceived benefit from introducing the device;
- usable, i.e. its use should not interfere unduly in the way a practitioner wishes to work (unless the benefits are so great that they warrant the introduction of new working practices);
- valid, i.e. the information it provides should be "correct" according to some measure such as statistical accuracy;
- verifiable, i.e. the end user should be able to understand how it represents the problem domain and how its conclusions are reached—it should be able to explain its reasoning.

This paper is concerned with the last of these requirements from the point of view of the most widely adopted artificial neural network—the multi-layer Perceptron (MLP) (Rumelhart et al., 1986). It addresses the problem of extracting a set of general (production rules) from an MLP, trained as a one from many classifier, and from its set of training data. Expert systems, by their adoption of a symbolic representation of knowledge, provide a natural framework for verification. Indeed, verification of the rule-base by domain experts plus a built-in capacity to explain how a particular conclusion is reached is regarded as a pre-requisite for good expert system design. However, in many fields the rules governing decision making are obscure and the problems of eliciting them from experts are well known. This may lead to a rule-base which is incomplete, contradictory and inaccurate, compromising both performance in the field and the client group's confidence in the technology. Furthermore, when uncertainties are present, either in the rules or in the data, or in both, serious difficulties are encountered in the reasoning process. These are addressed by a number of techniques (e.g. certainty factors, Dempster-Shafer theory (Gordon and Shortliffe, 1985; Shortliffe, 1976)) which rely on more or less plausible assumptions—the more plausible the assumptions, the more difficult it appears to be to implement the techniques in a practical system.

To overcome these difficulties, it would seem that rule-based systems might benefit greatly from an ability to adapt to actual practice so that any shortcomings in the rule-base would be rectified on the basis of experience (Langlotz et al., 1986), however, such systems are almost

always static and modifications to existing rules or the addition of new ones usually requires major intervention by a knowledge engineer.

In data-rich environments, automatic induction of symbolic rules from a set of exemplars may be appropriate. This overcomes the knowledge acquisition problem, replacing it with one of high quality data harvesting instead. Again the knowledge representation is symbolic and the inference process is based on production rules. Furthermore the question of reasoning with uncertainty may be ameliorated if the induction process is based upon probabilistic arguments as exemplified in the celebrated ID3 algorithm (Quinlan, 1986) which has been used successfully in many machine learning applications. Again ID3 suffers from the disadvantage of being static—new information can only be encoded by considering all previous categorisations as well.

Connectionist systems are also capable of performing inductive learning. They operate with symbolic or sub-symbolic data but knowledge is represented within them not as a set of production rules but by the interplay of the numerical value of their parameters and the topology of their interconnections. Thus, although ANNs can be trained to perform well from an external (input/output) viewpoint, their operation remains obscure. It is this "black-box" nature which may make potential users resistant to the use of ANN-based decision aids especially in safety critical applications or where there is significant risk of litigation (Hart and Wyatt, 1990). Nonetheless, ANNs have many attractive properties, the most important of which for the MLP are:

- it can approximate any "reasonable" function with arbitrary accuracy (Cybenko, 1989) with only one layer of hidden units (this is not to say that a more parsimonious representation might not be forthcoming if multiple hidden layers were to be used);
- in one-from-many statistical classification tasks and for an optimal set of weights, it approximates the probability of class membership conditioned on the data (the posterior probability) directly (Richard and Lippman, 1991), without having to make the often unwarranted assumptions about underlying probability structures which are usual in the development of Bayesian classification systems, e.g. independence of features.

From this we can conclude that given enough high quality data and sufficient time and computing resources it should always be possible to derive the Bayes optimal classifier for

some (stationary) domain. Given that this is the best that can be done from a statistical viewpoint, the potential utility of the MLP is clear[1].

Neural networks, with the notable exception of the adaptive resonance theory family of architectures (Carpenter and Grossberg, 1987), are almost always static in the sense that the knowledge embedded within them is learned once and subsequent learning is suppressed. They therefore suffer the same disadvantage mentioned earlier in the context of ID3. However, re-training such systems is a relatively simple matter when compared with modification and verification of an existing rule-base in an expert system. ANNs and rule-based systems (RBS) represent knowledge and intelligence at different levels (Fu, 1994a) and integration of the two types of system promises to overcome their individual shortcomings. ANNs can automatically learn underlying properties from the original domain data, but it is difficult to reveal explicitly what knowledge they have acquired. Conversely, most RBS use rules of inference based on high-level knowledge representations sympathetic to human interpretation, but suffer from the knowledge acquisition bottleneck. It seems then that successful integration of ANNs and RBSs may lead to completely automatic processing throughout the life-cycle of knowledge engineering. Extraction of rules from the ANN for use by the RBS is the essential step in the integration of the two systems.

A desirable characteristic of rule extraction is that the rules from the ANN should be:

- capable of interpretation by the RBS;
- accurate and complete with respect to what the ANN has acquired;
- efficient for the RBS to use.

A number of techniques aimed at deriving such a set of rules has been proposed in the past. Some of these are reviewed below.

1.1 Review of Earlier Work

Rule extraction from ANNs can be categorised into "black-box" and "white-box" approaches. The former ignores the internal structure of the network and generates rules referring only to

[1] Other feed-forward networks have these properties also as, of course, do many methods from non-linear, multivariate regression theory. The modification of the techniques described here to such cases is straightforward.

the correspondence of input and output values. The latter identifies rules according to the internal connection structure of the network.

Saito and Nakano extract rules from a trained MLP based on the correlation of inputs and outputs, as the MLP is fed with input vectors while the bits are switched (Saito and Nakano, 1990). The combinatorial problem occurs and the thrust of their work lies in trying to relieve this by limiting the switching only to positive input bits among the training patterns, and by controlling the number of input bits to be switched.

The knowledge-based artificial neural network (KBANN) system of Towell and Shavlik first encodes domain knowledge (rules) as the connections in an MLP (Towell and Shavlik, 1993). It then extracts the refined rules from the trained network. To reduce the number of combinations of rule antecedents KBANN clusters the weights into a few uniform groups during a secondary training phase, without changing the knowledge encoded. Identical weights in a group can then be represented as an N-of-M relation.

Carpenter and Tan extract rules from a self-organising supervised learning neural network—fuzzy ARTMAP (Carpenter and Tan, 1993). This is made possible owing to the localised representation within fuzzy ARTMAP so that each category node implements a rule directly. Two techniques are used: pruning nodes of low *utility* (a measure of both usage and accuracy), and quantisation and thresholding of the remaining weights.

Ridella and co-workers (Ridella et al., 1994) develop a collection of methods to generate hierarchical "if-then" rules from a multi-layer Class-Entropy Minimisation Network. The rule extraction procedure comprises cycles of partitioning, pruning and re-training. The rules are extracted from each connection layer in turn. As a consequence they are in a format of a cascaded embedding of the form "if-then-else...".

Fu presents a system—the knowledge-based connectionist neural network (KBCNN)—which constructs and revises a neural network according to rules encoded from the initial domain knowledge (Fu, 1994b). When the network performance gets stuck during training, new hidden units are added to different layers in order to generate new concepts. Rules are generated directly from the weighted connections in the trained network.

Andrews and Geva present RULEX (Andrews and Geva, 1995) which enables production rules to be extracted from a constrained error back-propagation multi-layer Perceptron by

direct interpretation of its parameters. This system also allows for the insertion of known rules enabling a priori priming with subsequent rule refinement.

Bochereau and Bourgine use a general formula called the validity domain to ease the generation of rules from an MLP (Bochereau and Bourgine, 1990), while Diederich demonstrates a connectionist semantic network having an explicit conceptual hierarchy as an explanation component to answer "how" questions (Diederich, 1990). Mahoney and Mooney describe the RAPTURE system (Mahoney and Mooney, 1993), which uses two subsystems: a modified MLP to refine the certainty factors of a MYCIN style rule, and a symbolic learning method heuristically using the ID3 information gain for the addition of new rules. McMillan and colleagues propose a system called RuleNet (McMillan et al., 1991) which extracts rules from a neural network and re-encodes them, in an iterative projection process.

Most attempts at rule extraction from ANNs have been made using the MLP or its variants because of the great success of MLP in machine learning and the fact that it is the most widely adopted ANN. However, rule extraction from MLPs has proved to be difficult owing to their non-linear behaviour and the mutual dependence of the layers.

In the black-box approaches, researchers face the combinatorial obstacle. The number of situations to test increases exponentially with domain size. Limits on the test number lead to the likelihood of missing some important features. Limitations of the white-box approaches include: reliance on a particular variant of the MLP architecture meaning that generality and learning capability are not guaranteed; network pruning requires retraining which can be time consuming and unreliable, and that only certain linear properties may have been identified.

It seems, therefore, that a combination of black-box and white-box techniques may be appropriate to derive a well organised *hybrid system*, which includes both an ANN as the automatic knowledge acquisition facility and a RBS as a high level inference engine and user interface. This may provide some major advantages:

- in overcoming the knowledge acquisition bottleneck problem of knowledge engineering, thus enabling an automatic knowledge engineering process;
- in providing an explicit representation of the knowledge encoded in ANNs, in the form of production rules, solving the problem of opaqueness of knowledge representation suffered by most ANNs;

- because the knowledge encoded in a trained ANN is uniform, whereas knowledge acquired from human experts often involves direct or indirect conflicts, which are difficult to detect and to avoid;
- symbolic knowledge representation may provide a better approach to the refinement of the knowledge acquired by ANNs, e.g. generality of the rule set can be adjusted even if the ANN has been under or over-trained;
- accuracy of the rule set representing the application domain can be optimised through a process of rule validation;
- probabilistic representation and reasoning with rules leads to system robustness in noisy and redundant environments.

The central themes of our hybrid system methodology are:

- selection of the optimal format for symbolic knowledge representation;
- translation of the implicit knowledge acquired by the ANN into an explicit knowledge format.

This paper introduces a common symbolic knowledge format, General Rules, in Section 2, to be used in a hybrid connectionist knowledge-based system, GR2. Section 3 introduces a novel, effective and efficient heuristic method for the extraction of knowledge from a trained multi-layer Perceptron. The GR2 system, which includes rule validation and multiple inference functions, is outlined in Section 4. Experiments in some artificial and real-world domains are reported in Section 5 and conclusions and recommendations for future work are presented in Section 6.

2. Abstract Knowledge Representation by General Rules

2.1 Definitions

We address binary problems in this paper.

Definition 1: A Boolean variable β, when it is instantiated, is represented in two forms:

- sub-symbolic form: 1 or 0, indicating instantiations $\beta = 1$ or $\beta = 0$ respectively. Traditionally, these are also referred to as positive or negative instantiations.

- symbolic form: β or, $\sim \beta$ indicating positive or negative instantiations of β respectively. Here β appears as the symbolic name of the variable.

Definition 2: L Boolean variables constitute an L-dimensional binary space, $B^L = \{0,1\}^L$. An element of the binary space is a vector of length L. A vector also has two forms:

- sub-symbolic form: $\langle \overline{b_1}, \overline{b_2}, \ldots, \overline{b_L} \rangle$, where $\overline{b_i}$ is either 1 or 0, corresponding to the i^{th} instantiated Boolean variable;
- symbolic form: $\langle \overline{\beta_1}, \overline{\beta_2}, \ldots, \overline{\beta_L} \rangle$, where $\overline{\beta_i}$ is either β_i or, $\sim \beta_i$ depending on the symbolic form of the instantiation of the i^{th} Boolean variable.

Definition 3: A binary problem is a triple: (I, O, F), where

- I is an N-dimensional binary *input* space, comprising N Boolean variables;
- O is a M-dimensional binary *output* space, comprising M Boolean variables;
- F is a function, which is a set of mappings $\{\overline{\omega} \rightarrow \overline{\gamma}\}$, where $\overline{\omega}$ is a vector from I and $\overline{\gamma}$ is an instantiated variable from O. Note, $\overline{\gamma}$ is not a vector.

Given a binary problem, representation of the function F in an effective, abstract format is the central target of an inductive learning system. An artificial neural network encodes the function in an implicit representation. A rule-based system, on the other hand, describes the function in an explicit symbolic representation.

Definition 4: Here MLP means a multi-layer Perceptron network having one layer of hidden units (the extension to networks having more than one hidden layer is trivial). Layers are fully interconnected. The input, the hidden and the output layers are denoted by $\{I_i\}$, $\{H_h\}$ and $\{O_o\}$ respectively, where the indices $i = 1,2,\ldots N, h = 1,2,\ldots Q$ and $o = 1,2,\ldots M$. The two layers of connection weights from the input to the hidden layer and from the hidden to the output layer are denoted by $W_1 = \{w_{ih}\}$ and $W_2 = \{w_{ho}\}$ respectively. An MLP is used to approximate the function, F, in a binary problem. *To ensure that MLP outputs are pseudo-binary (near 0 or 1) the network is trained until all output values lie in the ranges* $[0, \Delta]$ or $[1-\Delta, 1]$ *with* Δ *small and positive, a pre-defined tolerance.*

Definition 5: A *sole pattern*, $P_{jo} = \left(I^j, O_o^j\right)$, comprises the j^{th} input vector from the input space, taken together with *one* instantiated output variable, the o^{th} element in the j^{th} output vector. When the output variable is instantiated by 1, the sole pattern is said to be *positive*; otherwise, it is *negative*. Sole patterns collectively describe a function in a binary problem, and also the function of the MLP. A set of sole patterns can appear in contradiction to one another in a binary problem with noise, i.e. a distinct input vector may correspond to different values of an output variable. Sole patterns generated by the MLP are *uniform*, because there is a unique output corresponding to any input vector. *GR2 uses sole patterns generated by the trained MLP, and retains the uniformity of the acquired knowledge.*

Definition 6: A Rule is constituted with a premise part and a conclusion part, having the form

$$IF(\alpha_1, \alpha_2, \ldots, \alpha_L) THEN(\gamma)$$

where $0 < L \leq N$, α_j and γ are instantiated Boolean variables, either positively or negatively (headed with ~). An instantiated input variable, α_j, is called a premise, attribute or feature. The α_j are conjunctively related. An instantiated output variable, γ, is a conclusion or consequence. If γ is positive, the rule is a positive rule. If γ is negative, the rule is a negative rule. This form of rules is equivalent to the Horn Clause Format.

Definition 7: For a binary problem with N input variables, any rule having N premises is called a *specific rule*. A specific rule is the symbolic representation of a sole pattern. A rule, R, having L premises, where $0 < L < N$, is called a *general rule*. A general rule represents a set of specific rules whose conclusion is the same as that of the general rule, and whose premises are those of the general rule combined conjunctively with any possible combination of absent premises in the general rule. In other words, a general rule includes a set of specific rules, or it *covers* a set of sole patterns. In a general rule, the premises preserve the essential attributes, that are held in common in the set of sole patterns, by which this set is distinguished from other sole patterns. The absent premises in a general rule are not significant and thus may be ignored.

Definition 8: Given two general rules R_i and R_j, R_i is more general than R_j if it represents a set of specific rules subsuming the set of specific rules represented by R_j. Given any two general rules R_i and R_j from the same binary problem, the necessary and sufficient conditions

for R_i to be more general than R_j are, (a) R_i and R_j have the same conclusion; AND (b) R_i's premises form a subset of R_j's. Obviously, the fewer premises a general rule possesses, the more general it is. Two general rules are said to be *incomparable* if (a) they have different conclusions (either with a different variable, or with the same variable but instantiated by different values); OR (b) their premises mutually do not subsume.

Definition 9: A *non-redundant set* is a set of general rules, any pair of which is incomparable.

Sole patterns and rules are two different representations of the same objects. GR2 uses both representations for its hybrid components. Knowledge from a trained MLP is translated into a non-redundant set of general rules.

2.2 Generality vs. Accuracy

Accuracy and generality are essential criteria for machine learning. For GR2, accuracy is defined as the proportion of sole patterns correctly covered or classified by the general rules in the given pattern set, either the training set or the testing set. Generality is the degree of abstraction at which the general rules represent a property and is characterised by the average and spread of the number of premises in the extracted general rules. Generally speaking, the smaller the average number of premises of the rules, P_{av}, the more general the rule set is. To the same P_{av}, the larger the standard deviation of the premise numbers, P_{sd}, the more general the rule set is. A general rule set usually has fewer rules than a less general rule set, subject to the rule set being consistent and not redundant.

Predictive ability is attributed to generality of the rules provided that predictive accuracy on the training set is at an acceptable level. If two rule sets with different generality classify a training set to the same acceptable accuracy, the rule set with a higher generality is more likely to classify an independently selected testing set to a better accuracy than the other rule set. Because general rules disjunctively cover the training set and partially cover the domain space, our conclusion is similar to those given in (Holter et al., 1989) and (Quinlan, 1991), which declare that small disjuncts (each having small coverage in the training set) typically have poor predictive accuracy. This can be explained as the follows.

Figure 1 Generality leads to better predictive ability in the testing set range.

Figure 1 illustrates the fact that a few general rules (corresponding to the two large ellipses) can cover the range. Otherwise many more, but less general, rules (corresponding to the four smaller ellipses) are required. The importance of this is that the more general the rules are, the more likely it is that they evenly cover both ranges corresponding to the training and the testing (i.e. those not used in training the MLP) sole pattern sets. A less general rule, however, is unlikely to cover any sole pattern beyond the pre-determined range of the training set.

Any rule which is more general than necessary is more likely to lose the uniformity of its coverage and may misclassify some sole patterns. As shown in Figure 2, the two large ellipses cover the ranges across the boundary of Negative/Positive Sole Patterns. In other words, accuracy decreases as generality increases. The qualitative relationship between generality and accuracy is depicted in Figure 3.

Figure 2 Generality may cause partial mis-classification.

Figure 3 The relationship between generality and accuracy.

Optimisation of knowledge representation is beneficial to the trade-off between accuracy and generality. Sections 2.3 and 2.4 describe attempts at such optimisation, by which accuracy will suffer less as generality increases.

Generality is controlled simply in GR2 by choosing a threshold to be discussed later. Obtaining an adequate level of accuracy, is however, much more complicated. Note, an accuracy of 100% may be impossible in a practical data set which includes conflicting cases.

2.3 Probabilistic Rules

Rules whose coverage is assumed to be uniform are said to be *categorical*. Rules whose coverage is not uniform are said to be *probabilistic*. A confidence factor is used in conjunction with a probabilistic rule to decide conflicts statistically. It reflects the number of sole patterns correctly classified or mis-classified in the rule's coverage. The confidence factor, *cf*, for the i^{th} rule is defined by:

$$cf(R_i) = \frac{N_{cc} - N_{ic}}{N_{cc} + N_{ic}}$$

where N_{cc} is the number of sole patterns correctly classified, and N_{ic} is the number incorrectly classified. The value range is $[-1, 1]$. When the confidence factor is 1, the rule is equivalent to a categorical rule. Categorical rules are a special case of probabilistic rules. The confidence factor provides the capacity for classification under uncertainty. GR2 classifies sole patterns in a sequential procedure:

- *Categorical Reasoning:* If an input vector is covered by a set of categorical rules which have the same conclusion, the conclusion of the rules is the class the input vector belongs to. Otherwise:
- *Probabilistic Reasoning:* if the vector is covered by a set of rules which have different conclusions, its class is decided by the conclusion of those rules whose confidence factors, when summed, are greater than those of any opposing rules.

2.4 Exceptional Reasoning

GR2 usually generates both positive and negative rules for every output variable. However, as training patterns occasionally appear with features insufficiently distinct in every aspect, the rules for an output variable are provided only by either a positive or a negative form. GR2 performs *exceptional reasoning* to cope with this situation.

Exceptional Reasoning: check the input vector by the existing rules. If it is covered by any of the rules, the class is decided by the conclusion of the rules. Otherwise, the class is the opposite of the conclusion in the existing rules, which is equivalently used as a default rule in C4.5 (Quinlan, 1993).

If a training set contains patterns of one class as a great majority, it may be valid to extract rules only for the other classes and to imply the majority class as the default situation.

3. Extraction of General Rules

General rules are extracted from a trained MLP by two techniques performed in sequential passes. The first pass is a white-box approach, when the weight matrices are explored to obtain the critical data pathways through the MLP. These pathways are derived from static considerations only and thus take no account of the way a particular data vector (instantiation)

might sensitise or de-sensitise a path owing to its non-linear relationships. The information so gained is therefore inexact, requiring the second pass to confirm or otherwise the validity of any particular path when a particular input is present. The first pass is thus a heuristic to reduce the size of the search space for the second pass, a black-box approach. Here only the input/output behaviour of the MLP is observed and examined for the salient individual features of the input pattern set. As a result of these two passes, the rule extraction algorithm generates rules and controls the generality degree of the rule set with a threshold. The details are explained in the following three sections. This method was first introduced in (Ma et al., 1995b; Ma et al., 1995a), and is improved upon here.

The method does not require any modification to the standard MLP architecture or training algorithm, nor any retraining or weight clustering, and it is effective in both information dense cases, such as two (or more) bit AND, OR and parity problems, and in large-scale, noisy real-world domains.

3.1 The Potential Default Set

The relationship between any input unit and any output unit of the MLP can be *partially* captured by the matrix $L = (W_1 W_2)^t$. An element of L, $L_{oi} = \sum_h w_{ih} w_{ho}$ is the summed link strength between the i^{th} input unit I_i and the o^{th} output unit O_o, and measures the effect each input has on each output in a sense that will be made clear. That the elements of L form a useful measure is founded on (a) the monotonicity of the sigmoid function used in the MLP, and (b) the fact that the activations of the output units always fall in the tolerance ranges $[0, \delta]$ or $[1-\delta, 1]$, when the training input vectors are fed to the MLP. Here $0 < \delta < 0.5$ is a tolerance used in the MLP testing phase (as distinct from Δ used in training). Note: (a) can always be true for MLPs—it is the designers choice; (b) can always be enforced too, since δ can be loosely assigned (compared with the restriction on Δ) as long as all testing patterns are uniquely classified.

Observation is restricted to units I_i and O_o. If $L_{oi} > 0$, O_o tends to increase its activation as I_i switches from 0 to 1, and to decrease it as I_i switches from 1 to 0. However, if O_o activation is in the range $[1-\delta, 1]$ and I_i is 0, switching I_i will not impact on the classification result of O_o. I_i may therefore be ignorable in this case. Similarly, if I_i is 1, switching it will not change the

result if O_o is in the range $[0, \delta]$, I_i may again be ignorable. The situations are reversed when $L_{oi} < 0$. The situations where the I_i may be ignorable are summarised in Table I.

Table I Situations where I_i may be ignorable.

Let $\alpha_i, i = 1,2,\ldots,N$ denote the i^{th} input variable. Based on the foregoing analysis, the *Potential Default Set* (PDS) is defined as the subset of input units which potentially has no influence on the classification result of a particular output value, i.e. these units may as well remain at their default value.

Thus, given the o^{th} row of L, $L_o = \{L_{oi}\}$, we define two sets:

$$Z_1 = \{\alpha_i | L_{oi} \geq 0\} \qquad Z_0 = \{\alpha_i | L_{oi} < 0\}$$

and given an input vector $I = \{I_i\}$, we define another two sets:

$$N_1 = \{\alpha_i | I_i = 1\} \qquad N_0 = \{\alpha_i | I_i = 0\}$$

then the PDS of the input vector I, with respect to the output variable O_o, is defined by:

$$(Z_0 \cap N_1) \cup (Z_1 \cap N_0) \text{ if } O_o \in [1-\delta, 1]$$
$$(Z_0 \cap N_0) \cup (Z_1 \cap N_1) \text{ if } O_o \in [0, \delta]$$

The elements of the PDS are the candidates whose absence from a particular input vector are least likely to affect the classification result. Thus they may *potentially* be disregarded from the rules extracted from a sole pattern (I_j, O_o). However, it should be re-iterated here that, because the matrix L is based on entirely static considerations and is independent of the value of the current input vector, it would be dangerous to exclude the PDS from any rules on this basis alone. Rather it provides a heuristic method of reducing the search space for the black-box phase of the process. Only after this has been conducted is a decision made about whether or not to exclude members of the PDS. Experience has shown that use of the PDS reduces the problem size to approximately one half of the input dimension (Ma et al., 1995b; Ma et al., 1995a).

3.1.1 Extension to other neural networks

The method of determining the PDS presented above has concentrated on a "standard" MLP architecture. The extension to the cases of multiple hidden layers, partial inter-layer

connectivity, layer jumps and alternative activation functions is straightforward. Similarly its extension to other feedforward networks such as the Radial Basis Function Network (Moody and Darken, 1989) poses no problem.

One variation which does require a small modification is when output normalisation (the so-called Softmax activation function (Bridle, 1990)) is used. This form of output layer is especially useful when training a one-from-many classifier (in conjunction with a mutual information cost function) because it ensures that the values in the output layer sum to one, a necessary condition for the interpretation of these as posterior probabilities. Owing to the "sharing" of output activation under the Softmax function, outputs with low activation (low probability of being the correct consequent) should be ignored, i.e. negative rules should not be generated.

3.2 Feature Salient Degree

We now define the Feature Salient Degree (FSD) which provides a way of measuring how *informative* each input feature is, in the context of the training data. It is based on the degree of difference between each member of the training set and all the others. The definition given here differs slightly in detail from that given in our earlier papers (Ma et al., 1995b; Ma et al., 1995a) but is qualitatively similar. Results derived from both definitions are very similar but the this one has the better intuitive explanation.

The FSD is a $P \times N$ matrix, where P indicates the number of training patterns and N the dimension of the input space. First we define the matrix, *fsd*, whose ji^{th} element is given by the following equation. Both the FSD and the fsd matrices are defined w.r.t. the o^{th} output variable.

$$fsd_{ji} = \sum_{\left\{k \middle| \left(j \neq k, O_o^j \neq O_o^k, I_i^j \neq I_i^k\right)\right\}} \frac{1}{\left|P_{jo}, P_{ko}\right| 2^{\left|P_{jo}, P_{ko}\right|}}$$

and the Matrix *FSD* is defined thus:

$$FSD = \frac{fsd}{\max(fsd)}$$

P_{jo} and P_{ko} denote different patterns in the training set and $|P_{jo}, P_{ko}|$ is the Hamming distance of their input vectors. The *fsd* takes account of those patterns, P_{ko}, in the training set whose i^{th} input variable, I_i^k, and the output variable are different from those in P_{jo}.

Let $I(P_{jo})$ denote the input vector part of P_{jo}. To each pair of patterns, P_{jo} and P_{ko}, $I(P_{jo}) - I(P_{ko})$ may be regarded as the subset determining the class which the output of P_{jo} represents. The population of those possible input vectors subsuming this subset in the input space $\{0,1\}^N$ is

$$\frac{2^{N-|P_{oj},P_{ko}|}}{2^N} = \frac{1}{2^{|P_{jo},P_{ko}|}}$$

Furthermore, assuming each input in the property set equally characterises the property, the contribution of each input variable is $\frac{1}{|P_{jo}, P_{ko}| 2^{|P_{jo},P_{ko}|}}$. This explains the definition of *fsd*. This matrix is normalised by its maximal element so that all elements lie in [0,1]. This operation regularises the matrix when the training set has redundancy or is incomplete.

The FSD is a measure of the amount of information carried by the input units in the context of the training set. It represents the correlation of the changes on an instantiated input variable and an instantiated output variable, estimating the possibility of a change of the output value when the input variable is switched. Here the MLP is used as a black-box for computation of the output values of the sole patterns.

3.3 Rule Extraction with the PDS and the FSD

There is a key parameter used in conjunction with the FSD—the FSD class threshold, T. This is used to control the generality of the extracted rule set. Testing the elements of the FSD matrix against T determines if an individual input or a set of inputs is preserved for forming the premise(s) of an extracted rule. T must lie within the range [0,1]. T is specified for each class.

General rules are extracted from a sole pattern P_{jo}=(I^j, O_o^j), where I^j is an input vector $\langle I_1^j, I_2^j, \ldots I_N^j \rangle$, as follows.

3.3.1 Rule Generation

1. Compute the PDS and FSD_j (the j^{th} row of FSD) for P_{jo}
2. Generate a set $\psi = \left(I_i^j | FSD_{ji} \geq T\right)$. Generate a set, Θ, of "smallest subsets" of I, $\Theta = \left\{\theta_k | \forall \theta_1 : \theta_1 \neq \theta_k \Rightarrow \left(\theta \not\subset_k \theta_1, \theta_1 \not\subset \theta_k\right)\right\}$ i.e. all of the subsets, θ_k, are mutually exclusive, where $\theta_k = \left\{I_i^j | I_i^j \notin \text{PDS}, \frac{T}{N \log N} < FSD_{ji} < T, \sum_i FSD_{ji} \geq T\right\}$. The lower bound $\frac{T}{N \log N}$ to select the FSD elements being defined in order to avoid the exponentially increasing computation for testing all possible subsets, θ_k. The particular choice has been made empirically in the light of its effectiveness. We furthermore suggest the limit as 20. In practice, a rule with more than 20 premises is not easily understood.
3. Construct general rules by pairs (ψ, O_o) and all (θ_k, O_o) respectively. ψ and θ_k are exclusive of one another. The former, a set of instantiated input variables, is "symbolised" into premises. The latter, a single instantiated output variable, is "symbolised" into the conclusion. In each rule, the atomic truth value is defined by the variable's value. If the variable's value is 1, the corresponding atom has the truth value "True", and presents its symbol in the rule. If the variable's value is 0, the corresponding atom has the truth value "False", and the symbol in the rule is headed with ~.

These steps are expressed in the following algorithm.

3.3.2 Rule Generation Algorithm

Figure 4 shows the algorithm for rule extraction from an MLP with N inputs, M outputs and P training patterns.

Figure 4 Flow chart for rule extraction.

1. Evaluates sole patterns for every output bit.
2. Is as described in Sections 3.1 and 3.2.
3. Is the kernel procedure, as described in the previous subsection. The FSD threshold, T, is defined by the user in advance. A list of rules is therefore constructed.
4. This deals with two sets of rules, the new list and the global rule-base (GRB). For a rule R_i in the new list, do one of the operations in following cases:
   Specific: if R_i is more specific than any rule in GRB, R_i is ignored.

General: if R_i is more general than some rules in the GRB, R_i replaces all more specific rules in the GRB.
Generalisable: if R_i in combination with some other rules covers the combinations of a subset of its premises, merge them into a more general rule.
New: otherwise R_i is novel and is inserted into the GRB.
Note that in (i) to (iii) R_i is checked only with those rules in the GRB which are comparable with R_i.

5. Check each rule R_i (with output O_i) in the GRB against the sole patterns in the training set whose output bit corresponds to the consequence atom in R_i. If the rule is more general than a sole pattern (with output, O_k^j, i.e. the premise of R_i is subsumed by the input vector of the pattern) then:
   if $O_i = O_k^j$ (within the error tolerance), increment the correct occurrence count, N_{cc}, by 1
   otherwise increment the error occurrence count, N_{ic}, by 1.
   Compute the confidence factor cf_i for R_i.
   Remove those R_i s whose confidence factor is less than some non-negative threshold.
   The remaining rules in the GRB are said to be valid.

3.3.3 Computational Complexity

Assume the trained MLP has N inputs, Q hidden units and M outputs and there are P training patterns (i.e. MP sole patterns). It is usual for $N>>M$ and $P>>N$. The PDS requires a computation in $\max(O(N\times M\times P), O(N\times Q\times P))$, the latter value being for the matrix multiply W_1W_2. Computation of fsd_{ji} requires $O(N\times P)$, and the matrix fsd w.r.t. an output requires $O(N^2\times P^2)$. The computation of FSD w.r.t. all output bits requires $O(N^2\times M\times P^2)$, thus the rule generation algorithm takes computation of $O(N^2\times M\times P^2)$. In practice, 70-90% of the computational effort of the algorithm is consumed by the P times forward recall of the MLP for the evaluation of the output values of the sole patterns. This is, however, less than the computation of one epoch in the training stage of the MLP.

Comparison with the complexity of other methods is not straightforward because of a lack of similar analyses there. It is, however, worth noting that the KBANN approach (Towell and Shavlik, 1993) requires re-training to cluster the weights after training which takes computation of the same order as that of training. The method in (Saito and Nakano, 1990), a typical black-box example, is relatively easier to analyse. Assume there are $N/2$ positive inputs in each pattern of the training set, and the limit of the lengths on the extracted rules is L, the computation requires $P\times{}^LC_{N/2}$ recalls of the MLP, in contrast to the method in this paper taking no more than $1.5P$ recalls. In fact, since the number of positive inputs in each pattern of

the training set varies, the computation in (Saito and Nakano, 1990) may be much greater than the estimate given here.

3.4 Quantification of Rules and Their Elements

A general rule comprises two parts, the antecedent and the consequent. For maximum utility it is desirable to associate a numerical degree with each extracted rule which indicates its validity within the domain or at least that part represented by the training set. This validity factor is precisely the confidence factor defined in Section 2.3. Furthermore, a measure of the contribution any input channel makes to a particular rule provides the user with guidance on which premises may be regarded as noisy within a rule. We call this number the *contribution factor (CF)*.

3.4.1 Confidence Factors

Extracted rules are checked through the rule validation procedure. A confidence factor is generated for each consequent or, in the case of a single output system, for the extracted rule itself. Those rules having low or negative validity factors are removed from the rule set.

During rule validation each rule is checked against every pattern in the training set. If the rule covers the input vector of the pattern, i.e. the antecedents of the rule are a subset of the input vector, the rule is *comparable* with the pattern. Every comparable pattern is said to be a *matching* pattern if that pattern's output bit matches the rule's consequent. Otherwise it is in conflict with the rule.

Clearly, those rules with negative confidence factors must be eliminated. It may also be desirable to excise rules with low, positive *cf*. A user defined *cf*-threshold allows this to be done. Thus by listing rules and their confidence factors the user's confidence in any particular rule can be influenced.

3.4.2 Contribution Factors

For each input bit, the FSD reflects the probability of occurrence of that bit in the training set under the constraint of its forming combinations with other bits, i.e. the joint probability. We use the normalised FSD of an antecedent in a rule to express its contribution to that rule's consequent. The contribution factor for the i^{th} rule is defined thus

$$CF_i = \frac{FSD_{ki}}{\sum_j FSD_{kj}}$$

where j ranges over the antecedents of that rule.

4. GR2 System Architecture

The GR2 system is depicted in Figure 5. The first component is an MLP as defined in Section 2.1. After training, the MLP will remain static. The second component for Rule Extraction executes the algorithm described in Section 3.3.2 and categorical rules are generated.

The third component is for Rule Validation. Rule validation is a process to determine whether or not the rules perform with an acceptable accuracy over the training pattern set. The Rule Validation process includes several functions:

- computation of the confidence factors for rules, by checking rules in the training pattern set;
- elimination of those rules whose confidence factor is less than the confidence threshold;
- prevention of redundancy by deleting rules which are more specific than any other rules;
- generalisation of rules by combination with similar rules where possible.

Figure 5 Control and data flow for the GR2 system. Broken lines indicate sub-systems yet to be implemented.

The fourth component, Inference by Rules, is the inference engine classifying input vectors by rules. The inference is simple because the rules constitute a direct mapping from the input space to the output space, with no intermediate variables involved. The inference process is

- Do exceptional reasoning if necessary. Otherwise, do reasoning by direct matching. In both cases, the following two steps are executed:

  if the input vector is covered by categorical rules, do categorical reasoning; otherwise, do probabilistic reasoning.

5. Examples

GR2 has been successfully tested on a number of artificial binary problems and two medical applications using data gathered from clinical settings (Ma et al., 1995b; Ma et al., 1995a). Artificial problems are usually information dense and categorical reasoning is sufficient in those situations. Real-world domains are usually information sparse, where probabilistic reasoning and a trade-off between generality and accuracy is required.

In Section 5.1, some simple logic problems are used to demonstrate the rule extraction process. A four bit parity problem with an incomplete training set is presented in Section 5.2. The results of applying the method to a clinical problem are presented in Section 5.3.

5.1 Two Bit AND and XOR

5.1.1 Two Bit AND

To illustrate the rule extraction process a simple example is worked "by hand" which can be verified by inspection. The two bit AND problem is represented in Table II where A, B are one bit inputs, and C is one bit output. Although it is well known that hidden units are not necessary in the MLP to solve this simple domain, we use them for the purpose of demonstration.

Table II Two bit AND truth table

Using an MLP architecture with 2 input units, 1 hidden unit, and 1 output unit, with one bias to each of the hidden units and to the output unit, as shown in Figure 6.

Figure 6 A trained MLP for the two-bit AND data

The weight matrices after training are:

$$W_1 = \begin{bmatrix} -3.266 \\ -3.270 \end{bmatrix}; \quad W_2 = [-5.305].$$

Now we calculate the PDSs. Since both elements in $L = (W_1 W_2)^T = [17.326 \quad 17.347]$ are positive: $Z_0 = \{\}$ $Z_1 = \{A, B\}$.

Since there is only one output bit, we use P_j, O_j instead of P_{jo}, O_j^o for simplicity.

For the first sole pattern ($< 0,0 >, 0$), $N_0 = \{A, B\}$ $N_1 = \{\}$

As the output bit O_1=0, the PDS for pattern 1 is $PDS_1 = (Z_0 \cap N_0) \cup (Z_1 \cap N_1) = \{\}$

For the second pattern

(<0,1>,0), $N_0 = \{A\}$ $N_1 = \{B\} \therefore PDS_2 = (Z_0 \cap N_0) \cup (Z_1 \cap N_1) = \{B\}$

For the third pattern (<1,0>,0), $N_0 = \{B\}$ $N_1 = \{A\} \therefore PDS_3 = (Z_0 \cap N_0) \cup (Z_1 \cap N_1) = \{A\}$

For the fourth pattern (<1,1>,1), $N_0 = \{\}$ $N_1 = \{A, B\}$

and since the output bit O_1=1, $PDS_4 = (Z_0 \cap N_1) \cup (Z_1 \cap N_0) = \{\}$

Now delete the PDS from the full set of input symbols, we obtain the following remainder matrix, where – indicates absence of any symbol:

$$\begin{bmatrix} A & B \\ A & - \\ - & B \\ A & B \end{bmatrix}$$

To compute the FSD matrix, we first calculate *fsd*:

For the pattern (< 0,0 >,0), for input channel 1, only pattern 4, (<1,1>,1), is counted because both the first input bit and the output bit are different in the two patterns. The hamming distance $|P_1, P_4|$ is 2. Therefore the corresponding element in fsd is

$$fsd_{11} = \frac{1}{2 \times 2^2} = 0.125, \quad fsd_{12} = fsd_{11}$$

For the pattern (<0,1>,1), only pattern 4 has a different output bit and $|P_2, P_4|$ is 1. Therefore $fsd_{21} = \frac{1}{1 \times 2^1} = 0.5$ corresponding to input channel 1. For input channel 2, $fsd_{22} = 0$ because this pattern and pattern 4 have the same second input bit although they have different output bits.

Similarly for pattern (<1,0>,1), $fsd_{31} = 0$ and $fsd_{32} = 0.5$.

For pattern (<1,1>,1), all the other three patterns have different output bits so that whether or not they are counted for the FSD values for this pattern depends on whether or not their corresponding input bits are different from those in this pattern. For input channel 1, the

pattern is different from patterns 1 and 2, and the hamming distances of the input pairs are 2 and 1 respectively. Therefore

$fsd_{41} = \frac{1}{2 \times 2^2} + \frac{1}{1 \times 2^1} = 0.625$. Similarly for input channel 2, pattern 1 and pattern 3 are different from the pattern, and $fsd_{41} = fsd_{42}$. We have

$$fsd = \begin{bmatrix} 0.125 & 0.125 \\ 0.5 & 0 \\ 0 & 0.5 \\ 0.625 & 0.625 \end{bmatrix} \quad \text{and} \quad FSD = \begin{bmatrix} .269 & .269 \\ .731 & 0 \\ 0 & .731 \\ 1 & 1 \end{bmatrix}$$

There are two classes of rules, the positive class and the negative class according to the truth value of their consequences. We use the same threshold for both of the classes for this domain. If we choose an FSD class threshold, T, in the range [0.538, 0.731), there is no rule generated from the first pattern since $FSD_{11}+FSD_{12}$ is less than T, even though both A and B are not in the PDS.

From pattern 2, (<0,1>,0), A=0 remains since $FSD_{21} \geq T$. A rule is generated:

IF (~A) THEN (~C)

which is added to the GRB.

Similarly from pattern 3, (<1,0>,0), B=0 remains, generating a rule:

IF (~B) THEN (~C)

From pattern 4, (<1,1>,1), both elements of FSD_4 =[1 1] are greater than T, both input variables are retained as premises, generating a rule:

IF (A,B) THEN (C)

We quantify the rules according to their validity and contribution factors thus:

IF (~A(1.0)) THEN (~C) (1.0)

IF (~B(1.0)) THEN (~C) (1.0)

IF (A(0.5),B(0.5)) THEN (C) (1.0)

where the numbers in parentheses attached to each antecedent indicate the contribution factors and those to the right of each rule indicate confidence factors . The confidence factors indicate that all extracted rules have maximal validity (and are therefore necessary). The contribution factors indicate that the antecedents in the third rule are equally important for its firing.

Notice that if the threshold is chosen in the range (0,0.731), the extracted rule set is identical. The analysis is as the follows.

If a threshold T is chosen in the range [0.269,0.538), a rule IF (~A,~B) THEN (~C) will be generated from the first pattern (<0,0>,0) because FSD11+FSD12≥T. From the other three patterns, there will be three rules extracted, as before. Then the first rule would be subsumed into the next more general rule, either IF (~A) THEN (~C) or IF (~B) THEN (~C).

If a threshold T is chosen in the range [0.269,0.538), a rule IF (~A,~B) THEN (~C) will be generated from the first pattern (<0,0>,0) because $FSD_{11}\geq T$ and $FSD_{12}\geq T$ and both input bits are included in the rule according to the rule extraction algorithm. This rule will be subsumed too when the other rules are extracted from the other patterns as previously.

Thus the rule extraction method is relatively tolerant to the choice of the threshold.

5.1.2 XOR

The XOR problem has been widely studied as a learning task for MLP networks and presents problems if naive rule extraction procedures, based, say, on sensitivity analysis are used. We omit the detail here and remark that the FSD matrix bears a one in every element, thus for any choice of FSD threshold less than one, all input variables are retained as premises with equal contribution factors.

The extracted rules are precisely those of the symbolic form. This is a common feature of parity problems—they always result in FSD matrices whose every element is unity. This results in a full list of the rules in the original symbolic form, as should be the case. As a consequence, the confidence factor associated with each extracted rule is one.

5.2 Incomplete Training Set for a Four Bit Problem

Predictive performance is assessed by the accuracy of recognition on the patterns outwith the training set. Here we show how GR2 tackles this situation using the example of four bit parity.

A training set of four inputs, named A, B, C, and D, and one output, named E, includes only 11 of a potential 16 distinct patterns. Odd parity is assigned E and even parity is assigned ~E. The patterns are listed in Table III with those missing from the training set shown in shaded columns. The outputs are determined by a trained MLP (size 4:3:1). All unshaded vectors are correctly classified by the MLP, as is pattern number five. The others in the shaded columns are incorrectly classified.

Table III Patterns forming the incomplete 4 bit domain (shaded patterns excluded from the training set)

We now replace the FSD-threshold, T, by a pair of thresholds, T_p and T_n, which may be adjusted independently of one another depending on the appropriate classification—positive or negative.

In a dense, deterministic domain such as this, the best performance it is reasonable to expect is that all 11 patterns in the training set are correctly classified and that the other 5 patterns are misclassified. This occurs when the threshold for the positive class $T_p \in (0, 0.2]$ and that for the negative class $T_n \in (0, 0.8]$. However, the generality of the extracted rule set increases as higher thresholds are used even though the accuracy remains the same.

For example, if $T_p = T_n = 0.05$, the extracted rules correspond exactly to the 11 training patterns in symbolic form, where the number of rules $N_r = 11$, the average number of premises $P_{av} = 4$, and the standard deviation of the number of premises in the rules $P_{sd} = 0$.

If the thresholds are taken as $T_p = T_n = 0.2$, then the rule set is

IF (~A(0.39), ~B(0.11), ~C(0.11), ~D(0.39)) THEN (~E); (1.00)

IF (B(0.32), ~C(0.32), ~D(0.36)) THEN (~E); (1.00)

IF (~A(0.39), B(0.11), C(0.11), ~D(0.39)) THEN (~E); (1.00)

IF (~A(0.36), B(0.32), ~C(0.32)) THEN (~E); (1.00)

IF (A(0.24), B(0.26), C(0.26), D(0.24)) THEN (~E); (1.00)

IF (A(0.47), ~B(0.53)) THEN (E); (1.00)

IF (A(0.35), C(0.35), ~D(0.31)) THEN (E); (1.00)

IF (~B(0.53), D(0.47)) THEN (E); (1.00)

IF (A(0.35), ~C(0.31), D(0.35)) THEN (E); (1.00)

IF (~A(0.31), C(0.35), D(0.35)) THEN (E); (1.00)

and N_r=10, P_{av} =3.1 and P_{sd} =0.7. This rule set is more general than the previous one.

If the thresholds are taken as T_p=0.8 and T_n =0.2, the rule set is

IF (~A(0.50), ~D(0.50)) THEN (~E); (1.00)

IF (B(0.32), ~C(0.32), ~D(0.36)) THEN (~E); (1.00)

IF (~A(0.36), B(0.32), ~C(0.32)) THEN (~E); (1.00)

IF (A(0.24), B(0.26), C(0.26), D(0.24)) THEN (~E); (1.00)

IF (A(0.47), ~B(0.53)) THEN (E); (1.00)

IF (A(0.35), C(0.35), ~D(0.31)) THEN (E); (1.00)

IF (~B(0.53), D(0.47)) THEN (E); (1.00)

IF (A(0.35), ~C(0.31), D(0.35)) THEN (E); (1.00)

IF (~A(0.31), C(0.35), D(0.35)) THEN (E); (1.00)

and N_r=9, P_{av} =2.778 and P_{sd} =0.63. This rule set is again more general than the previous one.

Raising the thresholds until T_p=0.8 and T_n =0.8, the rule set becomes more general still, with only one more pattern misclassified.

The rules were then used directly to infer conclusions from the inputs. They provided an identical set of conclusions to those provided by the MLP, including the mis-assignments.

5.3 Diagnosis of Acute Myocardial Infarction (Heart Attack)

The early identification of patients with acute ischaemic heart disease remains a great challenge in emergency medicine. The electrocardiograph (ECG) only shows diagnostic changes in about half of acute myocardial infarction (AMI) patients at presentation (Adams et al., 1993b). None of the available biochemical tests becomes positive until at least three hours after symptoms begin, making such measurements of limited use for the early triage of patients with suspected AMI (Adams et al., 1993a). The early diagnosis of AMI, therefore, relies on an analysis of clinical features along with ECG data. An MLP has been shown to be a good method for combining clinical and electrocardiographic data into a decision aid for the early diagnosis of AMI (Harrison et al., 1991). The data used in this study were derived from consecutive patients attending the Accident and Emergency Department of the Royal Infirmary, Edinburgh, Scotland, with non-traumatic chest pain as the major symptom. The relevant clinical and ECG data were entered onto a purpose-designed proforma at, or soon

after, the patient's presentation. The study included both those patients who were admitted and those who were discharged. 970 patients were recruited during the study period (September to December 1993). The final diagnosis for these patients was assigned independently by a Consultant Physician, a Research Nurse and a Cardiology Registrar. This diagnosis made use of follow-up ECGs, cardiac enzyme studies and other investigations as well as clinical history obtained from review of the patient's notes.

The input data items for the MLP were all derived from data available at the time of the patient's presentation. In all, 35 items were used, coded as 37 binary inputs, whose medical definitions are given in the Appendix. For the purposes of this application, the final diagnoses were collapsed into two classes termed "AMI" (Q wave AMI and non-Q wave AMI) and "not-AMI" (all other diagnoses). AMI cases were assigned as positive diagnoses, not-AMI cases as negative diagnoses. The MLP was constructed with 37, 13 and 1 as the sizes of the input, hidden and output layers, respectively. The training error tolerance was $\Delta = 0.1$. The 970 patient records were divided into two data sets, 400 randomly selected as the training set, and the remaining 570 as the testing set.

In addition to crude accuracy, two additional performance figures are of interest in diagnosis: sensitivity and specificity.

Sensitivity (Sens) is defined as the ratio of the number of correct positive diagnoses to the number of positive outcomes, i.e. the proportion of diseased patients correctly identified as such. This is important because the disease is life-threatening and is just the accuracy of the classification on the positive training cases. Thus sensitivity may be influenced by the assignment of T_p.

Specificity (Spec) is defined as the ratio of the number of correct negative diagnoses to the number of negative outcomes, i.e. the proportion of healthy individuals correctly identified as such. This is important because treatment can be expensive and risky. Specificity is just the accuracy of the classification over the negative training cases and can be influenced by the choice of T_n.

Accuracy (Acc) is defined as the ratio of the number of correct diagnoses to the total number of patients.

Among the 970 records, 192 are positive diagnoses and 778 are negative diagnoses. We assign different FSD class thresholds for each of the classes. We find that extracting only positive rules and leaving the negative diagnoses to exceptional reasoning gives the best result, because the negative patterns constitute more than 80% of the data set. To achieve this, the threshold for the negative class is assigned as T_n>0.6, i.e. simply ignore negative patterns during rule extraction. The "optimal" threshold value for the positive class is found to be T_p=0.26, selected from a batch run with different values of T_p. R_{tot} and R_{val} indicate the total number of rules and the number of valid rules, respectively. P_{av} indicates the average number of premises in the valid rules and P_{sd} indicates the standard deviation of the numbers of premises of the valid rules.

The performance measures are subscripted with "test" or "train" to indicate to which data set they apply. The shaded column describes the "optimal" situation.

The performance results are listed in Table IV

Table IV GR2: Experimental Results on AMI Records

The extracted rules for the "optimal" situation ($T_n \in (0.5,1.0)$, $T_p = 0.26$), are given as follows (see Appendix for definitions of the antecedents).

GR1. IF (NEWQ(1.00)) THEN (AMI); (0.71)

GR2. IF (STELEV(1.00)) THEN (AMI); (0.64)

GR3. IF (EX_SMOKER(0.26), LVF(0.40), STTWAVE(0.34)) THEN (AMI); (0.25)

GR4. IF (NAUSEA(0.30), STTWAVE(0.70)) THEN (AMI); (0.23)

GR5. IF (SMOKES(0.26), SWEAT(0.29), STTWAVE(0.45)) THEN (AMI); (0.47)

GR6. IF (RARM(0.49), STTWAVE(0.51)) THEN (AMI); (0.21)

GR7. IF (AGE>65(0.14), SWEAT(0.37), HYPOPERF(0.49)) THEN (AMI); (0.80)

GR8. IF (~LARM(0.33), JAW(0.34), STTWAVE(0.33)) THEN (AMI); (0.27)

The rule generation processes took between 36 and 50 seconds on a Sun Sparcstation 10.

We have also used C4.5 (Quinlan, 1993) to extract rules from the same data. After generating the decision tree, we select a confidence level of 0.25% to extract the rules (using the

command 'c4.5rules -f fct -u -c .25'), which we found gave the best results over the course of many experiments. The rules extracted via C4.5 are listed below.

C1. IF (NEWQ) THEN (AMI) [60.3%]

C2. IF (STELEV) THEN (AMI) [59.2%]

C3. IF (~EX_SMOKER, ~LCHEST, STTWAVE, ~OLD_Q) THEN (AMI) [58.8%]

C4. IF (ALLTIGHT, HYPOPERF) THEN (AMI) [48.3%]

C5. IF (~FAM_IHD, ~HYPERTENSE, ~RCHEST, ~BACK,
~LARM, LVF, ~NEWQ) THEN (AMI) [38.5%]

The rules extracted by C4.5 are converted into GR2 format. There is no contribution factor in the premise part and the last figure of each rule is regarded as the predictive accuracy (see (Quinlan, 1993) pp 150–155).

A comparison of MLP, GR2 and C4.5 when each performs "optimally" is shown in Table V.

Table V Comparison of MLP, GR2 and C4.5 performance.

5.3.1 Evaluation of the rules

Here we discuss the validity of the rules from a medical point of view, dealing with groups of rules that are conceptually related.

GR1, GR2, C1 and C2. The appearance of new Q-waves (NEWQ) in the ECG traces is a major marker for recent MI. Here "new" means that the ECG under consideration differs from an accepted, healthy standard. Whether these changes are recent of not is not detectable unless an earlier ECG is available for comparison because Q-wave changes are non-volatile. For this reason we would expect the presence of NEWQ to be the second most discriminatory feature in a positive diagnosis. The most discriminatory would be expected to be the elevation of the ST-segment of the ECG (STELEV). Because this is volatile, its presence would indicate AMI with very high certainty. GR2 and C4.5 both reveal these important rules.

GR3 and C5. Left ventricular failure (LVF) is an important indicator of poor cardiac muscle function associated with unrecoverable damage. It assists therefore in discriminating between MI and unstable angina, which does not result in such damage. The conjunction of LVF with a major risk factor such as a history of smoking (EX_SMOKER) and non-specific changes in

the ECG (STTWAVE) militates strongly towards a diagnosis of MI in GR3. In C5, the development of LVF in the *absence* of other typical features (the negative antecedents) may over-ride their importance. This is especially true if the onset of LVF is sudden. Again, GR2 and C4.5 both deliver valid rules with a sound clinical explanation.

GR4, GR5, GR6 and GR8. These hinge on non-specific abnormalities in the ECG indicated by ST-segment depression and T-wave flatness or inversion (STTWAVE). Such patients are under suspicion because almost all MI cases will exhibit these changes. Signs of physiological distress such as nausea (NAUSEA) or sweating (SWEAT), risk factors such as a history of smoking (SMOKES) or other known discriminants such as pain down the right arm (RARM), combined with non-specific ECG changes all militate towards a positive diagnosis. Of particular interest here are GR6 and GR8 because although pain in the *left* arm is regarded as typical, it may be indicative of angina as well as of MI, whereas right arm pain is able to discriminate MI from angina as has been shown in a number of studies. Furthermore, GR8 is able to use atypical, but relevant, pain information to influence non-specific ECG readings.

GR7. This rule has identified the difficulties of diagnosing MI in elderly patients. In such cases it is important to combine signs of physiological distress such as sweating (SWEAT) and hypoperfusion (HYPOPERF) with other markers, owing to the high incidence of atypical presentations, e.g. silent (pain-free) infarctions. For instance, it is frequently difficult to detect recent ECG changes because an elderly patient may have suffered cumulative damage over many years. This would not necessarily be known to a front-line department.

C4. Pain being described as tight by the patient is typical but may also indicate angina, however, the indication of distress (HYPOPERF) once again militates towards a positive diagnosis.

C3. This rule is less straightforward than the others and, indeed, is the most complex one discovered. The patient is under suspicion owing to the non-specific ECG changes (STTWAVE), however, unlike the earlier cases, there is no simply interpreted support from the other features. The selection of ~EX_SMOKER is ambiguous because it may indicate a non-smoker or a smoker. There is no known link between the cessation of smoking and MI. The absence of Q-wave changes *known* to be old (~OLD_Q) increases the significance of any changes observed, while the absence of pain from the left chest (~LCHEST) is atypical.

In all cases the relative importance of the antecedents as expressed by the contribution factors appears to be appropriate.

6. Conclusions and Recommendations

A method for extracting general or production rules from a trained feedforward artificial neural network, in a binary domain, has been devised, tested and found to be successful. It is exemplified within the framework of the multi-layer Perceptron, although its generalisation to other feedforward architectures is straightforward. One feature which distinguishes the approach is its applicability to the standard configuration—no special structure is required, nor is any pruning or re-training.

A combination of black-box and white-box methods is used. The latter is a heuristic which makes use of the strengths of the (static) input/output pathways to isolate a set of inputs whose value is *potentially* unimportant to the current conclusion. These inputs form the potential default set which leads to substantial potential reductions in the size of the search space. However, owing to the non-linearity of the system and the interdependence of input/output channels, a second stage, treating the network as a black-box, is introduced to investigate actual input/output correspondences. These lead to an entropy-like measure: the feature salient degree. The validity of ignoring an input in the potential default set is checked via a threshold test on the feature salient degree. The generality of the extracted rule set is easily adjusted by varying this threshold. The rule extraction technique forms the core of a larger system, GR2, which, in addition performs inference.

The methodology is shown to be effective in information dense, logic domains and also in an important medical domain which is information sparse and noisy. In the latter, the medical veracity and validity of the extracted rules is evaluated and these are seen to be well founded. The method is compared with the C4.5 machine induction approach with which it compares favourably. In all cases extraction and inference by GR2 maintains the level of performance of the underlying neural network.

To enhance visibility and hence user confidence, and to improve the efficiency and effectiveness of the system, expansion of GR2 to include

- a case-by-case explanation facility, and
- a user query system which is sensitive to dynamic context

are seen as important next steps. Work is also under way to expand the methodology to encompass n-ary or fuzzy-valued inputs and also the problem of constrained sets of input features such as the coding for age, given in the Appendix.

Acknowledgement

This research work has been supported by the Engineering and Physical Sciences Research Council, UK, Grant Number GR/J29916.

References

Adams J.E., Abendschein D.R., & Jaffe A.S. (1993a) Biochemical markers of myocardial injury. Is MB creatine kinase the choice for the 1990s? *Circulation,* **88,** 750-763.

Adams J.E., Trent R., & Rawles J. (1993b) Earliest electrocardiographic evidence of myocardial infarction: implications for thrombolytic treatment. *British Medical Journal,* **307,** 409-413.

Andrews, R., & Geva, S. (1995) RULEX & CEBP networks as the basis for a rule refinement system. In Hallam, J. (Ed.), *Hybrid Problems, Hybrid Solutions: 10th Biennial Conference on AI and Cognitive Science*. Amsterdam: IOS Press.

Bochereau L. & Bourgine P. (1990) *Rule extraction and validity domain on a multi-layer neural network.* The International Joint Conference on Neural Networks.

Bridle, J.S. (1990) Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In Touretzky, D., S. (Ed.), *Advances in Neural Information Processing 2*. San Mateo: Morgan Kauffman.

Carpenter G. & Grossberg S. (1987) A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics and Image Processing,* **37,** 54-115.

Carpenter G. & Tan A. (1993) *Rule extraction, fuzzy ARTMAP and medical databases.* World Congress on Neural Networks.

Cybenko G. (1989) Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems,* **2,** 303-314.

Diederich J. (1990) *An explanation component for a connectionist inference system.* European Joint Conference on Artificial Intelligence 90.

Fu, L.M. (1994a) *Neural networks in computer intelligence*. McGraw-Hill.

Fu L.M. (1994b) Knowledge-based connectionism for revising domain theories. *IEEE Transactions on Systems,Man and Cybernetics.* **23,** 173-182.

Gordon J. & Shortliffe E.H. (1985) A method of managing evidential reasoning in a hierarchical hypothesis space. *Artificial Intelligence,* **26,** 323-357.

Harrison R.F., Marshall S.J., & Kennedy R.L. (1991) *A connectionist aid to the early diagnosis of myocardial infarction.* Third European Conference on Artificial Intelligence in Medicine. Maastricht.

Hart A. & Wyatt J. (1990) Evaluating black-boxes as medical decision aids: issues arising from a study of neural networks. *Medical Informatics,* **15,** 229-236.

Holter R.C., Acker L.E., & Porter B.W. (1989) *Concept learning and the accuracy of small disjuncts.* 11th Intrnational Joint Conference on Artificial Intelligence.

Langlotz C.P., Shortliffe E.H., & Fagan L.M. (1986) *Using decision theory to justify heuristics.* The National Conference on Artificial Intelligence.

Ma Z., Harrison R.F., & Kennedy R.L. (1995a) *GR2 - A hybrid knowledge-based system using general rules.* International Joint Conference on Artificial Intelligence. Montreal.

Ma, Z., Harrison, R.F., & Kennedy, R.L. (1995b) A heuristic for general rule extraction from a multilayer Perceptron. In Hallam, J. (Ed.), *Hybrid Problems, Hybrid Solutions: 10th Biennial Conference on AI and Cognitive Science*. Amsterdam: IOS Press.

Mahoney J. & Mooney R.J. (1993) *Combining neural and symbolic learning to revise a probabilistic rule-base.* European Joint Conference on Artificial Intelligence 93.

McMillan C., Mozer M.C., & Smolensky P. (1991) *Rule induction through integrated symbolic and subsymbolic processing.* Advances in Neural Information Processing Systems.

Moody J. & Darken C. (1989) Fast learning in networks of locally-tuned processing units. *Neural Computation,* **1,** 281-294.

Quinlan J.R. (1986) Induction of decision trees. *Machine Learning,* **1,** 81-106.

Quinlan J.R. (1991) Improved estimates for the accuracy of small disjuncts. *Machine Learning,* **6,** 93-98.

Quinlan, J.R. (1993) *C4.5: Programs for Machine Learning*. Morgan Kauffman.

Richard M. & Lippman R. (1991) Neural network classifiers estimate Bayesian *a posteriori* probabilities. *Neural Computation,* **3,** 461-483.

Ridella, S., Speroni, G., Trebino, P., & Zunino, R. (1994) Class-entropy minimisation networks for domain analysis and rule extraction. In *Neural Computing and Applications.* London: Springer-Verlag.

Rumelhart D., Hinton G., & Williams R. (1986) Learning representations by back-propagating errors. *Nature,* **323,** 533-536.

Saito K. & Nakano R. (1990) *Rule extraction from facts and neural networks.* International Neural Network Conference.

Shortliffe, E.H. (1976) *Computer-based medical consultations: MYCIN.* New York: Elsevier.

Towell G. & Shavlik J.W. (1993) Extracting refined rules from knowledge-based neural networks. *Machine Learning,* **13,** 71-101.

Appendix: Medical Definitions of AMI Symptoms

AGE<45: Age less than 45 years old.

AGE=45-65: Age between 45 and 65 years old.

AGE>65: Age greater than 65 years old.

SMOKES: Smokes.

EX_SMOKER: Ex-smoker.

FAM_IHD: Family history of ischaemic heart disease.

DIABETES: Diabetes mellitus.

HYPERTENSE: Hypertension.

HYPERLIPID: Hyperlipidaemia.

RETRO: Central chest pain.

LCHEST: Pain in left side of chest.

RCHEST: Pain in right side of chest.

BACK: Pain radiates to back.

LARM: Pain radiates to left arm.

JAW: Pain radiates to neck or jaw.

RARM: Pain radiates to right arm.

BREATHING: Pain is worse on inspiration.

POSTURE: Pain related to posture.

TENDER_CW: Chest wall tenderness

ALLTIGHT: Pain described as tight, heavy, gripping or crushing

ALLSHARP: Pain described as sharp or stabbing.

SWEAT: Sweating

S_O_BREATH: Short of breath.

NAUSEA: Nausea

VOMIT: Vomiting

SYNCOPE: Syncope

EPIS: Episodic pain.

LIKEMI: Worse than usual angina/similar to previous AMI.

LVF: Fine crackles suggestive of pulmonary oedema.

ADDED_HS: Added heart sounds.

HYPOPERF: Signs of hypoperfusion

STELEV: New ST segment elevation.

NEWQ: New pathological Q waves.

STTWAVE: ST segment or T wave changes suggestive of ischaemia.

BBB: Bundle branch block.

OLD_Q: Old ECG features of myocardial infarction.

OLD_ST: ECG signs of ischaemia known to be old.

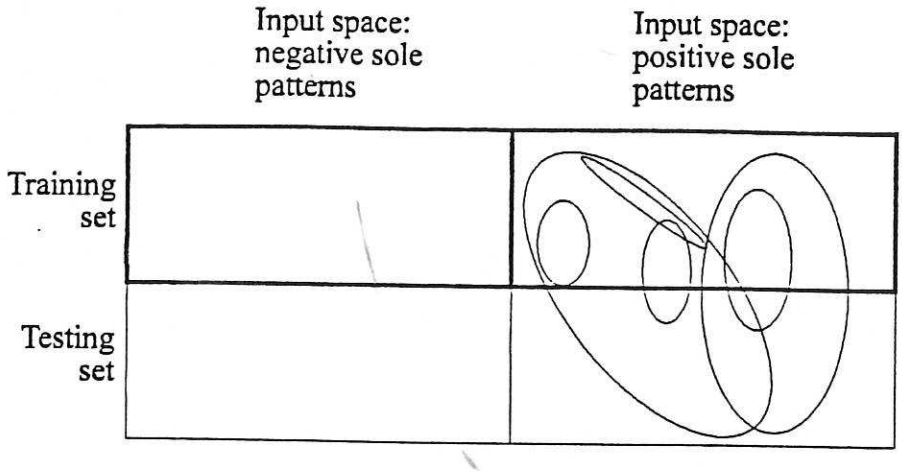

Figure 1 Generality leads to better predictive ability in the testing set range.

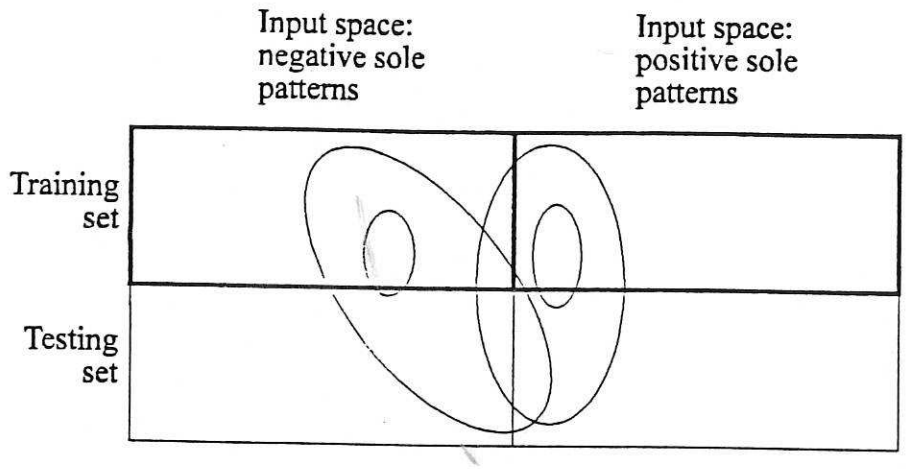

Figure 2 Generality may cause partial mis-classification.

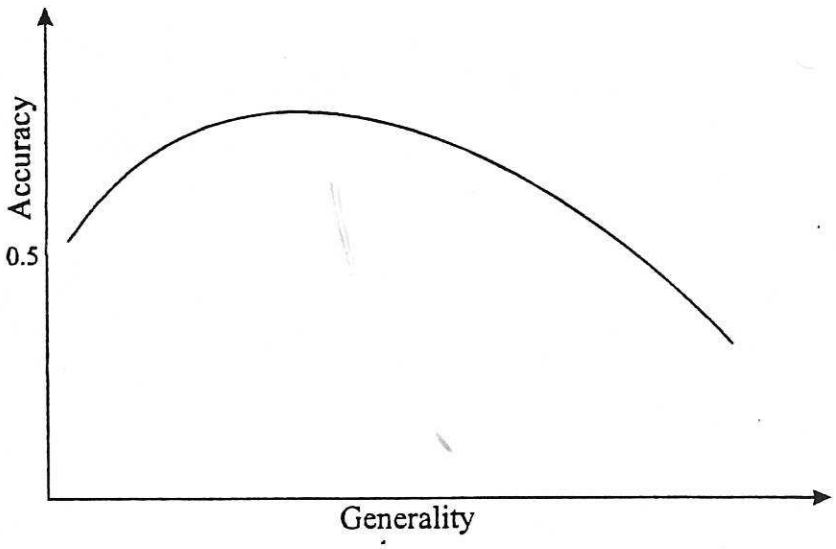

Figure 3 The relationship between generality and accuracy.

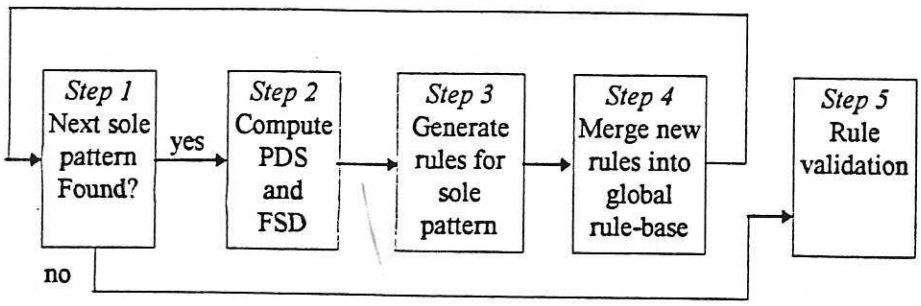

Figure 4 Flow chart for rule extraction.

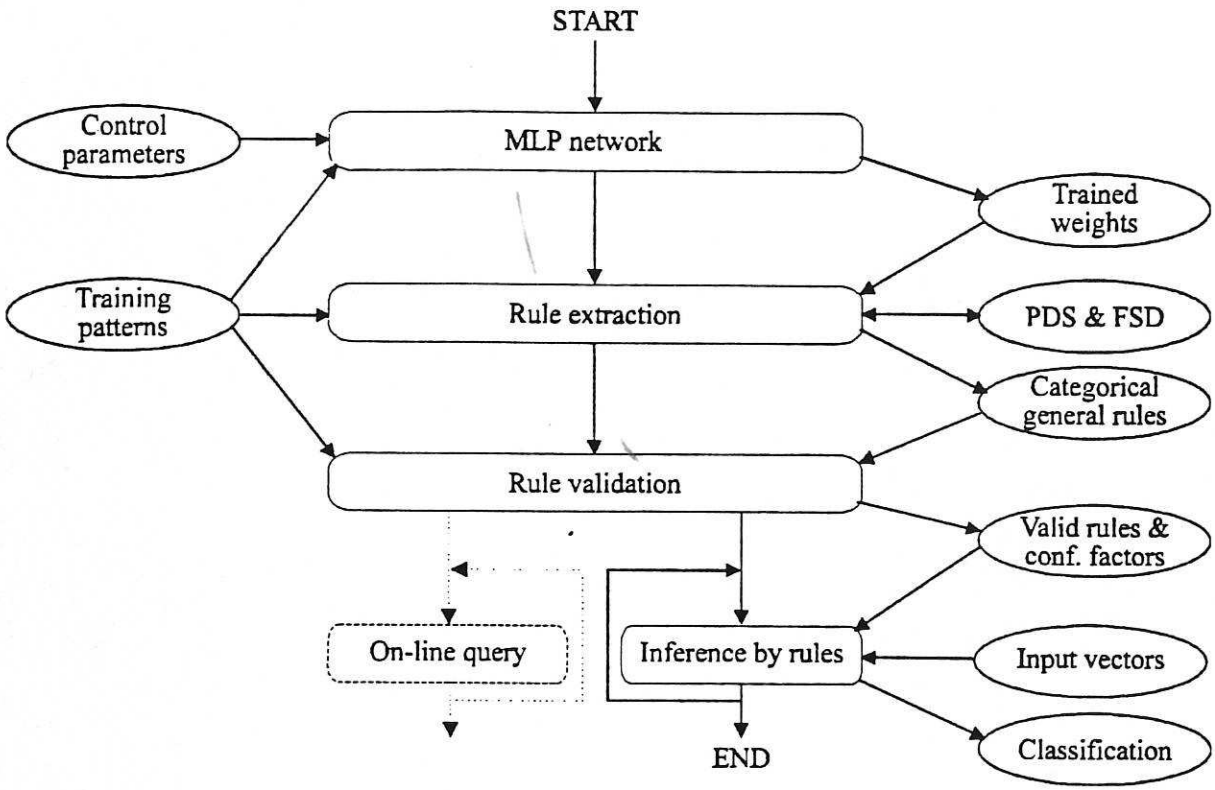

Figure 5 Control and data flow for the GR2 system. Broken lines indicate sub-systems yet to be implemented.

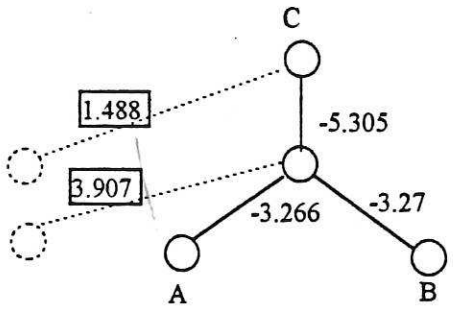

Figure 6 A trained MLP for the two-bit AND data

L_{oi}	I_i	O_o
>0	0	$[1-\delta, 1]$
>0	1	$[0, \delta]$
<0	1	$[1-\delta, 1]$
<0	0	$[0, \delta]$

Table I Situations where *Ii* may be ignorable.

Sub-symbolic form			Symbolic form
A	B	C	
0	0	0	IF(~A, ~B) THEN (~C)
0	1	0	IF(~A, B) THEN (~C)
1	0	0	IF(A, ~B) THEN (~C)
1	1	1	IF(A, B) THEN (C)

Table II Two bit AND truth table

Label	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_{10}	P_9	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}
A	0	1	0	1	0	1	0	1	1	0	0	1	0	1	0	1
B	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
C	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
D	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
E	0	1	0	0	1	1	0	1	1	1	0	1	1	1	1	0

Table III Patterns forming the incomplete 4 bit domain (shaded patterns excluded from the training set)

T_n, T_p	0.1 0.1	0.2 0.2	0.3 0.3	0.5 0.1	0.5 0.4	0.6 0.25	0.6 0.26
R_{tot}	42	23	7	29	7	14	14
R_{val}	32	13	5	21	5	9	8
P_{av}	4.03	2.62	1.6	5.24	1.2	2.78	2.33
P_{sd}	2.98	1.64	0.49	2.99	0.4	0.92	0.82
$Sens_{train}$ (%)	47.8	35.6	7.8	62.2	23.3	81.1	86.7
$Spec_{train}$ (%)	90.3	87.7	90.6	84.8	87.7	85.2	84.5
Acc_{train} (%)	80.8	76.0	72.0	79.8	73.3	84.3	85.0
$Sens_{test}$ (%)	38.2	33.3	8.8	52.0	31.3	84.3	87.3
$Spec_{test}$ (%)	86.3	82,9	86.8	80.6	84.4	85.7	84.6
Acc_{test} (%)	77.7	74.0	72.8	75.4	74.9	85.4	85.1

Table IV GR2: Experimental Results on AMI Records

	MLP	GR2	C4.5	C4.5 Rules
R_{tot}	N/A	14	N/A	21
R_{val}	N/A	8	N/A	5
P_{av}	N/A	2.33	N/A	3.20
P_{sd}	N/A	0.82	N/A	2.11
$Sens_{train}$ (%)	86.7	86.7	63.3	84.4
$Spec_{train}$ (%)	91.6	84.5	98.1	92.9
Acc_{train} (%)	89.1	85.0	91	91.0
$Sens_{test}$ (%)	79.4	87.3	62.8	81.4
$Spec_{test}$ (%)	89.4	84.6	96.6	87.0
Acc_{test} (%)	83.4	85.1	92.3	86.9

Table V Comparison of MLP, GR2 and C4.5 performance.