DETECTION OF BREATHING CIRCUIT FAULTS USING NEURAL NETWORKS

by P.P.M. Ahles

M.Sc. thesis
Work carried out from May 1995 to December 1995
under supervision of:  Prof. Dr. Ir. A. Hasman
                     Drs. B. Müller

The Department of Electrical Engineering of the Eindhoven University of Technology accepts no responsibility for the contents of M.Sc. theses or reports on practical training periods.
Acknowledgments

A lot of people have contributed to the work presented in this thesis. First of all, I would like to thank my supervisors, Prof. dr. ir. A. Hasman and drs. B. Müller for their help, comments, guidance and suggestions.

I would also like to thank Michael Egmont-Petersen for his help concerning neural networks and all of the staff of the division of Medical Electrical Engineering at the Eindhoven University of Technology and the department of Medical Informatics of the Rijksuniversiteit Limburg, Maastricht, where I performed most of the work.

Finally, I would like to thank my wife Angelique, for her love, support and patience.
Abstract

In the Servo Anesthesia project, at the division of Medical Electrical Engineering of the Eindhoven University of Technology, a real-time expert system for assisting patient monitoring and therapy is under development. At present we are concerned mainly with monitoring the breathing circuit, because a large percentage of anesthetic problems are due to faults in artificial ventilators. An intelligent alarm is being developed that can support the anesthesiologist in case of fault situations, by giving an indication of the type of fault and the location of this fault.

At present the intelligent alarm system is built by combining mathematical modeling of the lungs and breathing circuit and inductive machine learning techniques. Using the mathematical model, signal waveforms of flow, pressure and CO$_2$ are simulated. Descriptive features, like minima, maxima, slopes, volumes and time constants are extracted from those signals, and serve as input for the machine learning algorithm, which results in a decision tree.

To provide insight in the correctness of the method used, a comparative study has been performed using neural networks. Using the same inputs as the machine learning algorithm, several neural networks have been trained to classify those inputs into a number of fault situations.

Comparing neural network and decision tree results show that neural networks don't have better performance than decision trees. Neural networks give about the same performance as decision trees. Taking into consideration that neural network training is very time consuming, using decision trees is to be preferred.
## Contents

1 **Introduction** 7
   1.1 Anesthesia safety ............................... 7
   1.2 Alarm systems .................................. 7
   1.3 Intelligent alarm systems ....................... 8
   1.4 Project objective ................................ 8
   1.5 Chapter outline .................................. 8

2 **Breathing circuits** 9
   2.1 Introduction .................................... 9
   2.2 Artificial ventilation ............................. 10
   2.3 The circle breathing circuit ...................... 10
   2.4 Breathing circuit problems ....................... 12

3 **Neural networks** 13
   3.1 Introduction .................................... 13
   3.2 Neural network background and definition ........ 13
   3.3 Description of a neural network .................. 14
   3.4 Weight space .................................... 15
   3.5 Backpropagation neural networks ................. 16
   3.6 Training backpropagation networks ............... 18

4 **Patient data** 19
   4.1 Introduction .................................... 19
   4.2 Obtaining patient data ............................ 19

5 **Experiments** 25
   5.1 Introduction .................................... 25
   5.2 Topology and initial weight configuration ........ 25
   5.3 Determining the correctness of a neural network .. 26
   5.4 Experiment 1: determining the number of hidden PEs . 28
   5.5 Experiment 2: neural networks for the I:E ratios of 1:1, 1:2 and 1:3 28
   5.6 Experiment 3: Using data obtained from signals measured at a mechanically ventilated lung . 29
   5.7 Experiment 4: combining absolute and relative data .. 29

6 **Results** 31
   6.1 Experiment 1 .................................... 31
   6.2 Experiment 2 .................................... 31
Chapter 1

Introduction

1.1 Anesthesia safety

During surgery the anesthesiologist administers drugs to block the patient's pain, to relax the patient's muscles and to induce a state of unconsciousness, while maintaining essential life functions. Systems have been developed to support the anesthesiologist in making decisions based on the patient's current state [Sha91, Sha93, Sno93] and so relieve the anesthesiologist of having to keep track of all the available data. Still, failures in the patient breathing circuit cause many preventable anesthesia accidents [Gab87, Wei90]. Hose disconnects are the most common failure. Other failures include leaks in hose connections and leaks around the endotracheal tube cuff, airway obstructions, and incompetent nonrebreathing valves. All of these problems are harmless if detected and corrected quickly [Aa90, Orr91, Orr94, Sit92, Wat93].

1.2 Alarm systems

An alarm is a warning device designed to call attention to a particular condition or event. Traditionally, alarms used in monitoring devices sound when a monitored signal exceeds a preset threshold. To be useful, the threshold alarm requires the person responding to the alarm to be able to translate the exceeded threshold into information about patient or equipment condition. However, in an average modern operating room, as many as 30 different alarms may go off when (in many instances inappropriate) alarm limits are exceeded. Some say that this is not enough [Blo86], other say it is [Ham86]. The clinical reality is that current alarms often annoy rather than help [Sch86] the anesthesiologist as is evident from the frequency with which alarms are disabled and ignored by anesthesiologists [Aa90]. In order to decrease the burden on the anesthesiologist, the division of Medical Electrical Engineering at the Eindhoven University of Technology is investigating to what extent automatic anesthesia is possible, and the significance of automatic anesthesia [Bas89, Ned90, Oos89]. Results of the research group have been a blood pressure control system and an intelligent alarm system for automatic ventilation [Aa90, Blo90].
1.3 Intelligent alarm systems

With the growing complexity of monitoring, simply displaying a signal either graphically or numerically is not enough [Ben89]. Some form of integration is necessary [Car91, MyI93, Wes92]. To avoid confusion about the meaning of an alarm, every alarm should identify a unique condition or event [Orr91]. Several approaches have been undertaken to accomplish this goal [Ren88], using expert systems [Aa90, Bru89, Fag80, Loe89, Sch88, Sch91, Sha93, Rau91], neural networks [MyI93, Orr89, Orr91, Orr94], fuzzy logic [Bec94], clustering techniques [Oos93], Kalman filtering [Sit90]. In the Servo Anesthesia project, the division of Medical Electrical Engineering at the Eindhoven University of Technology is investigating to what extent automatic anesthesia is possible, and the significance of automatic anesthesia. In one of the sub-projects a real-time expert system for assisting patient monitoring and therapy is under development. This project is concerned with monitoring of the breathing circuit, because as pointed out earlier, anesthesia related deaths are mostly due to faults in the breathing circuit.

1.4 Project objective

In this report an investigation is made whether the use of neural networks for monitoring the breathing circuit is to be preferred over the use of inductive machine learning techniques or not. Comparisons will be made for correctness of classification of breathing circuit faults.

1.5 Chapter outline

Chapter 2 describes the ventilation of the lungs and how breathing circuits work.

In chapter 3 a description will be given of neural networks and how neural networks are trained.

Chapter 4 describes how patient data for training the neural networks is retrieved.

In chapter 5 a description will be given of the experiments that are performed in order to obtain a neural network that correctly can classify breathing circuit faults.

In chapter 6 and 7 the results of the experiments are presented and compared with the results of the inductive machine learning techniques.

Conclusions and recommendations are given in chapter 8.
Chapter 2

Breathing circuits

2.1 Introduction

The respiratory system is concerned with external respiration, the exchange of oxygen and carbon dioxide between alveolar air in the lungs and the blood. The subsequent exchange of oxygen and carbon dioxide between the blood and the cells of the body is known as internal respiration. In addition to external respiration, the respiratory system serves accessory functions such as olfaction and voice production.

The organs that directly serve the function of external respiration are the nose and nasal cavity, pharynx (throat), larynx (voice box), trachea (windpipe), bronchi and the lungs (see fig. 2.1).

During inspiration fresh air enters the human body through the mouth or nose. In the nose the incoming air is warmed, moistened, and filtered. The pharynx is a funnel shaped tube about 13 cm long and serves as a common passageway for both the respiratory and digestive tracts, and to provide a resonating chamber for speech sounds. The larynx which contains the vocal cords, is continuous with the pharynx above and the trachea below. Its two primary functions are to conduct air to and from the lower respiratory tract and to produce sound. The trachea is a tubular passageway for air about 2.5 cm in diameter and 12 cm in length. It extends from the pharynx to the upper boundary of the chest, where it bifurcates into the

Figure 2.1: The organs of the respiratory system. Modified from [Bru79].
right and left primary bronchi. Each bronchus enters into the corresponding lung and divides like the branches of a tree into smaller branches. The bronchi conduct air to and from the substance of the lungs.

The lungs are cone-shaped organs suspended in the thoracic cavity. Each lung is enclosed and protected by a thin membrane called the pleura. The inner layer, the visceral pleura, covers the lungs and the outer layer, the parietal pleura, is attached to the thoracic wall, and together they form a closed sac known as the pleural sac. The two pleural layers are separated by a potential space, called the pleural cavity, that normally contains a thin film of pleural fluid. Pleural fluid permits the lungs to move in their pleural sac without friction during respiration [Bru79].

2.2 Artificial ventilation

As long ago as 1555, when Vesalius first demonstrated thoracic anatomy, the need for artificial respiration was recognized⁰. Until a century ago however, the maintenance of life while the human chest was open posed serious problems. The lungs would immediately collapse, the patient’s respiration would become laboured, the mediastinum² would begin to ‘flap intensely’, and the circulation would become hindered [Ers95]. The precursor of today’s modern ventilators is generally considered to be the one developed by Giertz around 1916. His experiments on animals showed that rhythmic inflation was more effective than either negative or continuous positive pressure ventilation. In 1934 Frenckner, a Swedish ear, nose, and throat surgeon, enlarging upon Giertz’s work, developed the ‘Spiropulsator’, because he was convinced that ‘the rhythmic introduction of air into the trachea under favorable circumstances would be the most ideal type of positive pressure breathing’. Crafoord, working first with Giertz and later with Frenckner, found with further experimentation that the ‘Spiropulsator’ worked most effectively when the patient’s own respiratory efforts were eliminated [Kap83]. He observed that the patient being ventilated had a tendency to ‘fight’ the machine. When eliminating the patient’s own respiration this could be eliminated. In the next section the working of a now widely used breathing circuit is described.

2.3 The circle breathing circuit

To understand a breathing circuit alarm system, it is essential to understand the patient breathing circuit itself. General anesthesia is normally induced and maintained using a combination of inhalational and intravenous agents. The anesthesia machine is used to administer inhalational agents and to maintain respiration when the respiratory drive is suppressed by the agents. In figure 2.2 a schematic diagram of a widely used breathing circuit, the circle breathing

---

1. A history of thoracic anesthesia is described by Kaplan [Kap83].
2. The mediastinum is the mass of tissues and organs situated in the middle of the thoracic cavity.
Figure 2.2: The circle breathing circuit.

circuit, is shown. In this circular setup, during inspiration gas is blown into the patient's lungs by a working pressure on the bellows. This pressure pushes the bellows downward and squeezes out the gas inside. The unidirectional inspiratory and expiratory valves will force the gas to flow through the CO₂ absorber, and will prevent the gas from flowing in the opposite direction. The CO₂ absorber washes out carbon dioxide from the inspiratory gas, which is necessary because inspired gas is rebreathed by the patient. Normally the CO₂ level of expired gas is higher than that of inspired gas due to the production of CO₂ in the tissues. When CO₂ is not removed from the gas before inspiration, the carbon dioxide level will accumulate and become dangerously high. After CO₂ is removed, the gas enters the inspiratory limb of the circuit, where it is mixed with the fresh gas flow. The fresh gas flow comes from the hospital anesthesia machine or from gas supply tanks. It normally consists of a mix of oxygen, nitrous oxide and anesthetic agents. The gas mix enters the lungs via Y-piece and endotracheal tube. During expiration, the external pressure on the bellows is released, and as a consequence of the pressure difference the lungs will empty. Gas flows back into the bellows via the expiratory limb of the circuit where the unidirectional valves again prevent the gas from flowing back into the inspiratory limb. An inflatable cuff around the endotracheal tube keeps the tube firmly in its place and prevents the escape of gas from the lungs to the environment. A spill valve in the ventilator is used to release gas from the circuit. When the bellows is completely filled, this valve opens to let out the excess gas. Because the fresh gas flow is continuously present, the volume of gas inside the circuit would increase and pressures in the circuit would steadily increase if no spill valve would be present.
2.4 Breathing circuit problems

A number of problems with the circle breathing circuit can arise. Primarily these include hose disconnections, leaks, obstructions, and incompetent one-way valves [Orr91].

Gas leaks commonly occur in the area of hose connections or around the endotracheal tube cuff.Leaks in the breathing circuit can cause a reduction in the delivered tidal volume and, if uncorrected, can lead to hypoxia. Leaks around the endotracheal tube cuff have been the cause of hypoventilation leading to serious mishaps.

The most common breathing circuit critical event is a hose disconnection. Disconnections can occur at any hose interconnection in the system.

Another problem is a breathing circuit obstruction. Airway obstructions can lead to excessively high pressures or alternatively to hypoventilation.

Incompetent nonrebreathing valves lead to a build up of CO₂ in the breathing circuit.

In this report only leaks and obstructions will be considered.
Chapter 3

Neural networks

3.1 Introduction

Neural networks is a rediscovered field experiencing an explosive growth in research and application interest [Was88]. In the following sections a description will be given of neural networks and how they are trained to perform some specified task.

3.2 Neural network background and definition

From [Hec89]: From the advent of the first useful electronic digital computer (ENIAC) in 1946 until the late 1980s, essentially all information processing applications used a single basic approach: programmed computing. Solving a problem using programmed computing involves devising an algorithm and/or a set of rules for solving the problem and then correctly coding these in software (and making necessary revisions and improvements).

Clearly, programmed computing can be used in only those cases where the processing to be accomplished can be described in terms of a known procedure or a known set of rules. If the required algorithmic procedure and/or set of rules are not known, then they must be developed — an undertaking that, in general, has been found to be costly and time consuming. In fact, if the algorithm required is not simple (which is frequently the case with the most desirable capabilities), the development process may have to await a flash of insight (or several flashes of insight). Obviously, such an innovation process cannot be accurately planned or controlled. Even when the required algorithm or rule set can be devised, the problem of software development still must be faced.

A new approach to information processing that does not require algorithm or rule development and that often significantly reduces the quantity of software that must be developed has recently become available. This approach, called neurocomputing, allows, for some types of problems (typically in areas such as sensor processing, pattern recognition, data analysis, and control), the development of information processing capabilities for which the algorithms or rules are not known (or where they might be known, but where the software to
implementing them would be too expensive, time consuming, or inconvenient to develop.

The primary information processing structures of interest in neurocomputing are neural networks\(^1\). The formal definition of a neural network follows (cf. [Hec89]).

**Definition:** A neural network is a parallel, distributed information processing structure consisting of processing elements (which can possess a local memory and can carry out localized information processing operations) interconnected via unidirectional signal channels called connections. Each processing element has a single output connection that branches ('fans out') into as many collateral connections as desired; each carries the same signal — the processing element output signal. The processing element output signal can be of any mathematical type desired. The information processing that goes on within each processing element can be defined arbitrarily with the restriction that it must be completely local; that is, it must depend only on the current values of the input signals arriving at the processing element via impinging connections and on values stored in the processing element's local memory.

### 3.3 Description of a neural network

Neural networks are built of processing elements (PEs). PEs are usually arranged in layers, and the PEs in a layer are often connected to many PEs in other layers, to PEs in their own layer or even to themselves. Each PE processes the input it receives via these connections, and provides a continuous analog value to other PEs via its outgoing connections. As in biological systems, the strength of these connections can change when a network is trained using some learning law.

There are two characteristics that help us divide neural networks into a few basic categories:

- whether the data flows through a network in the forward direction only, as opposed to both forward and backward (through separate connections);
- whether the network is given the correct answer during training, or whether the network is left to figure this out for itself.

Networks where data flows only in the forward direction are called *feed-forward networks*. They are very popular due to their relative simplicity and stability. Those networks with connections that allow data to flow forward and backward are called *feedback networks*. As is common knowledge, a feedback system can work quite nicely, or can sometimes exhibit unstable oscillations (e.g. an adjustable thermostat to heat your home where the current temperature is fed back into the thermostat: one minute too cold, the next minute too hot...).

---

1. Neural networks are sometimes called artificial neural networks, since they are human attempts to simulate and understand what goes on in human nervous systems. In this report the term neural network will denote an artificial neural network unless otherwise stated.
In this report only feed-forward networks are considered: data flows only in the forward direction.

The second characteristic used to classify a network type is how the network is trained. The most common method for training a network is to provide the network with the input and desired output values of several examples, and using some learning law to adjust the strength or weight of the connections so that the values of the output PEs come as close as possible to the desired values, that is to make sure the network makes as little an error as possible. This type of learning is called *supervised learning*.

Opposing *supervised learning* there are two other types of learning. The first is called *graded* or *reinforcement* training. The network now is given data inputs, but is not supplied with the outputs. Instead, it is occasionally given a *grade* or *performance score* that tells it how well it has done overall since the last time it was graded. The second type is *self-organization*. The network now is provided only with the input values of several examples. Using some learning law the weights are adjusted in such a way that after a period of training the network will cluster the data provided at its inputs. In other words, the network will — after some time — have divided up the training set of examples into a number of classes: training examples that look alike will, when provided to the network, cause approximately the same values of the output PEs.

In this report only supervised learning is considered: the network will, when being trained, receive both inputs and desired outputs.

### 3.4 Weight space

To visualize the complexity of the task of training a network, imagine a network consisting of a number of PEs with \( n \) connections altogether. Each connection has a weight associated with it. We can now write a network weight vector \( \mathbf{w} \) as

\[
\mathbf{w} = (w_1, w_2, w_3, \ldots, w_n)^T
\]

The set of all possible \( \mathbf{w} \) vectors determines the set of all possible information processing configurations for this network. In other words, *if the information processing performance we seek is to be realized by this network, it will be found at some value of the vector \( \mathbf{w} \)*.

The challenge is to develop a learning law that will efficiently guide the weight vector \( \mathbf{w} \) to a location that yields the desired network performance.

There are many different approaches to this problem. Most learning laws are formulated with a specific goal in mind. A commonly encountered type of goal is to move \( \mathbf{w} \) to a position that yields a network that minimizes or maximizes some particular global neural network cost or performance function, such as mean squared error, net profit, fuel consumed, or average pattern classification error.

This type of law is the one that will be used to train the networks described in this report.
3.5 Backpropagation neural networks

For the networks described in this report we will make use of 'backpropagation neural networks'. It is a powerful mapping network that has been successfully applied to a variety of problems ranging from credit application scoring to image compression [Hec89].

The backpropagation neural network architecture is a hierarchical design consisting of fully interconnected layers or rows of PEs (see fig. 3.1). The information processing operation that backpropagation networks are intended to carry out is the approximation of a bounded mapping or function \( f : A \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m \) from a compact subset \( A \) of \( n \)-dimensional Euclidian space to a bounded set \( f[A] \) of \( m \)-dimensional Euclidian space, by means of training on examples \((x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k), \ldots\) of the mapping where \( y_k = f(x_k) \).

The first layer — the input layer — consisting of \( n \) PEs, takes as input a \( n \)-dimensional vector. The output of the PEs in the input layer, multiplied by the weights attached to the connections, is passed to the PEs in the hidden layer [Egm94].

A hidden PE collects the incoming weighted output values of the previous layer. Besides that, it receives also the weighted value of a bias PE (see fig. 3.1). This bias PE always outputs the value 1. It allows for adding an offset to the sum of
the weighted inputs, similar to an offset in a regression or discriminant function.

The sum of the weighted input values is passed through a nonlinear transfer function (see fig. 3.2). Various types of transfer functions have been proposed, such as sigmoidal, hyperbolic-tangent or logistic functions. The only requirements are that the output values of the function are bounded to an interval and that the nonlinear function can be differentiated. To avoid saturation of the nonlinear functions during training, the total input activation has to be bounded. This can be achieved in two ways. One either scales the weights from each input PE such that the orders of magnitude of the input value and the weights are reciprocal to each other. The other way is to scale the input to a certain range. In the networks used in this report the inputs values were scaled to the interval \([0,1]\).

The output of a PE in the hidden layer is fed into the PEs of the next layer, again multiplied by the weights attached to the connections. In principle, several hidden layers may be used. However, it has been shown that any arbitrary discriminant function can be build with a classifying neural network with only one hidden layer, if enough hidden PEs are used [Min83]. The PEs of the output layer have the same structure as the PEs of the hidden layer.

What remains to be done, is to define how the required output — in this case class membership — is mapped on the output range of the function. When the limit values of the transfer functions are used, the function has to be in saturation to achieve a good classification. This is often not possible. Therefore, several researchers have proposed to do a different mapping of the desired values, for example on \([-0.9,0.9]\) or on \([0.1,0.9]\). In the experiments in chapter 5 the sigmoidal function is used as the transfer function of the PEs and the desired outputs are mapped on 0.2 (not belonging to the class) and 0.8 (belonging to the class).
3.6 Training backpropagation networks

When training a backpropagation neural network we want to update the weights in the network for every training example in such a way that the error on the output PEs gets smaller. In other words, given a certain \( w \) vector we want to move the vector to a place on the error surface were the error gets smaller. This calls for a gradient descent method. The most widely used method makes use of the 'generalized delta rule'. For a full description see [Rum86]. Several others proposed other methods for training the backpropagation network, mostly to improve the speed of learning [Bab94, Fah88, Sit92]. The basic idea is that the weights are updated after every training example is presented to the network. The weights are updated using the following formula:

\[
\begin{align*}
    w'_{ij} & = w_{ij} + C_1 \cdot e_t \cdot x_{ij} + C_2 \cdot m_{ij} \\
    m'_{ij} & = w'_{ij} - w_{ij}
\end{align*}
\]

The weights are changed in proportion to the error \( e \), and the input to that connection \( x \). Because the error surface of a backpropagation neural network usually has several local minima, a second term is added called the momentum term, which is used to 'smooth' out the weight changes.

In the output layer, the error \( e \) is the current error transformed by the derivative of the transfer function. In the hidden layer it is the accumulated, transformed backpropagated error.
Chapter 4

Patient data

4.1 Introduction

In order to obtain data for training the neural networks, we need to perform measurements at a breathing circuit with which a patient is ventilated. This is not always easy, as we cannot introduce anesthetic mishaps on purpose in order to study their influence on vital signals that are measured at the patient — at least not not when ventilating a human. Müller did perform measurements on dogs, but a lot of measurements need to take place if enough data is to be gathered. This technique can be applied, however, if we use a substitute for the patient. We can use a mechanical lung model, and ventilate this using a breathing circuit during several malfunctions. This method is rather expensive, however, because we need a complete breathing circuit including a ventilator, which often has to be modified for the experiments. Instead, a mathematical model of the patient and breathing circuit can be used. If the model is accurate enough and validated, vital signals that are normally measured at the bedside, can now be simulated with the model during normal functioning and during different fault situations.

4.2 Obtaining patient data

For the experiments described in later chapters, a comprehensive set of patient data is generated with a mathematical model, written in PASCAL. These data consist of airway pressure, expiratory flow and CO₂ signals, measured at the circle breathing circuit (see fig. 4.1), during the normal, fault-free situation and during one of twelve fault situations. The modeled patient data group was divided into a training set and a test set. The training set consisted of 54 patients, with airway resistance varying between 1.0 and 2.0 cmH₂O/l/s (step size 0.2 cmH₂O/l/s) and lung/thorax compliance varying between 0.02 and 0.1 l/cmH₂O (step size 0.01 l/cmH₂O). The test set consisted of 40 patients with resistance and compliance values that also fell within these intervals, but were independent from the training set: resistance ranged from 1.1 to 1.9 cmH₂O/l/s and compliance ranged from 0.025 to 0.095 l/cmH₂O with the same step sizes as in the training set. For each patient, a breath-to-breath random error was added to the airway resistance and lung/thorax compliance, so that no two consecutive
breaths for one patient would be exactly the same. This is in accordance with real respiratory data, where breath-to-breath variations are found in airway resistance and lung/thorax compliance values that are calculated from the data. The random errors added to resistance and compliance values in the modeled patients have variances based upon the variance measured in resistance and compliance values from data from mechanically ventilated sheep. These variances are assumed to be valid for humans as well, because the lung anatomy of sheep is comparable to that of (small) humans.

The faults that were introduced in the model are: large and small inspiratory hose leaks, large and small expiratory hose leaks, large and small Y-piece leaks, large and small inspiratory hose obstructions, large and small expiratory hose obstructions and large and small Y-piece obstructions. Each patient was ventilated with a respiratory rate of 10 breaths/minute and a tidal volume of 350 ml. First normal signals were recorded. Then for a time interval consisting of seven breaths, one of the twelve faults was introduced, followed by an interval of five normal breaths, which proved to be sufficient for the patient/circuit to return to the pre-fault state. The process went on until all faults had been introduced once for a patient. The first eight normal breaths and the five middle breaths of each fault were used for training the neural network. The first normal breath served as a reference for all other normal and faulty signals. The normal breaths between faults were not included in the training set.

From each breath a set of signal features was derived from the three signals (see fig. 4.2, 4.3 and 4.4). The complete set of features is listed in table 4.1.
Figure 4.2: The flow signal with the extracted signal features

Figure 4.3: The pressure signal with the extracted signal features
If one or more of the features was invalid (when it could not be identified by the parameter extraction algorithm), or one or more of the three signal waveforms could not be identified at all, the breath was not used in the training experiments. In table 4.2 the sizes of the training and test sets are shown.

Three sets of features were created:

- a relative feature set, in which the signal feature values were divided by the corresponding signal features of the reference normal signal;
- an absolute feature set, in which the feature values of the reference normal signal were subtracted from the corresponding features of the current signal;
- a combined relative and absolute parameter set.

Table 4.1: Features extracted from the signal waveforms

<table>
<thead>
<tr>
<th>Expiratory flow</th>
<th>CO₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimum flow</td>
<td>minimum CO₂</td>
</tr>
<tr>
<td>maximum flow</td>
<td>maximum CO₂</td>
</tr>
<tr>
<td>inspiratory time</td>
<td>inspiratory time</td>
</tr>
<tr>
<td>expiratory time</td>
<td>expiratory time</td>
</tr>
<tr>
<td>mean flow</td>
<td>up slope</td>
</tr>
<tr>
<td>expired volume</td>
<td>down slope</td>
</tr>
<tr>
<td>time constant down slope</td>
<td>plateau slope</td>
</tr>
</tbody>
</table>

Table 4.2: Sizes of the training and test sets

<table>
<thead>
<tr>
<th>Feature set</th>
<th>Training Set</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relative</td>
<td>100%</td>
<td>50%</td>
</tr>
<tr>
<td>Absolute</td>
<td>90%</td>
<td>45%</td>
</tr>
<tr>
<td>Combined</td>
<td>80%</td>
<td>40%</td>
</tr>
</tbody>
</table>

Figure 4.4: The CO₂ signal with the extracted signal features
Table 4.2: Sizes of training and test sets. Relative and absolute sets have equal sizes. The maximum size of a training set is 3510, and the maximum size of a test set is 2600.

<table>
<thead>
<tr>
<th>I:E ratio</th>
<th>training set</th>
<th>test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:1</td>
<td>3376</td>
<td>2506</td>
</tr>
<tr>
<td>1:2</td>
<td>3440</td>
<td>2546</td>
</tr>
<tr>
<td>1:3</td>
<td>3423</td>
<td>2543</td>
</tr>
</tbody>
</table>

To investigate whether the I:E (inspiration time over expiration time) ratio has any influence on the learning process, three different training and test sets were created: for an I:E ratio of 1:1, for an I:E ratio of 1:2 and for an I:E ratio of 1:3. The I:E ratio is a setting of the ventilator, just as tidal volume and respiratory rate.
Chapter 5

Experiments

5.1 Introduction

In order to obtain a neural network with good performance, a great number of neural networks have to be trained. These neural networks differ in initial weight configuration and size or topology (mainly the number of hidden PEs). This chapter explains how these different neural networks are constructed and which experiments were performed to obtain neural networks for detecting breathing circuit faults.

5.2 Topology and initial weight configuration

The number of hidden PEs to use in a neural network is hard to determine. There are no real rules to determine it. Only one rule has to be accounted for: the number of weights in the neural network cannot be greater than the number of cases in the training set.

Using this rule a neural network can be constructed as follows: let the number of cases in the training set be $c$, the number of input PEs $i$ and the number of output PEs (classes) $o$. The number of hidden PEs will be denoted by $h$. The number of weights from the input layer to the hidden layer, $i + 1$ (input PEs plus bias PE) multiplied by $h$, plus the number of weights from the hidden layer to the output layer, $h + 1$ (hidden PEs plus bias PE) multiplied by $o$, cannot be greater than $c$, or

$$(i + 1) \times h + (h + 1) \times o \leq c \quad (5.1)$$

This implies that the number of hidden PEs has an upper limit

$$h \leq h_{\text{max}} = \frac{c - o}{i + o + 1} \quad (5.2)$$

Neural networks with a different topology can now be constructed by using only a limited number of hidden PEs (less than $h_{\text{max}}$), since in general using the maximum number of hidden PEs is not needed.

An example:
The training set to be used contains 3440 cases. The number of input PEs is 15
and the number of output PEs is 13. This gives an upper limit for the number of hidden PEs of

\[
h_{\text{max}} = \frac{3440 - 13}{15 + 13 + 1} = \frac{3427}{29} = 118.7
\]

Of course, this must be an integer number, so it is rounded towards zero (otherwise the number of weights will be more than the number of cases). So

\[
h_{\text{max}} = 118
\]

As explained in chapter 3 we want the weight vector \( \mathbf{w} \) to point to a location that yields the desired network performance, that is the location that yields a minimum for the mean squared error of the output PEs. Therefore we train a number of networks with the same number of hidden PEs and the same training set but with different initial weight configurations hoping that one of these configurations is close to the global minimum of the error surface for this network.

In combining the different initial weight configurations and the topology a large number of neural networks is obtained. All these neural networks are trained and the results for each neural network are compared with the results of the other neural networks.

### 5.3 Determining the correctness of a neural network

After a network is trained some sort of quality test has to be performed, in order to determine which network performs best. For this purpose a test set is used. Every case in the test set is provided to the network and the result of the classification is compared with the desired classification. This can be shown in a confusion matrix or contingency table (see table 5.1). This is the matrix belonging to an actual network described in experiment 2. The rows represent the desired classification, the columns represent the classification of the network. In the matrix we can see for instance that 193 small inspiratory leaks (sil) are classified as a small Y-piece leak (syl), 6 as a large expiratory leak (lel) and 1 as a large Y-piece leak (lyl). If a quality measure has to be assigned we can look at the percentage of correct classifications. In the matrix we sum the elements on the main diagonal and divide this by the total of all classifications. In this report this score will be called 'score 1'. As is visible in the matrix the neural network has trouble in distinguishing small inspiratory, expiratory and Y-piece leaks from one another. The same goes for large leaks. For this reason Müller used another score: score 2. Score 2 is the percentage of correct classifications when small leaks and large leaks each are considered as one single fault class. Score 1 will be relatively low because of the indistinguishability of the different small and large leaks. Due to the location in the breathing circuit where the different signals are measured (all between Y-piece and valves), it is almost impossible to see the difference between inspiratory, expiratory and Y-piece leaks. The effects
Table 5.1: Confusion matrix for the 1:1 absolute network, tested with the 1:1 test set. Rows represent the true classes, columns represent the classification made by the neural network. Score 1 = 71.2 %, score 2 = 99.1 %. (sil=small inspiratory leak, sel=small expiratory leak, syl=small Y-piece leak, lil=large inspiratory leak, lel=large expiratory leak, lyl=large Y-piece leak, sio=small inspiratory obstruction, seo=small expiratory obstruction, syo=small Y-piece obstruction, lio=large inspiratory obstruction, leo=large expiratory obstruction, lyo=large Y-piece obstruction)

<table>
<thead>
<tr>
<th></th>
<th>sil</th>
<th>sel</th>
<th>syl</th>
<th>lil</th>
<th>lel</th>
<th>lyl</th>
<th>sio</th>
<th>seo</th>
<th>syo</th>
<th>lio</th>
<th>leo</th>
<th>lyo</th>
<th>nor</th>
</tr>
</thead>
<tbody>
<tr>
<td>sil</td>
<td>0</td>
<td>0</td>
<td>193</td>
<td>0</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sel</td>
<td>0</td>
<td>0</td>
<td>197</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>syl</td>
<td>0</td>
<td>0</td>
<td>194</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>lil</td>
<td>0</td>
<td>0</td>
<td>149</td>
<td>20</td>
<td>31</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lel</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>98</td>
<td>72</td>
<td>27</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lyl</td>
<td>0</td>
<td>0</td>
<td>109</td>
<td>25</td>
<td>66</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sio</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>seo</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>syo</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lio</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>leo</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>112</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lyo</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>114</td>
<td>0</td>
</tr>
<tr>
<td>nor</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>276</td>
</tr>
</tbody>
</table>
that the leaks have on the flow, pressure and CO₂ signals we measure, are the
same for each type of leak. Small and large leaks on the other hand, are easily
distinguishable as two separate classes, each of them containing leaks at three
different locations. Score 2 will therefore be much higher than score 1.

5.4 Experiment 1: determining the number of hidden PEs

In order to determine the number of hidden PEs needed in the neural networks,
a number of networks are trained with the same training set. The number of
hidden PEs was varied by using 40 %, 50 % and 60 % of the maximum number
of hidden PEs for this network. As turned out by testing the networks, the
classification capabilities of the network using these numbers of hidden PEs
were comparable. For this reason another series of networks with only 5 %
and 10 % of the maximum number of hidden PEs were trained with the same
training set.

5.5 Experiment 2: neural networks for the I:E ratios of 1:1, 1:2
and 1:3

Several networks with the number of hidden PEs equal to 10 % of the maxi­
mum number of hidden PEs1 were trained with six different training sets: three
training sets for the relative features (I:E ratio of 1:1, 1:2 and 1:3) with sizes of
3376, 3440 and 3423 cases and three training sets for the absolute features (I:E
ratio of 1:1, 1:2 and 1:3) with the same sizes as for the relative features.

For the relative features 15 inputs and 13 outputs are used, for the absolute
features 19 inputs and 13 outputs are used.

In order to separate the influence of changing topology from that of changing
the initial weight configuration in the best possible way, a set of ten different
initial weights were assigned to the neural network with most weights. Neural
networks with less weights use the appropriate subset of weights and biases.
E.g. in figure 5.1 a neural network is shown using four input PEs, three hidden
PEs and three output PEs (example adapted from [Egm94]). The weights with
dotted connections are not used in neural networks with only two hidden PEs.

The obtained networks are tested by providing the test sets for every I:E ratio
to every network, that is a network trained with the 1:1 training set is tested
with the corresponding 1:1 test set but also with the test sets for I:E ratios of
1:2 and 1:3 (analogous for 1:2 and 1:3).

1. The maximum number of hidden PEs differs for each training set, since the sizes of the
   training sets are all different.
5.6 Experiment 3: Using data obtained from signals measured at a mechanically ventilated lung

The networks in the previous section that performed best on their corresponding test set, were also tested with data obtained from signals measured at a mechanically ventilated lung — the Gainesville Anesthesia Simulator. There were only 7 fault classes: inspiratory leak, expiratory leak, Y-piece leak, inspiratory obstruction, expiratory obstruction, y-piece obstruction and normal. Because the networks were trained to classify into 13 classes, the test set was adapted. The leaks were mapped to large leaks and the obstructions were mapped to large obstructions. In the results some adaptation had to be made too. Small and large inspiratory leaks were combined, as were small and large expiratory leaks, small and large Y-piece leaks, small and large inspiratory obstructions, small and large expiratory obstructions and small and large Y-piece obstructions.

5.7 Experiment 4: combining absolute and relative data

A new 1:1 training set was constructed using both absolute and relative features. The networks now had 34 input PEs. Ten networks were trained and afterwards tested with three test sets with combined features (for I:E ratios of 1:1, 1:2 and 1:3).
Chapter 6

Results

6.1 Experiment 1

Several networks with varying numbers of hidden PEs were all trained with the same training set. Per number of hidden PEs ten networks were trained, so in all 50 networks had to be trained. They all were tested with the same test set with corresponding I:E ratio. The results are shown in table 6.1.

Table 6.1: Mean correctness for networks with different hidden layer sizes all tested with the same test set. For every hidden layer size 10 networks were trained.

<table>
<thead>
<tr>
<th>number of hidden PEs (perc. of maximum)</th>
<th>5 %</th>
<th>10 %</th>
<th>40 %</th>
<th>50 %</th>
<th>60 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>score 1</td>
<td>65.7 %</td>
<td>68.0 %</td>
<td>68.6 %</td>
<td>67.9 %</td>
<td>68.4 %</td>
</tr>
<tr>
<td>score 2</td>
<td>95.9 %</td>
<td>98.4 %</td>
<td>98.8 %</td>
<td>98.5 %</td>
<td>99.0 %</td>
</tr>
</tbody>
</table>

6.2 Experiment 2

A set consisting of 10 networks was trained for each of the three I:E ratios with the absolute features, as was a set of 10 networks with the relative features, summing to 60 networks. Every network was tested with the corresponding test set and with the other two test sets. In table 6.2 the results are shown of the test with the corresponding test set. Only the scores of the ‘best’ network are presented. In table 6.3 the results are shown of the tests with the other test sets. These are the scores from the same networks as in table 6.2. For the purpose of making a comparison between the results of neural networks and decision trees, the results obtained by Müller are also shown.

6.3 Experiment 3

The networks from the previous experiment that performed best on their corresponding test set were tested with data obtained from signals measured at the
Table 6.2: Scores for the best networks, each trained with a different training set and tested with the corresponding test set.

<table>
<thead>
<tr>
<th>trainingset</th>
<th>parameters</th>
<th>neural network</th>
<th>decision trees</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>score 1</td>
<td>score 2</td>
</tr>
<tr>
<td>1:1</td>
<td>absolute</td>
<td>71.2 %</td>
<td>99.1 %</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>67.5 %</td>
<td>98.0 %</td>
</tr>
<tr>
<td>1:2</td>
<td>absolute</td>
<td>68.6 %</td>
<td>99.0 %</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>71.1 %</td>
<td>99.7 %</td>
</tr>
<tr>
<td>1:3</td>
<td>absolute</td>
<td>69.5 %</td>
<td>99.7 %</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>69.1 %</td>
<td>97.2 %</td>
</tr>
</tbody>
</table>

Table 6.3: Scores for the same networks as in table 6.2 when tested with the test set of the other I:E ratios.

<table>
<thead>
<tr>
<th>trainingset</th>
<th>testset</th>
<th>parameters</th>
<th>neural network</th>
<th>decision trees</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>score 1</td>
<td>score 2</td>
</tr>
<tr>
<td>1:1</td>
<td>1:2</td>
<td>absolute</td>
<td>66.3 %</td>
<td>93.0 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
<td>63.9 %</td>
<td>85.5 %</td>
</tr>
<tr>
<td></td>
<td>1:3</td>
<td>absolute</td>
<td>57.7 %</td>
<td>77.2 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
<td>59.1 %</td>
<td>76.1 %</td>
</tr>
<tr>
<td>1:2</td>
<td>1:1</td>
<td>absolute</td>
<td>67.0 %</td>
<td>98.0 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
<td>63.7 %</td>
<td>90.9 %</td>
</tr>
<tr>
<td></td>
<td>1:3</td>
<td>absolute</td>
<td>66.8 %</td>
<td>90.7 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
<td>68.9 %</td>
<td>97.8 %</td>
</tr>
<tr>
<td>1:3</td>
<td>1:1</td>
<td>absolute</td>
<td>55.8 %</td>
<td>85.7 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
<td>58.9 %</td>
<td>88.3 %</td>
</tr>
<tr>
<td></td>
<td>1:2</td>
<td>absolute</td>
<td>70.7 %</td>
<td>99.9 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
<td>67.8 %</td>
<td>98.9 %</td>
</tr>
</tbody>
</table>

**Gainesville Anesthesia Simulator** [Aa90], a mechanical lung that is ventilated with a circle breathing circuit and that produces CO₂. These signals were recorded during normal operation and malfunctioning breathing circuit performance. In table 6.4 the classification performance of the 6 networks tested on the measured signals are shown.

### 6.4 Experiment 4

Using a 1:1 training set with combined absolute and relative features 10 networks were trained. They were tested with combined absolute and relative test sets for I:E ratios of 1:1, 1:2 and 1:3. In table 6.5 the results are shown. The results from test sets 1:2 and 1:3 are from the network that performed best on the 1:1 test set.
Table 6.4: Scores for the same networks as in experiment 2, when tested with the data from the Gainesville Anesthesia Simulator.

<table>
<thead>
<tr>
<th>trainingset</th>
<th>parameters</th>
<th>neural network</th>
<th>decision trees</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>score 1</td>
<td>score 2</td>
</tr>
<tr>
<td>1:1</td>
<td>absolute</td>
<td>43,7 %</td>
<td>64,8 %</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>25,4 %</td>
<td>49,3 %</td>
</tr>
<tr>
<td>1:2</td>
<td>absolute</td>
<td>40,9 %</td>
<td>66,2 %</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>42,3 %</td>
<td>66,2 %</td>
</tr>
<tr>
<td>1:3</td>
<td>absolute</td>
<td>54,9 %</td>
<td>77,5 %</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>31,0 %</td>
<td>63,4 %</td>
</tr>
</tbody>
</table>

Table 6.5: Scores for the best networks when trained with a training set consisting of both absolute and relative features. The scores for the test sets with 1:E ratio of 1:2 and 1:3 are from the best network when tested with an 1:E ratio of 1:1.

<table>
<thead>
<tr>
<th>trainingset</th>
<th>testset</th>
<th>parameters</th>
<th>neural network</th>
<th>decision trees</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1:1</td>
<td>absolute + relative</td>
<td>68,0 %</td>
<td>99,5 %</td>
</tr>
<tr>
<td></td>
<td>1:2</td>
<td>absolute + relative</td>
<td>66,7 %</td>
<td>94,9 %</td>
</tr>
<tr>
<td></td>
<td>1:3</td>
<td>absolute + relative</td>
<td>60,4 %</td>
<td>76,6 %</td>
</tr>
</tbody>
</table>
Chapter 7

Discussion

7.1 Experiment 1

From the first experiment we can conclude that using 10% of the maximum number of hidden PEs in this case is sufficient for training neural networks. When using less than 10%, correctness of the networks starts to drop, when using more, correctness approximately stays the same. Since training neural networks is very time consuming we want to use networks that are as small as possible, but still perform good enough.

7.2 Experiment 2

In experiment 2 several networks were trained. The best network trained with a particular training set was chosen as the network that performed best on the corresponding test set. The reason for this is that we wanted to investigate the influence of the I:E ratio on the networks.

As is shown in table 6.2 the neural networks don't perform better than the decision trees when they are tested with the test set with the same I:E ratio as the training set. When looking at table 6.3 we see that the neural networks trained with a particular training set perform better when tested with a test set of another I:E ratio than the decision trees. Apparently the inspiration time and expiration time features have less influence in neural networks than in decision trees.

7.3 Experiment 3

Looking at table 6.4, we see that the neural networks perform badly on the signals from the Gainesville Anesthesia Simulator compared to the results of the decision trees. The reason for this can be that neural networks are more sensitive to differences in the features between the Simulator data and the mathematical model data.

Decision trees can classify a breath using only a limited number of features, while neural networks use all features even if the associated weights in the
network are low. When checking the neural networks for the input which had most effect on the output values it turned out that this was the flow time constant. When this feature was varied slightly the output values changed dramatically. This can lead to misclassifications especially with expiratory and Y-piece obstructions. In the decision trees this feature had almost no influence when classifying the signals, so better scores for decision trees are explainable.

7.4 Experiment 4

When the absolute and relative features are combined in one training set the same thing happens as in experiment 2. The network does not perform better when tested with the corresponding test set, but does when tested with the other two test sets. Because there was no time, the other two training experiments with an I:E ratio of 1:2 and 1:3 could not be performed. However, looking at the previous results it is safe to predict that these networks will show the same behaviour.
Chapter 8

Conclusions and recommendations

Training neural networks in a way that performance is good is difficult. Choosing the size of the network is very important: if the network is too small performance may degrade, if it is too large training time is greatly increased. Carefully choosing the right size may save a lot of time.

Of big importance also is the fact that simply training one network isn’t good enough. Several networks with different initial weight configurations have to be compared for their performance, since the risk of becoming trapped in a local minimum of the error surface of the network is high. If lots of networks are trained the chance of finding a network with good performance gets better, but the chance of finding one that has a performance that is worse is still high. Patience in this case is a good thing, but this regretfully depends on the time one has.

Looking at the time needed to train a neural network, the results obtained with neural networks aren’t better than or at most equal to the results obtained with decision trees. Using inductive machine learning techniques for this particular problem — automatic generation of breathing circuit alarms — has to be preferred over the use of neural networks.

Müller also performed experiments with training sets consisting of features from combined I:E ratios. Further experiments with neural networks have to be performed to examine whether neural networks and decision trees perform equally well in this case too. Preliminary testing show this is true, but this is based on performance scores of only a few neural networks.
References

[Aa90] van der Aa, J. J. L. C. M.
INTELLIGENT ALARMS IN ANESTHESIA: a real time expert system application.

[Bab94] Baba, N. and Y. Mogami, M. Kohzaki, Y. Shiraishi, Y. Yoshida.
A HYBRID ALGORITHM FOR FINDING THE GLOBAL MINIMUM OF ERROR FUNCTION OF NEURAL NETWORKS AND ITS APPLICATIONS.

[Bas89] Bastings, R. H. A.
TOWARD THE DEVELOPMENT OF AN INTELLIGENT ALARM SYSTEM IN ANESTHESIA.

A FUZZY LOGIC APPROACH TO INTELLIGENT ALARMS IN CARDIOANESTHESIA.
Volume 3.

ALARMS AND THEIR LIMITS IN MONITORING.

[Bl86] Block, Jr, F. E.
DO WE MONITOR ENOUGH? WE DON’T MONITOR ENOUGH.

[Bl90] Blom, J. A.
THE SIMPLEXYS EXPERIMENT: real time expert systems in patient monitoring.

[Br79] Bruck-Kan, R.
INTRODUCTION TO HUMAN ANATOMY.

PROTOTYPE VENTILATOR AND ALARM ALGORITHM FOR THE NASA SPACE STATION.
PROGRESS WITH MEASUREMENT, INFORMATION AND DECISION-MAKING IN CRITICAL CARE MEDICINE.

ON THE QUALITY OF NEURAL NET CLASSIFIERS.
Artificial Intelligence in Medicine, Vol. 6(1994), p. 359-381.

Ersoy, G. A.
ELECTRICAL MODELS OF BREATHING CIRCUITS FOR MONITORING VARIOUS FAULTS BY USING PSPICE.
Ms.C. thesis, Eindhoven University of Technology, Faculty of Electrical Engineering, Division of Medical Electrical Engineering, June 1995.

Fagan, L. M.
VM: representing time-dependent relations in a medical setting.

Fahlman, S. E.
AN EMPIRICAL STUDY OF LEARNING SPEED IN BACK-PROPAGATION NETWORKS.

ANESTHETIC MISHAPS: breaking the chain of accident evolution.

Hamilton, W. K.
DO WE MONITOR ENOUGH? WE MONITOR TOO MUCH.

Hecht-Nielsen, R.
NEUROCOMPUTING.
Reading, MA, Addison-Wesley, 1989.

Kaplan, J. A.
THORACIC ANESTHESIA.

THE UTAH ANESTHESIA WORKSTATION.
Anesthesiology, Vol. 70(1989), No. 6, p. 999-1007.

Minsky, M. and S. Papert.
PERCEPTRONS.

INTEGRATION OF MONITORING FOR INTELLIGENT ALARMS IN ANESTHESIA: neural networks — can they help?

Nederstigt, J. A.
DESIGN AND IMPLEMENTATION OF A SECOND PROTOTYPE OF THE INTELLIGENT ALARM SYSTEM IN ANESTHESIA.
Eindhoven University of Technology, Division of Medical Electrical Engineering, 1990, Report EUT 90-E-233.

[Oos89] van Oostrom, J. H. M.
INTELLIGENT ALARMS IN ANESTHESIA: an implementation.
Eindhoven University of Technology, Division of Medical Electrical Engineering, 1989, Report EUT 89-E-229.

[Oos93] van Oostrom, J. H. M.
A SYSTEM FOR AUTOMATIC ALARM LIMIT SETTING IN ANESTHESIA.

A BREATHING CIRCUIT ALARM SYSTEM BASED ON NEURAL NETWORKS.

[Orr91] Orr, J. A.
AN ANESTHESIA ALARM SYSTEM BASED ON NEURAL NETWORKS.

A BREATHING CIRCUIT ALARM SYSTEM BASED ON NEURAL NETWORKS.

DECISION SUPPORT IN PATIENT MONITORING BY INTER­ACTIVE INFORMATION SYSTEM INCLUDING KNOWLEDGE-BASED COMPONENTS.

ARTIFICIAL INTELLIGENCE RESEARCH IN ANESTHESIA AND INTENSIVE CARE.

LEARNING INTERNAL REPRESENTATION BY ERROR PROP­AGATION.

ALARMS: help or hindrance?
Letter to the editor.

KNOWLEDGE-BASED DECISION SUPPORT IN ANESTHESIA: a case study.
Volume 2.

A KNOWLEDGE-BASED APPROACH TO INTELLIGENT ALARMS IN ANESTHESIA.
IEEE Engineering in Medicine and Biology, Vol. 10(1991), No. 4, p. 38–44.

KAVE: a tool for knowledge acquisition to support artificial ventilation.

[Sha93] Shahsavar, N.
DESIGN, IMPLEMENTATION AND EVALUATION OF A KNOWLEDGE-BASED SYSTEM TO SUPPORT VENTILATOR THERAPY MANAGEMENT.
Linköping Studies in Science and Technology, Dissertations No. 317.

[Sit90] Sittig, D. F. and M. Factor.

A PARALLEL IMPLEMENTATION OF THE BACKWARD ERROR PROPAGATION NEURAL NETWORK TRAINING ALGORITHM: experiments in event identification.

AN ADVISORY SYSTEM FOR ARTIFICIAL VENTILATION OF THE NEWBORN UTILIZING A NEURAL NETWORK.
Medical Informatics, Vol. 18(1993), No. 4, p. 367–376.

NEURAL NETWORKS: what are they and why is everybody so interested in them now?

ALARMS AND ANESTHESIA: challenges in the design of intelligent systems for patient monitoring.
ERGONOMIC AND HUMAN FACTORS AFFECTING ANESTHETIC VIGILANCE AND MONITORING PERFORMANCE IN THE OPERATING ROOM ENVIRONMENT.
INTELLIGENT ALARMS REDUCE ANESTHESIOLOGIST'S RESPONSE TIME TO CRITICAL FAULTS.
Anesthesiology, Vol. 77(1992), No. 6, p. 1074–1079.