Recognition of patterns in multichannel recorded data using artificial neural networks and fuzzy rule based systems; application to daily life motor activities

Publication of this thesis has been supported by the Stichting Anna-Fonds.

CIP- GEGEVENS KONINKLIJKE BIBIOTHEEK, DEN HAAG

Kiani, K
Recognition of patterns in multichannel recorded data using artificial neural networks and fuzzy rule based systems; application to daily life motor activities
Thesis Rotterdam.- With summary in Dutch
ISBN: 90-9011109-3

Cover design: Kourosh Kiani

Printing: Optimadruk, Molenaarsgraaf

Recognition of patterns in multichannel recorded data using artificial neural networks and fuzzy rule based systems; application to daily life motor activities

Patroonherkenning van multi-kanaal geregistreerde gegevens door middel van kunstmatige neurale netwerken en vage regel-gebaseerde systemen; toepassing in dagelijkse motorische activiteiten.

Proefschrift

ter verkrijging van de graad van doctor
aan de Erasmus Universiteit Rotterdam
op gezag van de rector magnificus
Prof.dr P.W.C. Akkermans M.A.
en volgens besluit van het College voor Promoties

De openbare verdediging zal plaatsvinden op
donderdag 13 november 1997 om 11.00 uur

door

Kourosh Kiani
geboren te Pol Sefid, Iran

Promotiecommissie:

Promotoren:	Prof.dr.ir. C.J. Snijders Prof.dr. E.S. Gelsema
Leden:	Prof.dr. H.J. Stam Prof.dr.ir. E. Backer Prof.dr. F.G.A. van der Meché

to Mandana and Pantea

to Ali and Shahin, my parents

to the families of
Mohajer, Karoon, Nateghi, Parviz, Parvin, Houshang, Ciavosh and Khosroo Kiani

Chapter 1

Introduction

1. Introduction

1.1 Introduction

A large number of people with a movement problem forms a relevant social and medical problem in all countries. The rapidly growing number of elderly people, who inevitably experience increasing limitations in their functioning as they grow older, is a cause of major international concern. Only in the European Community, 10% of the population is suffering from more or less severe motor problems [1]. Awareness of disability costs and demographic developments have directed the policy of governments to quality of life problems. More than in the past, research devoted to diseases of the neuro-musculo-skeletal system is supported. This regards diagnosis, surgical and non-surgical treatment, rehabilitation and prevention. In all of these areas biomechanics is essential for the assessment of the mechanical functioning of healthy subjects and patients. Movement analysis is one of the most important parts of biomechanical research.

Since the end of the 19th century there have been attempts to assess movement in an objective and quantitative manner (Muybridge, 1887; Marey, 1894; Braune & Fischer, 1895). During the past 20 years, regular technological developments like microelectronics and fast computational tools have made this goal easier to achieve. Nowadays, in the field of Biomechanical Engineering more and more sophisticated systems for movement analysis(MA) have been developed.

Significant results have been obtained, in several fields such as Rehabilitation, Ergonomics, Sport, Biomechanics and orthopedics. However, in rehabilitation, MA has received limited clinical acceptance, at least in Europe [2].

In 1989, the European Community approved a project on Computer Aided Movement Analysis in a Rehabilitation Context (CAMARC). In general terms, the purpose of the project was to render procedures and instruments for MA useful for patients and clinical doctors through suitable refinements of both instrumentation and software [3].

In other terms, the overall objective of the CAMARC project was the transfer of the ever-improving bioengineering methodology and techniques for MA to the clinical environment[3].

An important cause of the gap between the laboratory and the clinic could be the fact that stance and movement analysis procedures are generally aimed at the understanding of mechanisms at a rather basic level, whereas many clinical questions require an overall assessment of motor behavior in terms of skills instead of functions [4].

In the rehabilitation WHO uses the following classification:

- the level of impairment;
- the level of disability;

- the level of handicap.

In terms of the WHO (1980), it can be stated that in the field of rehabilitation medicine the development of reliable procedures for the assessment of disabilities is more important than the development of methods directed at the detailed analysis of impairment [4][5].

At this moment, the majority of movement analytical applications is focused at the level of impairment, which is especially relevant for orthopedic and surgical procedures.

Rehabilitation medicine, however, is primarily focused at the level of disability. It is remarkable that adequate instruments on the level of disability are relatively scarce.[6]

In rehabilitation medicine there is a growing interest in the investigation of daily life motor activities, as a suitable way to study motor disabilities and as an aid to clinical decision-making.

To facilitate such investigations, new and advanced procedures and instrumentation for monitoring and documenting of daily life motor activities have been developed. At present, a large diversity exists in the employed methods and technology.

The system for Ambulatory Monitoring of Motor Activities (AMMA) is one of the developments that enables clinicians and research workers to obtain quantitative information about daily life motor activities. This information can be used for assessing the functional level of a patient.

1.2 Methods of Monitoring

In general, there are three possibilities to monitor and document daily life motor activities and other clinical parameters relevant for the clinician:

- Questionnaire-based monitoring;
- Laboratory-based monitoring;
- Ambulatory-based monitoring.

Questionnaire-based monitoring [7][8][9][10][11][12][13] gives a good indication of the activities of patients through their self-reports. The questionnaires can be distinguished in accordance to the following main areas:

- Those which examine mainly daily life motor activities. Some examples are:

    Katz ADL scale;

    Barthel Index;

    Amputee activity.

- Those which are multidimensional assessments

    These are designed to contain the physical, social, cognitive and emotional factors which determine an individual's level of functioning. Some examples are:

Multilevel Assessment Instrument;

Sickness Impact Profile;

Comprehensive Assessment and Referral Evaluation.

- Those which examine only daily life motor activities

There is no such questionnaire until now

The following paragraphs give a short description of two of the above mentioned questionnaires.

Amputee Activity

The Amputee Activity questionnaire is an instrument designed to derive a numerical Activity Score from the limb amputee's answers. In this questionnaire, the total time of some activities such as sitting, walking, standing, climbing stairs, and use of a wheelchair play an important role.

Sickness Impact Profile

The SIP is intended as a health status measure to be used in health surveys and in patient progress monitoring. The SIP questionnaire consists of 136 items grouped into 12 categories. Each category represents a different aspect of daily functioning. In the context of the present research it should be emphasized that the level of daily life motor activities has an important influence on the total SIP score.

For a complete description of these two and other questionnaires we refer to the literature. In Appendix A, a part of these two questionnaires is shown.

An important point to be stressed is that there are many serious questions which have to be investigated further. Questions such as: Does this Questionnaire-based monitoring effectively mirror changes in a patient's status? To what extent are the answers objective about parameters such as duration and frequency of daily life motor activities? Are they sufficiently sensitive to detect clinically important changes over time? Do they mirror clinically relevant changes in a short period? Do they mirror clinically relevant changes in a long period? Is the scoring procedure reliable? What is the correlation of the individual question score with the overall score?

Laboratory-based monitoring is until now the most frequently used possibility to monitor and document the relevant parameters in MA. For a laboratory or clinical situation it is relevant to develop a set of elaborate methods and protocols and to exploit the potentialities of the most sophisticated existing instruments for measurement and analysis (3D video recording, moving through magnetism, etc.). In this approach, the patients are attached to a stationary recording instrument through cables. Recording is on-line and the number of simultaneously measured parameters can be high. Although laboratory-based monitoring provides highly accurate and valuable clinical information about daily life motor activities, it has to be considered that in the laboratory environmental circumstances are artificial and are not similar to those in the natural environment. The discordance between laboratory and natural environment initiated the discussion about the validity and objectivity of laboratory based monitoring.

Ambulatory-based monitoring seems to be an appropriate approach for monitoring and documenting daily life motor activities in natural environmental circumstances. "Various parameters from patients are measured in such a manner that subjects are not hooked-up by wire to stationary recording instruments and accordingly their movements are almost unrestricted during these measurements"[14].

The Ambulatory Monitoring of Motor Activities (AMMA) system is one of the Ambulatory-based monitoring systems that enables clinicians and research workers to obtain quantitative information about the motor activities of the human body.

The development of the AMMA system was started in the framework of the CAMARC-II project which is still in progress in the Biomedical Physics and Technology Department (BNT) of the Erasmus University in Rotterdam. The developing of the instrumentation was initially oriented to posture analysis in the context of occupational medicine.[15]

One of the main aims of the AMMA system is to provide means for the automation of the recognition of daily life motor activity classes such as walking, standing, sitting, lying on the back and automation of the computation of other relevant clinical parameters from long term recorded data with high efficiency at reasonable speed, in order to avoid time consuming human intervention.

1.2.1 Ambulatory Monitoring of Motor Activities (AMMA) system

Figure 1 shows the basic configuration and highlights the issues involved in the AMMA system, which is composed of five functional blocks:

- an instrumental block;
- a downloading block;
- a preprocessing block;
- a classification block;
- a visualization block;
- a clinical block.

The instrumental block is the core of the AMMA system. It contains the necessary hardware for the collection of data from moving subjects. Information about posture and movement of the human body and the environmental effects on it, is obtained by the use of sensors which are attached to the body. The sensor responses which represent a given physiological or environmental parameter are sampled and directly stored on a recording device (RAMCORDER) carried by the subject.

Two accelerometers (movement sensors) are placed on the middle of both thighs and collect the acceleration of the thighs perpendicular to the femur. Two other sensors are placed on the sternum; one is to sense acceleration of the trunk in the sagittal direction, the other in the lateral direction.

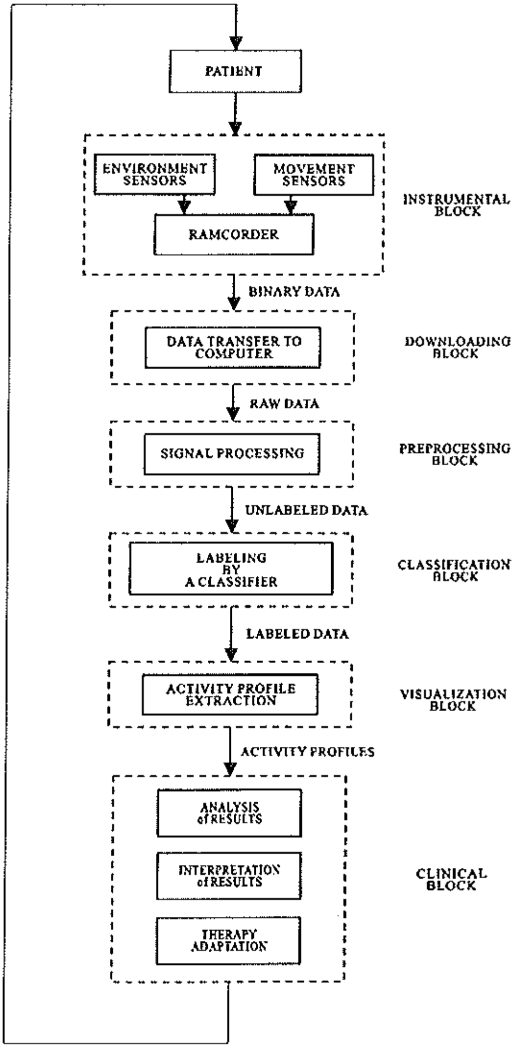


Figure 1: the structure of the AMMA system

The environmental sensors (such as light and sound sensors) are placed somewhere on the subject's clothing[16][17]. Figure 2 shows a patient who is equipped with the ambulatory hardware.

The downloading block transfers the recorded data from the RAMCORDER to a computer, using the custom made software package RAMTALK[18]. RAMTALK also offers some types of file conversion (e.g., from binary to ASCII) to create a new data file format compatible with the next block.

The data preprocessing block transforms raw data into preprocessed unlabeled data and contains the necessary software tools (such as the Codas package[19]). This block can perform some operations on the signals like filtering, offset correction and smoothing operations. Figure 3 shows the output of the four movement sensors after signal processing (the output of the environmental sensors is not included here).

The classification block transforms unlabeled data to labeled data and contains an artificial neural network or fuzzy system as a classifier. This block will be discussed in the following chapter. Figure 4 shows the unlabeled data of Figure 3, but now with labels which indicate daily life motor activities.

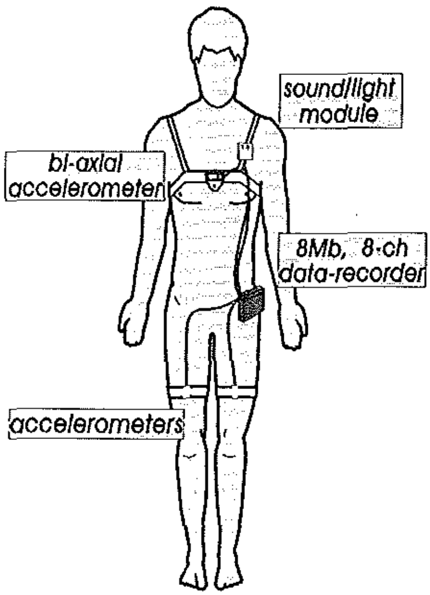


Figure 2: patient instrumentation

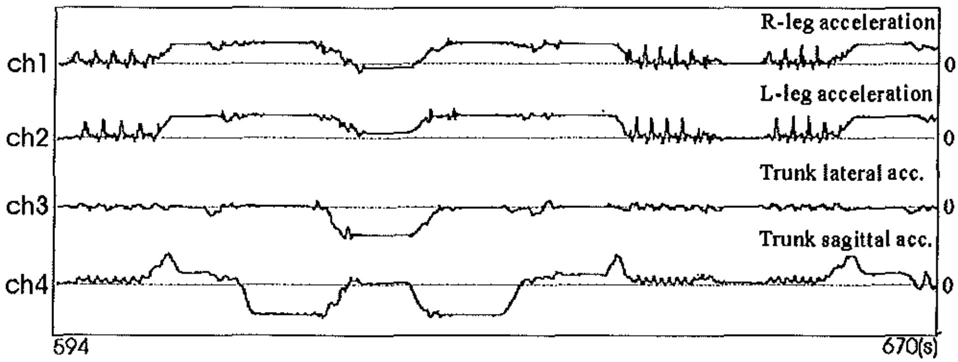


Figure 3: Representative signals for some activities measured with four accelerometers (unlabeled)

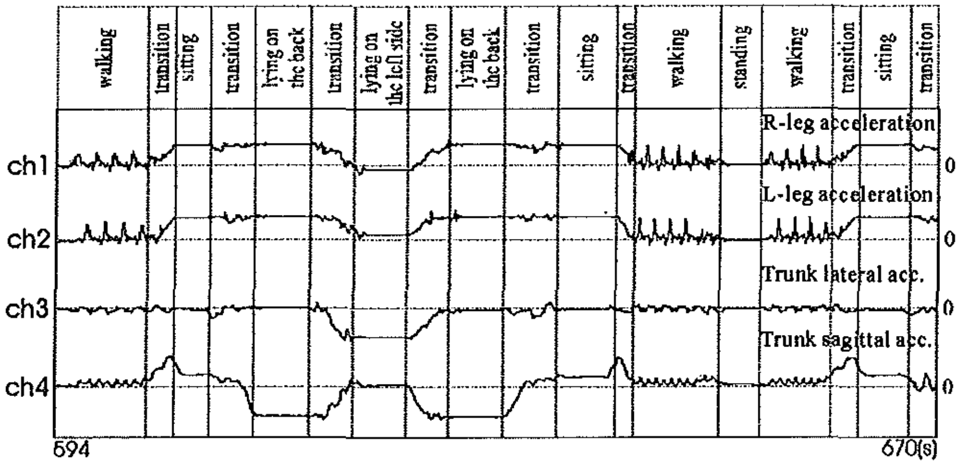


Figure 4: Representative signals for some activities measured with four accelerometers (labeled).

The data visualization block transforms numerical data to graphical output for better interpretation in the subsequent block. The way in which the output is presented, is fundamental for its optimal evaluation by the clinicians. In section 1.2.2, some suggestions are presented for the numerical and graphical representation of the information extracted from the output file of the classification block, which is manipulated in the data visualization block.

The clinical block performs the interpretational, evaluational and analytical operations on the graphical and numerical results, and it may suggest some therapeutical strategies. It is the clinician's concern to improve this block in future.

1.2.2 Activity Profiles

Figure 5 shows an 'Activity Profile'. Here the occurrence of a certain activity over a certain recording time is drawn. In this example, the activities have been classified into six classes. A class can be divided into subclasses, when more details of an activity are of interest. During the recording session, the subject performed a sequence of activities as is clearly demonstrated in

Figure 5. The signal waveforms that could not be recognized by the classification block are indicated as a separate class "unlabeled". All transition activities, e.g., from standing to sitting, from sitting to lying, etc., are shown as the class "transition".

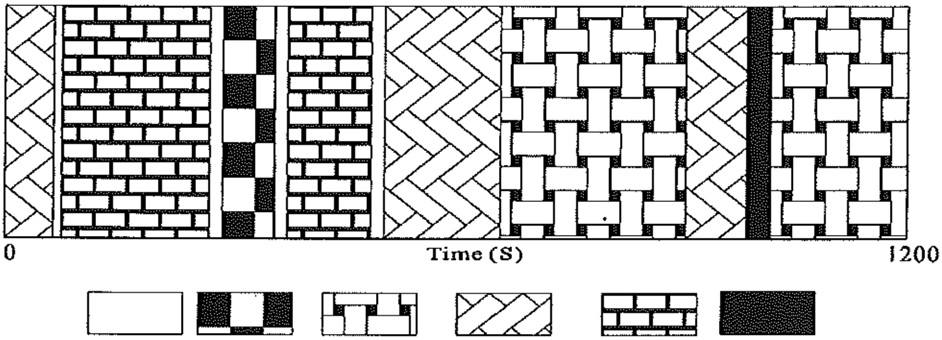


Figure 5: Graphical presentation of an activity profile: sequence and duration of daily life motor activity classes over a period of 20 minutes

An overview of the distribution in time of daily-life activity classes of two amputees during a long term recording is presented in the pie graph in Figure 6. This shows the duration of each activity as a percentage to the total recording time (± 10 hours). In this figure the classes 'transition' and 'unclassified' are put together.

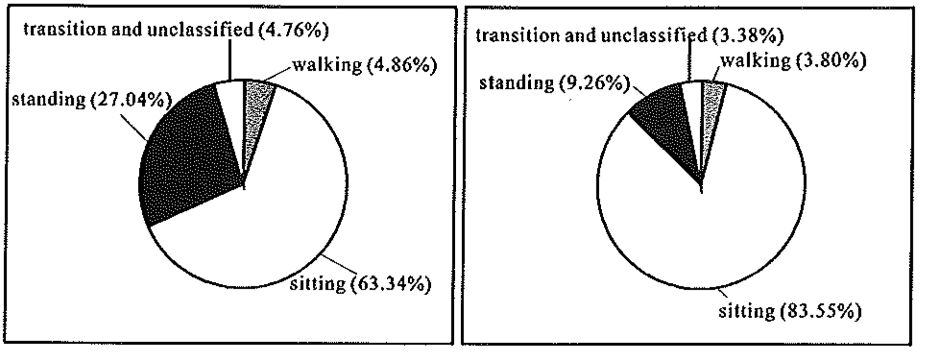


Figure 6: overview of an activity profile: distribution of activities as a percentage of long term recording time for two subjects.

Figure 7 shows two histograms in which the horizontal axis displays the duration of the activity 'walking' divided in category-intervals of 10 seconds, and the vertical axis displays the frequency of each category-interval as recorded during the total recording time.

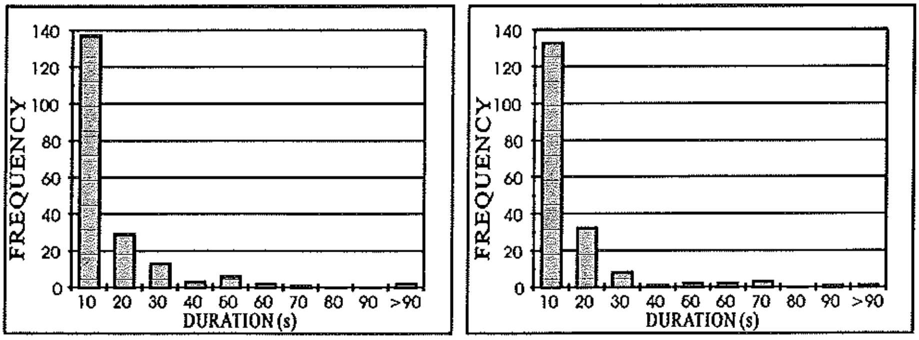


Figure 7: histogram of an activity profile for two subjects: the activity walking is divided in category-intervals (walking blocks) of 10 seconds. In these cases the frequency distribution of the duration of the walking blocks shows a prevalence of short walking periods.

Figure 8 shows a 3-D bar graph of the mean footstep time, as a function of walking block interval time and monitoring time.

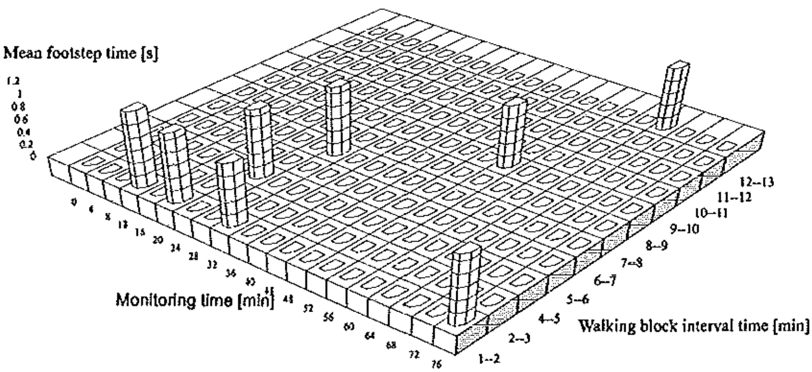


Figure 8: 3-D bar graph of an activity profile: the mean footstep time as a function of monitoring time and walking block interval time.

BLOCK	BTIME(min)	LENGTH(min)	AVG (s)	MODE (s)	STDS (s)	LARGEST (s)	SMALLEST (s)
1	12	1.70	1.17	1.18	0.10	1.43	0.65
2	18	1.30	1.13	1.12	0.11	1.96	1.00
3	22	4.33	1.10	1.06	0.07	1.53	0.87
4	27	6.90	1.07	1.06	0.04	1.28	0.96
5	34	1.85	1.04	1.06	0.05	1.18	0.65
6	49	8.51	1.04	1.06	0.06	1.25	0.65
7	58	12.38	1.04	1.06	0.08	2.03	0.65
8	72	1.92	1.12	1.12	0.08	1.43	0.65

Table 1: some computed parameters for walking block interval time

Table 1 shows additional computed parameters for each walking block.

Where:

BTIME represents the begin time of a walking block;

LENGTH represents the duration of a walking block;

AVG represents the average of all footstep time values in a walking block;

MODE is the footstep time that appears most frequently in a walking block;

STDS represents the standard deviation of all footstep time values in a walking block;

LARGEST represents the largest footstep time value in a walking block;

SMALLEST represents the smallest footstep time value in a walking block.

In Table 2 an example is given of a numerical representation of activities. The statistical data refer to the time necessary for a patient to transit from standing to sitting and from sitting to lying on the back. It is clear that in a similar way transition times for other 'transitions' can be calculated.

	Minimum (s)	Maximum (s)	Average (s)
Transition time standing to sitting	1.63	3.75	2.77
Transition time sitting to lying on back	3.41	5.09	4.63

Table 2: Some statistics about transition time.

1.3 Relevant motor activities and related clinical parameters

It is up to the clinical user of the AMMA system to indicate what information is of relevance. Péruchon [20] postulated that according to the clinicians' point of view and need, the following activities and parameters can correctly reflect the functional profile of a patient.

1.3.1 Relevant motor activities

Figure 9 shows some proposed dynamic and static motor activities and their subclasses which have been suggested in a questionnaire by many clinicians [20][21]. At present, our system is able to detect all daily-life motor activities which are shown in gray in Figure 9.

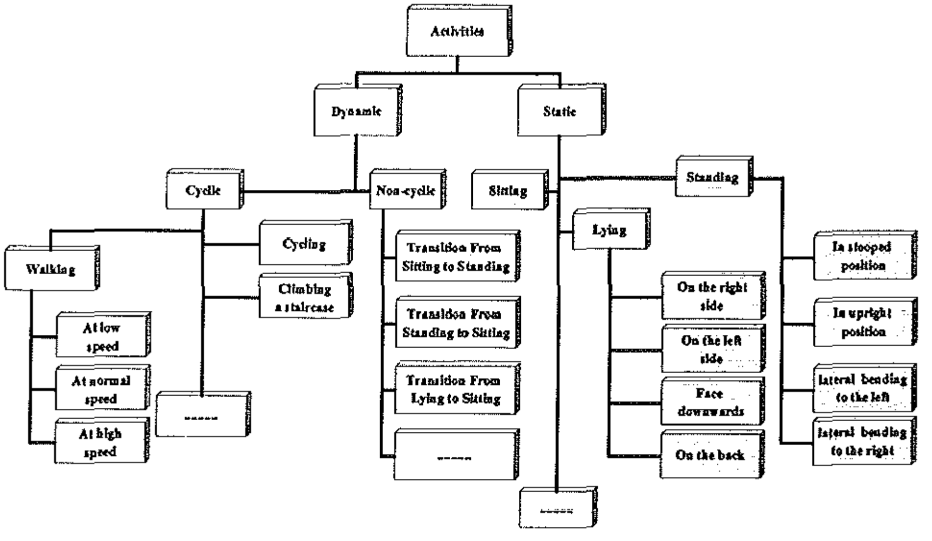


Figure 9: Main relevant motor activities

Some additional clinically representative activities have been proposed such as: to squat, to run, to turn about, etc.

1.3.2 Relevant related parameters

The following parameters are suggested in a questionnaire by many clinicians[20][21]:

total duration in lying position;

total duration in sitting posture;

total daily walking time;

total duration in standing position;

number of times of each activity;

mean duration of each activity;

max, min and average time of transition activities;

speed of walking.

Completion of this list will require further investigations. Of course, it must be noticed that the nature and relevance of motor activities and related parameters depend on the type of the considered pathology[20][21].

1.4 Aim of study

The aim of this study is to investigate and to analyze the abilities of Artificial Neural Networks (ANNs) and Fuzzy Rule Based Systems for the automated recognition of daily life motor activity classes and the computation of relevant clinical parameters.

One of the main aims of the AMMA system is to provide a means for automating the recognition of activity classes in long term recorded data with high efficiency and at reasonable speed, in order to avoid time consuming human intervention.

Some attempts were made to solve the automatic recognition problem by means of a simple signal processing approach, like peak detection, threshold setting, correlation, smoothing, etc. However, all these techniques operated with low efficiency. This is comprehensible, because each class of daily life motor activities shows extreme inter-and intra individual variation, as was discussed in [22]. It is demonstrated that such an approach is not suitable for the recognition of patterns in a noisy environment.

We have chosen the artificial neural network and the fuzzy rule based classifier as an alternative because these have been successfully applied in other pattern recognition problems.

1.5 Overview

The contents of this work are as follows.

Chapter 2 briefly reviews various pattern recognition techniques and addresses a new method for generating new features, which are critical for solving the automated pattern recognition problem for the AMMA system.

Chapter 3 begins with an introduction to neural networks. A short general introduction on the subject is given first, followed by somewhat more detailed descriptions of a number of specific networks. Also, our experiment with neural networks as a pattern recognition system will be considered.

In chapter 4, the basic concepts of fuzzy set theory, fuzzy logic and fuzzy systems are shortly summarized. Also our experiment with a fuzzy rule based system as a pattern recognition system will be considered.

Chapter 5 presents an overview of the different approaches towards constructing Neuro-fuzzy decision systems and their application to the multichannel recorded data.

In chapter 6, summary and future direction are presented.

References

[1] J. Pauwels and T. Mulder, "A preliminary task set-up for disability-oriented assessment ", Deliverable N. 10, CAMARC-II (A2002)/AIM/DG XIII, 22-37, 30 June 1993.

[2] T. Leo and S. Fioretti, "Assessment of clinical protocols ", Human Movement Science, Nr. 10, 515-525, North-Holland, 1991.

[3] V. Macellari, R. Bedini, T. Leo, "Telemetry for movement monitoring in a rehabilitation context: State of the art and future prospect ", Biotelemetry XI, 356-359, 4 Sep 1990.

[4] A.C.H. Geurts, T. Mulder, R.A.J. Rijken, B. Nienhuis, "From the analysis of movement to the analysis of skills ", Journal of rehabilitation sciences 4, Nr. 1, 9-12, 1991.

[5] J. Pauwels, T. Mulder , " The concept of disability ", Deliverable N. 26, CAMARC-II (A2002)/AIM, 2-11, 30 September 1993.

[6] Bussmann JBJ., Stam HJ, " The value of Ambulatory monitoring in rehabilitation research." ISBN: 90-74443-04-4, in Dutch, 1992.

[7] Katz S., Ford A.B., Moskowitz R. W., Jackson B.A., Jaffe M.W. Studies of illness in the aged. The index of ADL: a standardized measure of biological and psychosocial function. JAMA 1963; 185: 914-19

[8] Lawton M.P., Brody E.M. Assessment of older people: Self-maintaining and instrumental activities of daily living. Gerontologist 1969; 9:179-86

[9] Iverson I.A., Silberberg N.E., Stever R.C., Schoenning H.A. The revised Kenny Self-Care Evaluation: a numerical measure of independence in activities of daily living. Minneapolis, Minnesota: Sister Kenny Institute, 1973.

[10] H. J. B Day., "The assessment and description of amputee activity", Prosthetics and Orthotics International, 1981, 5, 23-28

[11] Lawton M.P., Moss M., Fulcomer M., Kleban M.H. A research and service oriented Multilevel Assessment Instrument. J Gerontol 1982; 37: 91-99.

[12] Bergner M., Bobbitt R.A., Carter W.B., Gilson B.S. The Sickness Impact Profile: development and final revision of a health status measure. Med Care 1981; 19:787-805.

[13] Gurland B., Kruriansky J., Sharpe L., Simon R., Stiller P., Birkett P. The Comprehensive Assessment and Referral Evaluation (CARE) -- rationale, development and reliability. Int j Ageing Hum Dev 1977; 8:9-42.

[14] W. H. Groeneveld, E. Péruchon, "Parameter for Ambulatory Monitoring of Motor Activity ", Deliverable N. 10, CAMARC-II (A2002)/AIM/DG XIII, 81-102, 30 June 1993.

[15] Snijders CJ., Riel MP., Nordin M. "Continuous measurements of spine movements in normal working conditions over periods of 8 hours or more." Ergonomics, 639-653, No 30, 1987.

[16] W.H. Groeneveld, "A prototype system for ambulatory monitoring of motor activity" Deliverable N.14,CAMARC-II(A2002)/AIM/DG XIII, 25 July 1994

[17] W.H. Groeneveld, K.J. Waterlander, A.V.L. De Moel, H.J. Konijnendijk and C.J. Snijders. "Instrumentation for ambulatory monitoring of patient movement ", Biotelemetry XII August 31 September 5, 1992, Ancona, Italy

[18] Waterlander KJ., Groeneveld WH., Konijnendijk HJ., "A prototype system for ambulatory monitoring of motor activities" Deliverable N.14,CAMARC-II (A2002)/AIM, 13 September 1994.

[19] CODAS, Software package for data acquisition, visualization and analysis. DTATAQ Instruments, Akron, Ohio, USA

[20] Péruchon E., Pélissier J. " Clinical relevance of ambulatory monitoring of motor activities" Deliverable N.26, CAMARC -II(A2002)/AIM/DG XIII, part B 1, 30 September 1994.

[21] Bussmann JBJ., Veltink PH., Koelma F., Van Lammel RC., Stam HJ. "Ambulatory Monitoring of Mobility-Related Activities: the Initial Phase of the Development of an Activity Monitor" European Journal of Physical Medicine and Rehabilitation. 1-7, vol 5, No 1, March, 1995.

[22] W.H. Groeneveld., K. Kiani., "Classification of AMMA data ". Deliverable N. 26, CAMARC-II (A2002)/AIM,33-46, 13 September 1993.

Chapter 2

Pattern Recognition

2. Pattern Recognition

In the following sections, we provide an introduction to many of the key concepts in pattern recognition and to various techniques for solving pattern recognition problems which are relevant to this thesis.

2.1 Introduction

The term pattern recognition encompasses a wide range of information processing techniques of great practical significance, from computer vision tasks, speech recognition, fingerprint identification, and character recognition, to fault detection in machinery and medical diagnosis. Although such tasks can often be solved without much conscious effort by humans, their solution using computers has, in many cases, proved to be immensely difficult.

Pattern recognition can be defined as a process of identifying structure in data, often by comparison to known structure; the structure may be developed through methods of clustering. Basically, clustering seeks the structure in data, whereas classification attempts to assign new data to one of the classes defined in the classification process. The components of a pattern recognition system are illustrated in Figure 10.

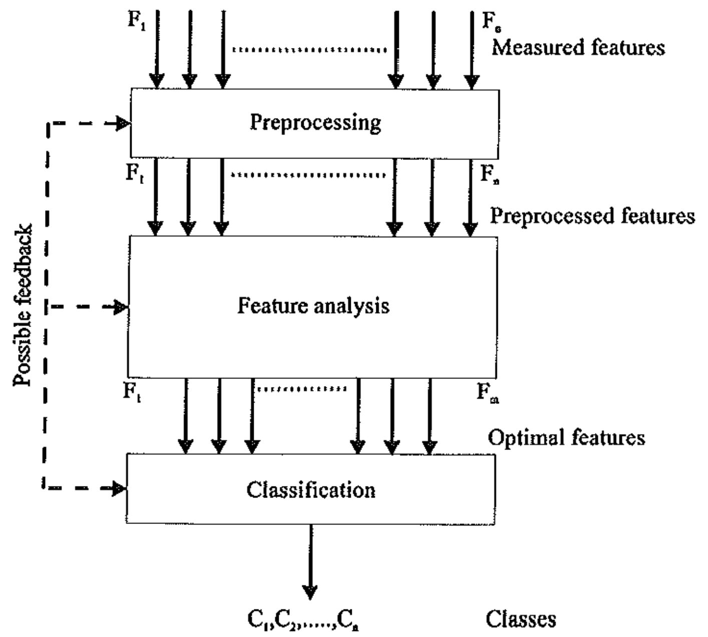


Figure 10: Pattern recognition system in design mode.

The purpose of this pattern recognition system is to assign an observation as represented by a fixed number of measured features to one of C possible pattern classes. Presumably, different input observations should be assigned to the same class if they have similar features and to different classes if they have dissimilar features. A set of features is called a pattern or feature vector and is described by a vector $F = \{ F_1, F_2, \dots F_n \}$. The n individual features, $F_1, \dots F_n$, are assumed to be representative and sufficient to recognize the underlying pattern.

The preprocessing part often has a significant effect on the performances of the total system.

2.2 Feature analysis

Feature analysis refers to methods for conditioning the raw data (measured features) so that the information that is most relevant for classification and recognition is enhanced and represented by a minimal number of features. Feature analysis consists of two component:

- feature selection
- feature extraction

Feature selection refers to choosing the subset of m features with the highest discriminating ability from the n original features (m<n), as illustrated in the example in Figure 11. In general, the selection of the features is more important than the choice of a specific classifier. When features with no discriminating ability are used, no classifier will give acceptable results. On the other hand, when features with a very high discriminating ability are used, all classifiers will give comparable results. In almost all problems, one does not know beforehand how many features must be used in the classifier. One of the simplest techniques for dimensionality reduction is to select a subset of the features, and discard the remainder. This approach can be useful if there are features which carry little useful information for the solution of the problem, or if there are strong correlation between pairs of features so that the same information is repeated in several variables. There are many procedures for feature selection[1][2][3].

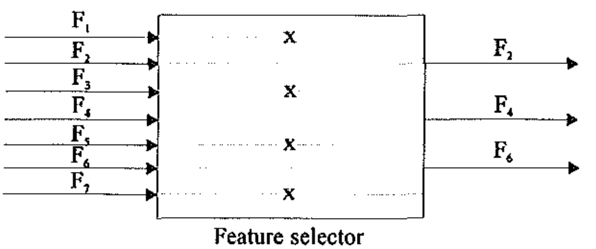


Figure 11: Dimensionality reduction by feature selection.

Feature extraction (FE) refers to the process of transforming the original n-dimensional feature space into an m-dimensional space in some manner that preserves or enhances the information available in the original n-dimensional space, as illustrated in the example in Figure 12. It is accomplished mathematically by means of either some linear or nonlinear combination of the original features.

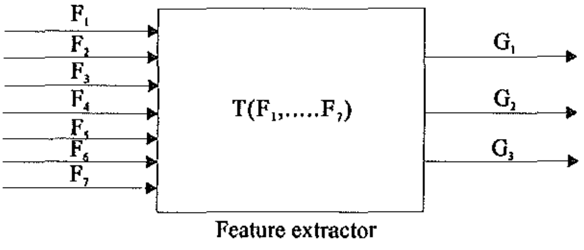


Figure 12: Dimensionality reduction by feature extraction.

2.3 Partitioning of the feature space

Partitioning the feature space into c regions that are associated with classes, is usually in the domain of classifier design. Usually, the feature space is R^n, and classifiers partition R^n into c disjoint regions. A region is called a decision region, and its boundary is a decision boundary.

Sometimes the decision regions can be linearly separated (i.e., by straight lines in R^2, by planes in R^3, by hyperplanes in higher dimensional spaces). If decision regions are intertwined in a way that makes it impossible to separate them with linear decision boundaries, the problem is called non-linearly separable. Figure 13 illustrates two examples of partitioning in a two- dimensional feature space.

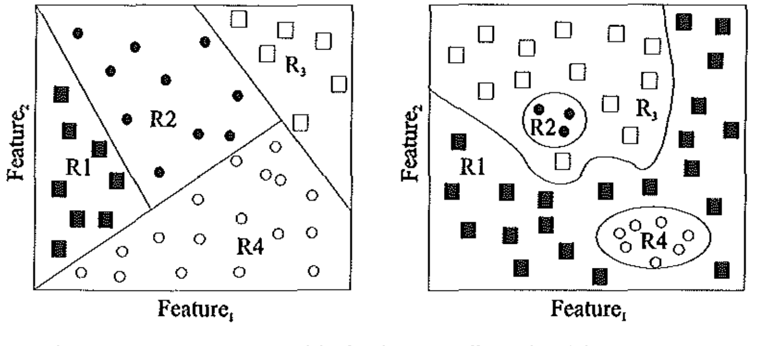


Figure 13: Examples of partitioning in a two-dimensional feature space:

left: partitioning by linear decision boundaries.
right: a general partitioning of the feature space.

A classifier generally consists of a set of discriminant functions $g_i(F)$, i=1,....c where F is the input feature vector, and c is the number of classes [4]. A discriminant function $g_i(F)$ is defined for each decision region i. The classifier assigns an observation with input feature vector F to region m associated with class m if the relationship

$g_m(F) > g_i(F)$ for all $i \neq m$

holds. That is, the classifier computes c discriminant functions and selects the class corresponding to the largest discriminant.

The decision boundaries can be easily found from this description; e.g., the boundary between the regions i and j is given by the equation:

$g_i(F) - g_j(F)=0$

This representation of a classifier is given as a block-diagram in Figure 14.

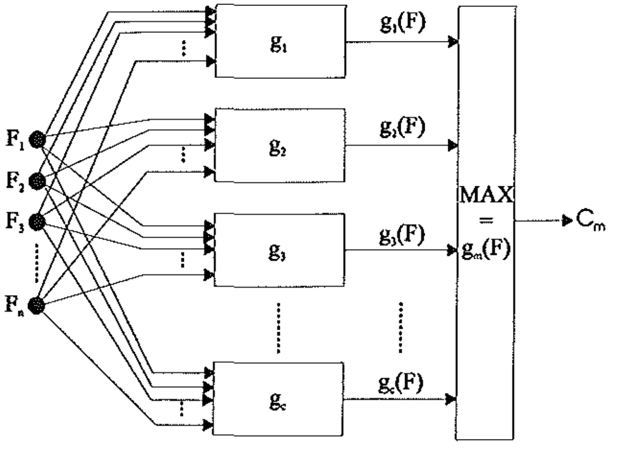


Figure 14: A pattern classifier.

The discriminant functions may be linear or nonlinear combinations of the input feature values. An example of a linear discriminant function is given by:

$g_i(F)=W_i^T.F + w_{i0} = w_{i1}.F_1+w_{i2}.F_2+\ldots w_{in}.F_n+w_{i0}$

where vector W_i is a so-called weight vector associated with class i (the superscript T stands for transposition of a (column) vector) and vector F is the pattern vector (or feature vector). The weights (free parameters) of each linear discriminant function are calculated to minimize the rate of misclassification [4][5].

It should be pointed out that the choice of dicriminant functions is not unique; every $g_i(F)$ can be replaced by $f(g_i(F))$ without influencing the decision, if f is a monotonically increasing function.

There are various techniques for solving pattern recognition problems. Some of them which are relevant to this thesis will be discussed in the following sections.

2.4 Statistical pattern recognition techniques

Statistical pattern recognition system assigns a randomly observed pattern vector $F \in R^n$ to a decision region $R_i \subset R^n$ associated with class i using the distribution of the pattern vectors in R^n as established in a learning phase. This is usually accomplished by defining suitable discriminant functions that divide the n-dimensional feature space into regions that correspond to the different classes. Suppose $P(c_i)$ is the (known, estimated, or assumed) a priori probability of occurrence of patterns from class c_i, and $P(F \mid c_i)$, the class-conditional density function of the random variable F given that the corresponding sample belongs to class c_i (this density function is estimated during learning or training). From Bayes theorem we have:

$$P(c_i \mid F) = P(F \mid c_i).P(c_i) / P(F)$$

where the posterior probability $P(c_i \mid F)$ gives the probability of the pattern belonging to class c_i once we have observed the feature F, and P(F) is the density function for F irrespective of class, and is given by

$$P(F) = \sum_{i=1}^{c} P(F|c_i)P(c_i)$$

The division by P(F) ensures that the posterior probabilities sum to unity :

$$\sum_{i=1}^{c} P(F|c_i) = 1$$

Therefore, we can easily compute $P(c_i \mid F)$ once we have the estimates for $P(F \mid c_i)$ and $P(c_i)$. These are then used to assign the sample corresponding to the measurement vector F to the class c_m if

$$P(c_m \mid F) > P(c_i \mid F) \quad \text{for all } i \neq m$$

We can then represent the discriminant functions as

$$g_i = P(c_i \mid F)$$

The remaining problem is how to determine the class-conditional probability density functions (PDFs) $P(F \mid c_i)$. This problem can be solved using training sets. A training set contains observations, either with or without class labels. Using a training set for the estimation of the PDFs is referred to as learning or training.

There are two broad types of statistical learning methods: parametric and non-parametric.

Parametric methods assume a specific functional form of the class-conditional density functions $P(F \mid c_i)$ for each pattern class c_i. Such functions contain a number of adjustable parameters which are optimized by fitting the model to the training set. The simplest, and most widely used, parametric model is the normal or Gaussian distribution, which has a number of convenient analytical and statistical properties. The drawback of such an approach is that the assumed parametric form for the density function may not be a good representation of the true density. It should be emphasized that accurate modeling of PDFs from finite data sets in spaces of high dimensionality is, in general, extremely difficult. However, parametric models allow the density function to be evaluated very rapidly for new values of the input vector.

Non-parametric methods do not assume a particular functional form for the density functions, but allow the form of the density to be determined entirely by the training set. A number of non-parametric techniques are available [2][4][6].

A popular example of a non-parametric method is the nearest neighbors method in which a sample is assigned to the class of its nearest neighbor(s) (in terms of a suitable measure of distance) in the training set. Non-parametric methods suffer from the fact that the number of parameters in the model grows with the number of training data points.

The field of statistical pattern recognition techniques is very large, and it is not possible to give a complete description of all the aspects and issues mentioned in this section. The implementation of these technique as a neural network will be discussed in chapter 3

2.5 Symbolic Artificial Intelligence Techniques

Symbolic processing, as the name suggests, deals with information in terms of symbols. The symbols, representing pieces of information, are usually manipulated using IF-THEN rules. IF-THEN rules are expressions of the form IF input is A THEN output is B, where A and B are symbols characterized by appropriate characteristic functions (membership functions).

An example that describes a simple fact is:

If *pressure* is **high**, then *volume* is **small.**

Where *pressure* and *volume* are variables, **high** and **small** are symbols that are characterized by membership functions. Examples are shown in Figure 15, for high and small symbols.

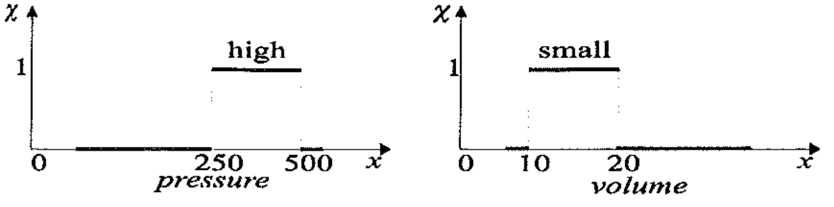


Figure 15: Membership functions for (left) high pressure and (right) small volume.

Expert systems are well-known systems for manipulating symbolic representations [7][8][9]. Rather than discussing all different kinds of expert systems we will focus our attention mainly on an expert system which uses the rules formalism for representation of the knowledge. There are several formalisms available and used for representing the different types of knowledge: rules, frames, semantic network, inheritance, predicate calculus. Basically, an expert system is composed of four blocks, as shown in Figure 16:

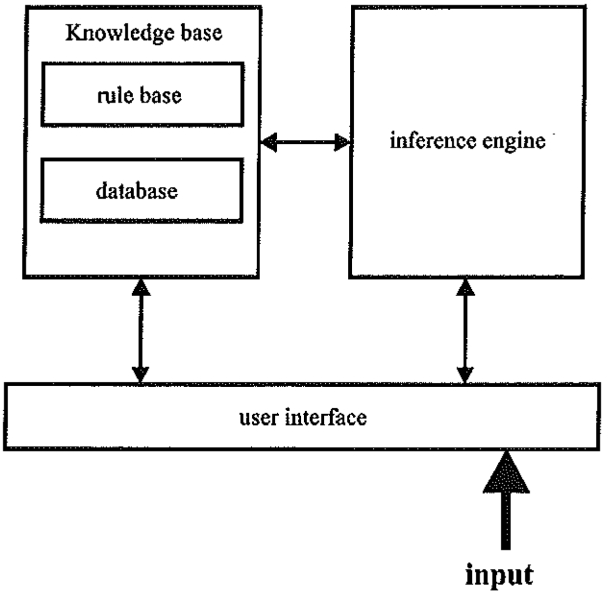


Figure 16: Structure of an expert system.

- a rule base containing a number of IF-THEN rules;
- a database which defines the membership functions of the symbols in the rules;
- an inference engine which performs the inference operations on the rules;
- a user interface which facilitates data-entry and data-retrieval. Furthermore, it is used to give explanations to the user.

Usually, the rule base and the database are jointly referred to as the knowledge base. The domain-specific knowledge is stored in a knowledge base. Since the knowledge base is separated from the other blocks of system, it is possible to use the inference mechanism and the user interface of such developed expert systems as a tool to build other expert systems. One needs to replace only the knowledge base by another one. This reduces the development time of an expert system considerably. A significant characteristic of expert systems is that they operate in a transparent fashion, i.e. the path to their conclusion can be traced. New knowledge in the form of new rules may be added, hence allowing for incremental development, refinement and tuning of the knowledge base. This expert system is similar to a fuzzy rule-based system which will be in introduced section 2.7 and fully discussed in chapter 4.

2.6 Neural network techniques

Artificial neural networks are processing structures that are inspired on the architecture and functioning of the human nervous system. They have become popular in many research fields because of their ability to solve complex problems for which no analytical solutions appear to be feasible. Therefore, there is an exponentially increasing growth of ANN research. As a result of this research, ANN's have been used in a broad range of applications which include pattern recognition [10], classification [11], approximation [12], optimization [13], prediction [14], control [15], speech recognition [16], modeling [17], systems identification [18], etc. They are able to learn by example; given a set of examples and their class, a network can learn to emulate the required decisions. They are able to generalize: an ANN can not only learn to recognize patterns which are used to train it, but it can also recognize similar unseen patterns.

Basically, an artificial neural network consists of many, highly interconnected simple processing elements, called "neurons", "units" or "nodes". The neurons are usually arranged in series of layers, bounded by input and output layers with generally a number of hidden layers in between. Each of these neurons receives input from other neurons in the previous layer and applies an activation function to its summed inputs in order to obtain an output which is propagated to other neurons in the next layer. Information from the input layer is propagated through the network to the output layer. The input and output neurons are the means of the network to communicate with the outside world. Input layer neurons are merely a mechanism for distributing the input signal to the subsequent hidden layers. Figure 17 shows a simple fully connected feed-forward neural network with one hidden layer.

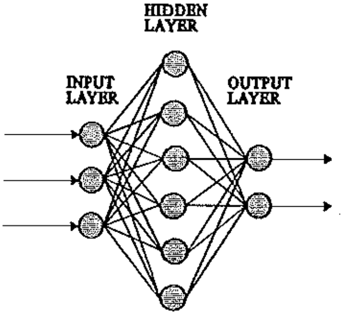


Figure 17: A feedforward neural network with one hidden layer.

There is a number of different types of artificial neural network architectures which are usually characterized by:

- the topology of the network; the topology of a neural network is the organization of neurons into layers and the connections between them;
- the characteristics of the neurons;
- the type of learning scheme.

A detailed description of these items is presented in chapter 3. The following paragraphs focus on how artificial neural networks can be used in pattern recognition tasks.

As explained in section 2.3, classifiers are designed to determine the decision regions in feature space. Artificial neural networks as classifiers can be categorized according to the manner in which they estimate the decision regions in feature space [19] as follows:

- Kernel classifier;
- Hyperplane classifier;
- Probabilistic classifier;
- Exemplar classifier.

In Kernel classifiers each neuron has a kernel function centered around a location in the feature space. The idea is to cover the feature space with kernel functions. A neuron gives a maximum response to input vectors near the center of its kernel function. We say that each neuron has its own receptive field in the feature space. Decisions regarding the classification are made by using a weighted summation of the outputs of neurons. One of the special basis functions that are commonly used is a Gaussian kernel function. Figure 18 shows schematically how a Kernel classifier estimates the decision regions in feature space. An example of an artificial neural network that belongs to this category is the radial basis function networks (RBFN) [20][21].

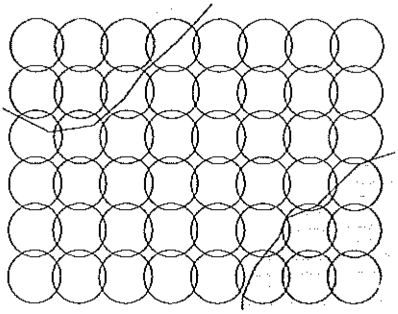


Figure 18: The estimated decision regions by Kernel classifiers.

Hyperplane classifiers partition the feature space by hyperplanes which are generated by computation of a sum of weighted inputs for every neuron and applying a non-linear transfer function to this sum. Figure 19 shows an example of a partitioning which is produced by a hyperplane classifier. An example of this type of classifier is the back propagation neural network (BPN).

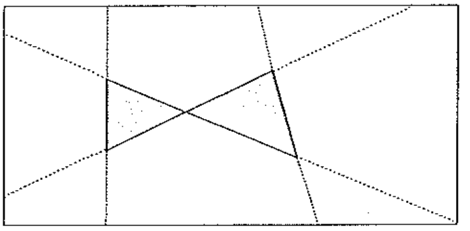


Figure 19: A schematic illustration of a decision boundary produced by a hyperplane classifier.

A probabilistic classifier is simply an example of the parametric, non-parametric and mixture approach to density estimation implemented as a neural network. The activation function of each neuron is replaced by a statistically derived one. Figure 20 shows a schematic illustration of a decision boundary which produced by a probabilistic neural network classifier. An example of this type of classifier is the Gaussian mixture classifier.

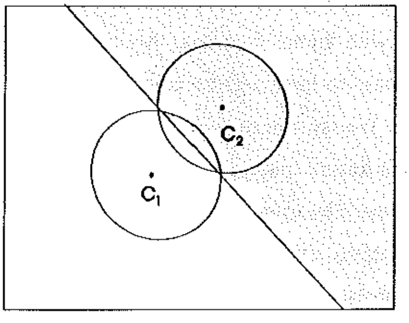


Figure 20: A schematic illustration of a decision boundary produced by a probabilistic neural network classifier.

Exemplar classifiers check the distance (Euclidean or using some other metric) separating an unknown input pattern from each member of the training set. The unknown pattern will then be assigned to the class to which the closest training set member belongs. Some examples of this type of classifier are Adaptive Resonance Theory (ART) [22], Kohonen networks [23], Learning Vector Quantization LVQ

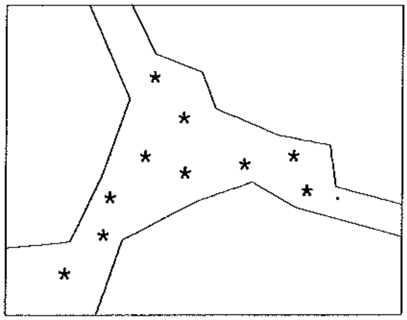


Figure 21: A schematic illustration of the decision regions in feature space produced by an exemplar classifier.

Despite this obvious diversity in the discussed categories of artificial neural network classifiers, all of them perform the same task: partitioning of the feature space into decision regions. They differ only in the way they perform this task.

It is difficult to determine which type has the greatest probability of success. The choice of a classifier from the numerous available artificial neural network classifiers is most often based on its success in previous applications, practical experience with a wide variety of them used in various applications and the complexity of the problem. Engineering judgment and creativity are nearly always required. In this way it may be possible to provide acceptable solutions to problems that were not yet solved otherwise.

2.7 Fuzzy Rule-Based techniques

A fuzzy rule-based system is an extension of the crisp rule-based system discussed in section 2.3. This section will introduce the way in which fuzzy rule-based system can be used in pattern recognition. A detailed discussion of this subject will be presented in chapter 4.

Unlike conventional (crisp) approaches of pattern classification, fuzzy classification assumes that the boundary between two neighboring classes is an overlapping area within which a pattern (an object) has partial membership in each of the two classes. This viewpoint not only reflects the reality of many applications in which categories have fuzzy boundaries, but also provides a simple representation of the potentially complex partitioning of the feature space. The classifier is described by fuzzy IF-THEN rules. Typical fuzzy classification rules for a 2-dimensional feature space are like:

R_1: *IF* x_1 is small *AND* x_2 is very large *THEN* $x = (x_1, x_2)$ belongs to class c_1

R_2: *IF* x_1 is large *AND* x_2 is small *THEN* $x = (x_1, x_2)$ belongs to class c_2

R_3: *IF* x_1 is **small** *OR* x_2 is **small** *THEN* $x = (x_1, x_2)$ belongs to class c_3

⋮

⋮

R_i: *IF* x_1 is **very small** *AND* x_2 is **very large** *THEN* $x = (x_1, x_2)$ belongs to class c_k

where R_i is the i.th classification rule, c_k indicates an output class, x_1 and x_2 are the features of a pattern (or object), very small, small, large and very large are linguistic terms characterized by appropriate membership functions and *AND* and *OR* are fuzzy logical operations. Figure 22 shows a three-class classification problem. Three membership functions are associated with each feature, so the feature space is partitioned into 9 fuzzy regions (subspaces), each of which is governed by a fuzzy IF-THEN rule. The antecedent part of a rule defines a fuzzy region, while the consequent part specifies the output within this fuzzy region.

If one tries to classify all the given patterns by fuzzy rules based on a simple fuzzy grid, a fine fuzzy partition and 9 rules (3x3=9) are required. However, it is easy to see that the patterns may be correctly classified by the five fuzzy IF-THEN rules as follows:

R_1: *IF* x_1 is NOT **Low** *AND* x_2 is **Low** *THEN* $x = (x_1, x_2)$ belongs to class C_1

R_2: *IF* x_1 is **Low** *AND* x_2 is NOT **Low** *THEN* $x = (x_1, x_2)$ belongs to class C_1

R_3: *IF* x_1 is **Low** *AND* x_2 is **Low** *THEN* $x = (x_1, x_2)$ belongs to class C_2

R_4: *IF* x_1 is NOT **Low** *AND* x_2 is **High** *THEN* $x = (x_1, x_2)$ belongs to class C_2

R_5: *IF* x_1 is NOT **Low** *AND* x_2 is **Medium** *THEN* $x = (x_1, x_2)$ belongs to class C_3

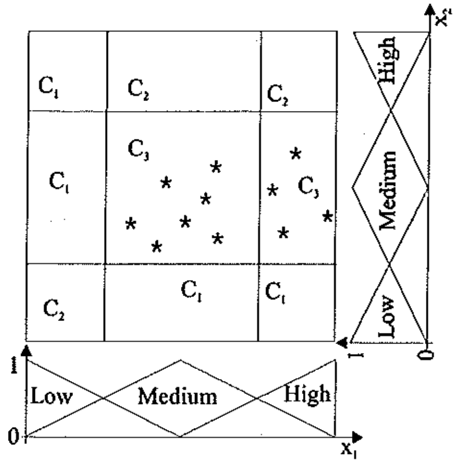


Figure 22: Fuzzy partition with 9 fuzzy regions.

2.8 Neural-Fuzzy techniques

Neural-Fuzzy hybrid systems combine the advantages of fuzzy systems and neural networks.

While neural networks are good at recognizing patterns, they are not good at explaining how they reach their decisions. It is difficult to explain the knowledge learnt by a neural network. Neural networks have a black box nature.

Fuzzy logic can encode expert knowledge directly using rules with linguistic terms, and are good at explaining their decisions but they can't automatically extract the rules which they use to make decisions. Also, it usually takes much time to design and tune the membership functions which quantitatively define the linguistic terms.

These limitations have led to the creation of neural-fuzzy networks where neural networks and fuzzy logic techniques are combined in a manner that overcomes the limitations of the individual techniques. Neural-Fuzzy networks try to remove the mentioned limitations by combining the learning capabilities of neural networks together with the interpretability properties of fuzzy systems. Neural-Fuzzy networks partition the feature space better than fuzzy rule-based systems, because the membership functions in Neural-fuzzy networks are tunable. A full description of Neural-Fuzzy Networks will be given in chapter 5.

2.9 AMMA signal processing with the previous described techniques

It has to be considered that ambulatory monitoring of daily life motor activities is conceptually new. Therefore, it is difficult to find an automated system for the recognition of AMMA-signals in literature. The only similar monitoring system which can be found in literature [24], uses several forms of signal-processing such as high pass and low pass filtering, rectifying procedures, and frequency analysis to automate the recognition of daily life motor activities. This system has still the following shortcomings:

- the system is patient dependent;
- the system splits activities only in two categories, dynamic and static, and is not able to classify subclasses of dynamic activities.

The novelty of the AMMA-signals, the lack of references in literature, the inability of conventional signal processing techniques and the success of neural network and fuzzy logic techniques in other application of pattern recognition led us to apply these techniques to AMMA-signals.

In chapter 3, the applications of two types of neural networks to AMMA-signals in order to recognize and classify daily life motor activities will be discussed. One of the applied neural networks is an implementation of the Bayes decision strategy which is called Probabilistic Neural Network (PNN). The other one is a BackPropagation Neural Network (BPN). In chapter 4, we will apply a fuzzy rule-based technique to AMMA-signals to overcome the shortcomings of the neural network techniques. Before that, in the following sections, we address some other related subject.

2.10 Sensors

The silicon accelerometers register the orientation and movement of body segments. Each daily life motor activity is reflected in specific orientations and movements of some body segments. Because the output of an accelerometer is a mixture of two components, a gravitational component (DC-component,) and a component of the change in velocity (dynamic- component), both static and dynamic motor activities are reflected. The component which reflects static activity is more constant in time and the component which reflects a dynamic activity changes rapidly with time. Figure 23 shows the effect of a rotation of the sensor on its output due to the earth's gravity field.

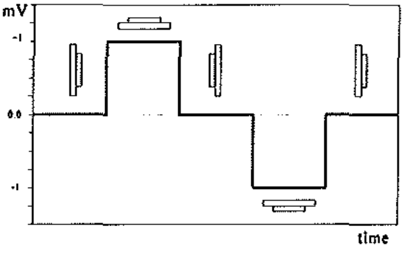


Figure 23: The effect of a rotation of the sensor on its output (mV) due to the earth's gravity field.

2.11 Definition of the patterns of activities

A first requirement for the classification of daily life motor activities is that all activities are defined. It was investigated how to acquire recognizable and accurate information on a subject's basic daily life motor activities with a minimal number of sensors. Figure 24 shows the output signals of four accelerometers which are attached to the subject's body.

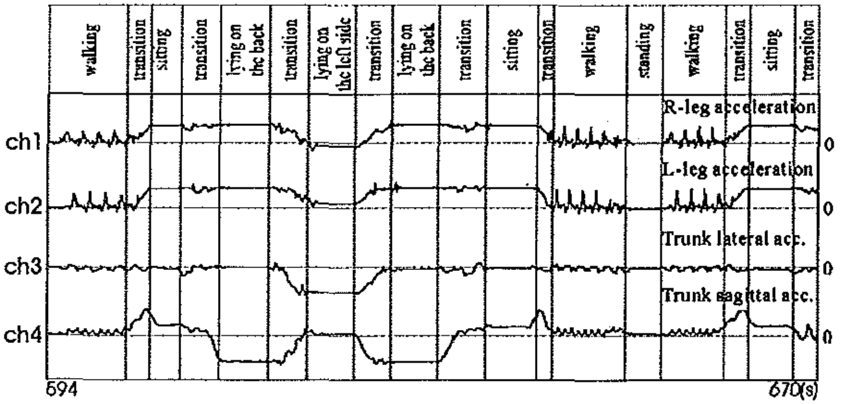


Figure 24: Representative signals for some activities measured with four accelerometers

Two accelerometers (represented by ch_1 and ch_2) are placed on the middle of both thighs and collect the acceleration of the thighs perpendicular to the femur. Two other sensors represented by ch_3 and ch_4 are placed on the sternum, to sense acceleration of the trunk in the lateral and in the sagittal direction, respectively. It was investigated that these four sensors give sufficient information about basic activities. From these sensor's outputs, a trained eye can recognize each specific posture and movement of the subject as reflected in specific waveforms of the signal in each channel. Based on knowledge of the sensor characteristics and visual interpretation of the recorded reference data which is obtained according to a protocol, each activity class can be defined. In the following paragraphs basic postures and movements will be defined.

Walking

Walking is a dynamic and cyclic activity which has the highest variability of all basic classes that are recognized. Walking can vary from fast running to shuffling and its pattern (waveform) shows extremely large inter- and intra- individual variation. Figure 25 shows the signals of the two accelerometers on the right and left upper leg for a single subject during walking.

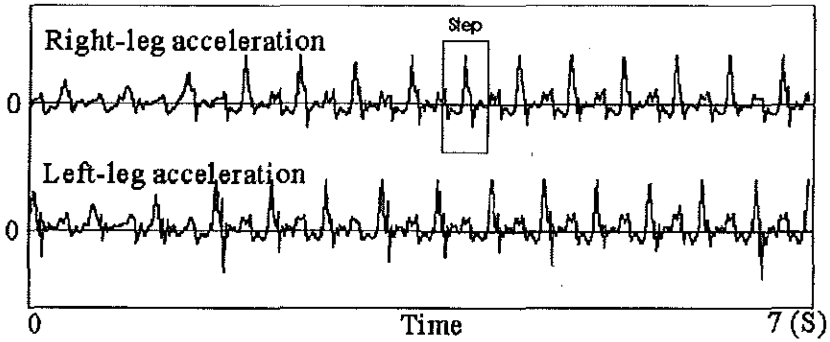


Figure 25: Typical example of signals from the accelerometers on right and left upper leg, while walking.

As shown in Figure 25, the fluctuations of the waveform of a step are very large. Each step pattern differs in amplitude and slope from other steps. The subject of this example was healthy and walked normally. It is obvious that walking at various speeds will produce even more varying patterns. Figure 26 demonstrates the intra-subject variability of acceleration of the upper leg during 20 step-cycles.

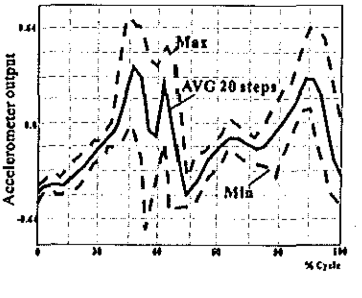


Figure 26: Intra-subject variability of sensor output of the upper leg.

This and the inter-subject variability of the pattern of steps render automatic recognition by conventional signal processing techniques extra difficult. Walking as a class is divided into two subclasses, i.e., walking slowly and at normal speed and a second sub-class that covers fast walking. A trained eye can recognize walking activity by looking at the first and second channels (left and right upper leg accelerometer outputs). However, using only one of them is sufficient to detect walking activity. Figure 27 shows a part of recorded data of the two accelerometers on the right and left upper leg when subject walks normally. In this figure, the cursor positioned on the peak of a step pattern in channel 1, its amplitude value is 1.2700 Volt.

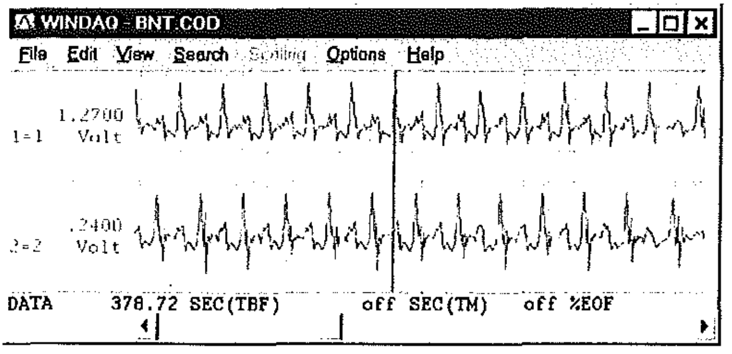


Figure 27: A part of recorded data during normal walking activity.

Figure 28 shows a part of recorded data of the two accelerometers on the right and left upper leg when subject walks slowly. In this figure, the cursor positioned on the peak of a step pattern in channel 1, its amplitude value is 0.3400 Volt.

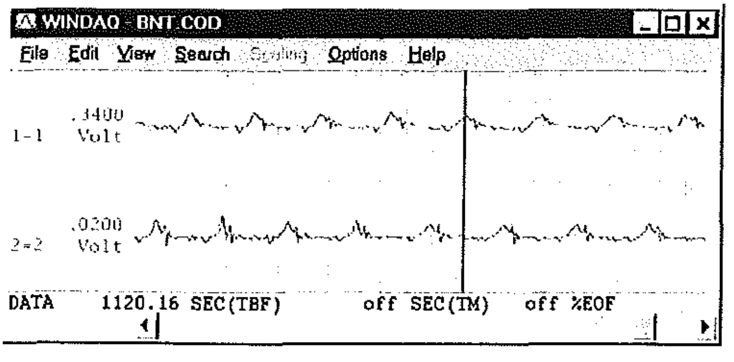


Figure 28: A part of recorded data during slow walking activity.

Figure 28 shows again a part of recorded data of the two accelerometers on the right and left upper leg when subject walks normally. But in this figure, the cursor positioned somewhere between two step patterns in channel 1, its amplitude value is 0.4700 Volt.

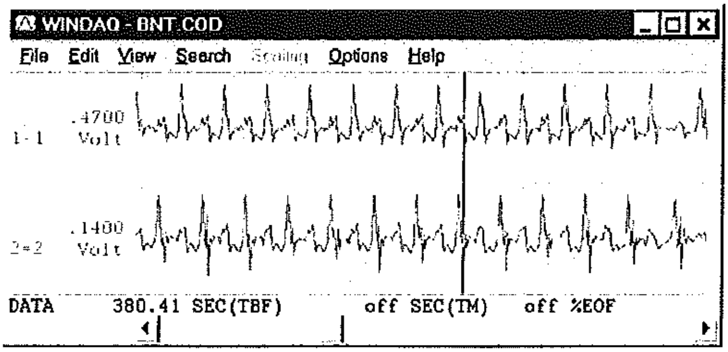


Figure 29: A part of recorded data during normal walking activity.

From Figure 27-29 one can see that conventional peak detection technique fails to find a magic threshold for recognition of step pattern.

Sitting

Sitting is a static and non-cyclic activity and the trained eye can recognize it easily by using both the left and right upper leg accelerometers and the sagittal trunk accelerometer. The trunk lateral movements are not used for the definition of the activity sitting. Figure 30 illustrates the onset and end of activity class sitting. The use of the sagittal trunk accelerometer (fourth channel) is necessary in order to avoid misclassification. If one only uses the first and second channels (right-leg, left-leg) to classify the sitting activity, there is the chance that an incorrect label is placed, since the activity lying on the back can result in similar waveforms in the first and second channels. The patterns of sitting posture are also variable but compared to steps, this pattern is much more stable and thus, easier to recognize.

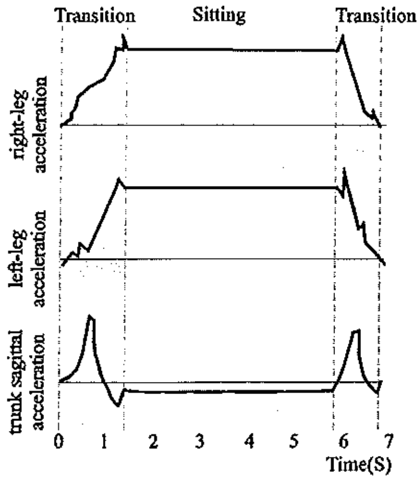


Figure 30: Specific waveforms for activity class sitting.

Standing upright

Standing is also a static activity and its recognition from the four sensor signals is based on the combination of the following criteria:

- one of the legs is in, or close to, the vertical position; thus the variations in that signal are relatively small.

- none of the activities: lying on the left or on the right side, are recognized simultaneously with the first condition.

The second criterion is applied in order to avoid misclassification. Using only the first and second channels to classify the standing activity may result in misclassification. The activities lying on the left and right side can result in a similar waveform in the first and second channels. The activity bending down can be differentiated from standing upright, by considering the trunk accelerations (the third and fourth channel).

Lying

Lying is a static activity and is divided into the following three sub-classes:

- lying on the back;
- lying on the right side;
- lying on the left side;

Information on the subclass is obtained from the trunk accelerometers sensing the movements or postures in the frontal plane. The class 'lying on the back' is recognized if a single or both leg sensor(s) as well as the trunk are in the horizontal position at the same time.

The class 'lying on the left or right side' is determined by combining the information from 'lying on the back' (it is assumed that the class 'lying on the left or right side' will occur after the class of 'lying on the back') and is determined by the output of the lateral sensor on the sternum.

2.12 Modified pattern recognition system and feature generation

In this section, we present a modified version of the pattern recognition system discussed in section 2.1 and shown in Figure 10, and describe an added component (feature generation) to this modified system. Although a trained eye can recognize all activities by using the four channels (four continuous features), and a patient dependent ANN based classifier is also able to classify the desired activities automatically with the same number of sensor outputs, our research showed that using only the four features (the four accelerometer outputs) cannot eliminate patient dependency of the system. To overcome this shortcoming, we have modified the pattern recognition system described in section 2.1 in such a way that it generates new features. Figure 31 shows this modified pattern recognition system.

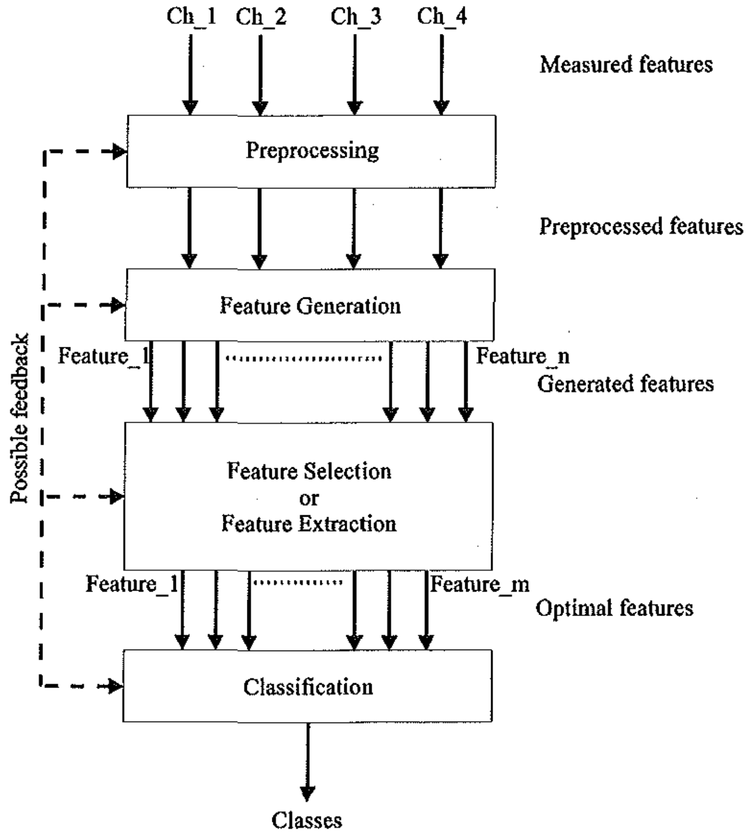


Figure 31: A schematic representation of a modified pattern recognition system.

In chapter 4, we will show that the added feature generation block serves to improve the automated classification of daily life motor activities. In the next section, the feature generation block and its implementation are discussed.

Feature generation and implementation outline

As mentioned before, using only the four continuous features (the outputs of the four accelerometers) is not enough to design a patient independent classifier, for the recognition of activities (different waves in signals), and the detection of onsets and endpoints of the waves. This initiated the search for a new representation (features) generated from the preprocessed measured features. Feature definitions may be constructed 'by hand', based on some understanding of the problem (i.e., the incorporation of prior knowledge), or features may be derived from the preprocessed measured data by automated procedures. Prior knowledge may be incorporated into all parts of the pattern recognition system.

For the generation of new features from the original data, a variety of methods is available. This study focuses on two methods both employing a running window technique. Figure 32 illustrates how two points of a new feature can de derived from the input data. This figure employs a window of width sixteen. The bottom line represents the preprocessed measured data values. The top line forms the new input data values. The core of the operation consists of the following steps:

1. put the window at the beginning of the input data;
2. evaluate the function F: $R^{16} \rightarrow R^{16}$ on the vector selected by the window;
3. evaluate the function G: $R^{16} \rightarrow R$ at function value in step 2;
4. move the window one point to the right;
5. go to step 2.

This is repeated for the entire length of the input data. Two examples of $G(F([x_1, \ldots x_{16}]))$ are:

- Norm (Cumulative_Sum($[x_1, \ldots x_{16}]$)),
- Average(Outer_product ($[x_1, \ldots x_{16}]$)).

Prior knowledge which refers to relevant information (shape of waves, length of waves, sampling rate, etc.) can be of help for choosing the two functions **F** and **G** and the window size.

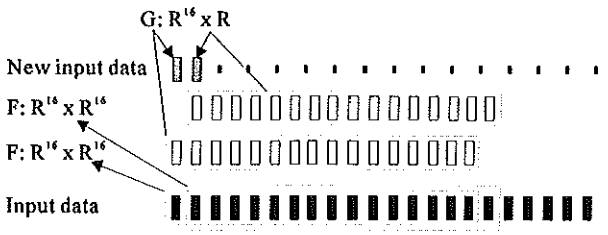


Figure 32: schematic explanation of the first feature generation procedure

Another method, similar to the previous one, but consisting of only four steps is :

1. put the window at the beginning of the input data;
2. evaluate the function G: $R^{16} \rightarrow R$ on the vector selected by the window;
3. move the window one point to the right;
4. go to step 2.

These steps are repeated for the entire length of the input data. Figure 33 illustrates how two points of the new feature are constructed by this procedure.

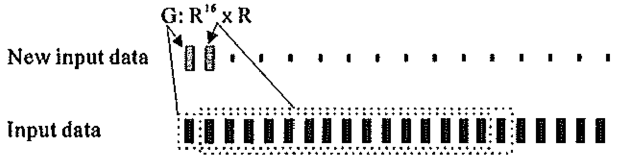


Figure 33: schematic explanation of the second feature generation procedure

This figure employs also a window of width 16. It is important to select an optimal window size. Its size depends on the sampling rate (the sampling rate of AMMA signal was 32 per second) , and it should be noted that it is not necessary to have the same window size for different features.

Two examples of $G([x_1,\dots x_{16}])$ are:

- Norm $([x_1,\dots x_{16}])$;
- Inner_product$([x_1,\dots x_{16}])$.

The average, standard deviation, sine, cosine, Fourier transform, cumulative sum, norm, inner product, outer product, max, min, etc. are typical **F** and **G** functions which were used to generate new features.

References

[1] Hand D. J. "Discrimination and classification", New York: John Wiley, 1981,

[2] Fukunaga K. "Introduction to Statistical Pattern Recognition", San Diego: Academic Press, 1990.

[3] Siedlecki W. and Sklansky. " On automatic feature selection", International Journal of Pattern Recognition and Artificial Intelligence, 197-220, vol 2, No 2, 1988.

[4] Duda. R.O. and Hart. P.E. : Pattern Classification and Scene Analysis. New York: Wiley-Interscience, 1973

[5] Afifi A.A. and Azen S.P. : Statistical Analysis: A Computer Oriented Approach. New York: Academic, 1972.

[6] Devijver P and Kittler J. : Pattern Recognition: A Statistical Approach. Englewood Cliffs, New Jersey: Prentice-Hall, 1982

[7] Buchanan B.G and Schortliffe E. H : Rule-based expert systems. The MYCIN experiments of the Stanford Heuristic Programming Project. Addison-Wesley, 1984.

[8] Hayes-Roth F, Waterman D.A. and Lenat D. : Building expert systems. Addison-Wesley, 1983.

[9] Brachman R.J and Levesque H.J. : Readings in knowledge representation. Morgan Kaufmann Publ. Inc., 1985.

[10] Wong AJ. : Recognition of general patterns using neural networks." Biological Cybernetics, 1988;58:361-372.

[11] Raychaudhuri T, Chieh-hong Yeh J, Hamey LGC. Westcott CT : Baked product classification with the use of a self-organizing map. Proceedings of the sixth Australian conference on Neural Networks;1995 Feb 6-8; Sydney. Sydney: Dept. of Electrical Engineering, University of Sydney, Australia,1995;152-155.

[12] Hornik KM, Stinchcombe M, White H. :Multilayer feedforward networks are universal approximators. Neural Networks 1989;2:359-366.

[13] Favata F, Walker R. :A study of the application of Kohonen-type neural networks to the traveling salesman problem. Biological Cybernetics 1991;64:463-468.

[14] Azoff EM. :Neural network time series forecasting of financial markets. New York: John Wiley & Sons, 1994

[15] Van der Smagt P, Groen FCA., Kröse BJA. :A monocular robot arm can be neurally positioned. Proceedings of International Conference on Intelligent Autonomous Systems (IAS-4);1995,1995;123-130..IOS. Press 1995.

[16] Lippmann R.P.: Review of Neural Network for Speech Recognition. Neural Computation, 1989;1:1-38.

[17] Back AD, Tsoi AC. : FIR and IIR synapses, a new neural network architecture for times series modeling. Neural Computation 1991;3(3):375-385.

[18] Narendra KS, Parthasarathy K. : Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Network 1990;1(1):4-27.

[19] Lippmann R. P. : Pattern classification using neurál networks. IEEE communications Magazine, pp. 47-64, November 1989.

[20] Moody J.E. and Darken C.J. : Learning with localized receptive fields. In Proceedingsof the 1988 Connectionist Models Summer School, eds. Touretzky D., G. Hinton, and Sejnowski, pp. 133-143. San Mateo, CA: Morgan Kaufmann.

[21] Moody J.E. and Darken C.J. : Fast learning networks of locally-tuned processing units. Neural Computation, 1989;1(2):281-294

[22] Grossberg S. : Competitive learning: From interactive to adaptive resonance. Cognitive Science, 1987; 11:23-63.

[23] Kohonen T. : Self organization and associative memory. New York, Springer-Verlag, 1988.

[24] Bussmann JBJ., Veltink PH., Koelma F., Van Lammel RC., Stam HJ. "Ambulatory Monitoring of Mobility-Related Activities: the Initial Phase of the Development of an Activity Monitor" European Journal of Physical Medicine and Rehabilitation. 1-7, vol 5, No 1, March, 1995.

Chapter3

Neural Networks

This chapter is a generalization of the following paper:

Recognition of Daily Life Motor Activity Classes Using an Artificial Neural Network

Kourosh. Kiani*MSc, Chris. J. Snijders*Phd, Edzard. S. Gelsema**Phd

Faculty of Medicine and Allied Health Sciences, Erasmus University Rotterdam

* Department of Biomedical Physics and Technology

** Department of Medical Informatics

Accepted by Archives of Physical Medicine and Rehabilitation 1997

3. Neural Networks

3.1 Introduction

Artificial neural networks have become popular in many research fields because of their ability to solve complex problems for which no analytical solutions appear to exist. Therefore there is an exponentially increasing growth of ANN research. As a result of this research, ANN's have been used in a broad spectrum of applications which include pattern recognition [1][2], classification [3], approximation [4], optimization [5], prediction [6], control [7], speech recognition [8], modeling [9], systems identification [10], etc. Artificial neural networks are processing structures inspired by the architecture and functioning of the human nervous system[11][12], but they are only loosely related to them. ANN's are massively parallel systems that rely on dense arrangements of interconnections and surprisingly simple processors. In a neural network, each processor is linked to many of its neighbors (typically hundreds or thousands) so that there are many more interconnection than processors. The power of the neural network lies in this tremendous number of interconnections. The strongest feature of neural networks is their ability to accept examples and generalize from them; i.e., an ANN can not only learn to recognize patterns which are used to train it, but it can also generalize and recognize similar patterns.

3.1.1 History of Neural Network

The history of neural networks started in 1943 with the publication of a paper by McCulloch and Pitts [13], which introduced a model of a neuron that was capable of performing useful logical and arithmetic functions. In 1949, D. O. Hebb [14], proposed a learning law that became the starting point for artificial neural network training algorithms and inspired many researchers to study neurocomputing. Around 1960 there was a wave of activity centered around the group of Rosenblatt, concentrating on networks called perceptrons. These networks are limited to two layers of processing units with a single layer of adaptive weights between them. That was the beginning of a golden period in neurocomputing research, which was to last until 1969, when many artificial neural network were developed, implemented and applied to a wide variety of problems. Many pioneers expressed a great deal of enthusiasm and hope that such machines could be a basis for artificial intelligence. This enthusiasm soon proved to be an illusion. Perceptrons failed to solve problems superficially similar to those that had been successfully solved. Minsky and Papert [15] in their book Perceptrons, showed that Rosenblatt's perceptron was theoretically incapable of solving many simple problems, including the function performed by a simple exclusive (XOR) operation. Rosenblatt had also studied structures with more layers of adaptive weights and believed that such networks could overcome the limitations of the simple perceptrons. However, there was no learning algorithm known which could adapt the weights. With the negative assessment of perceptrons by Minsky and Papert, the absence of an analytical approach to the neural network and the absence of the learning algorithm had enormous consequences and effectively led to the dampening of continued interest in neural network research. Many of the researchers deserted the field, only a handful of the early pioneers maintained their commitment to neural networks.

In the 1980s, major contributions to the theory and design of neural networks were made on several fronts, which led to a rebirth of interest in neural networks.

Grossberg 1980 [16], established a new principle of self-organization, using his earlier work on competitive learning [17][18][19].

In 1982, Hopfield used the idea of an energy function to formulate a new way of understanding the computation performed by recurrent neural networks with symmetric synaptic connections. His work motivated many researchers to (re)start research on neural networks. Another important development in 1982 was the publication of Kohonen's paper on the self-organizing map [20]. In 1983, Cohen and Grossberg [21] established a general principle for designing a content-addressable memory. In 1985, Ackley, Hinton, and Sejnowski [22] exploited the idea of simulated annealing (simulated annealing is rooted in statistical thermodynamics) in the development of a stochastic learning algorithm that uses some nice properties of the Boltzmann distribution-hence the name Boltzmann learning.

In 1986, the development of the back-propagation algorithm was reported by Rumelhart, Hinton and Williams [23]. In that same year, the two-volume book, " Parallel Distributed Processing" (often referred to as PDP) was published. This latter book has had a major influence on the use of back-propagation learning, which has since emerged as the most popular learning algorithm for the training of multilayer perceptrons. In fact, back-propagation learning was discovered independently in two other places about the same time (Parker, 1985[24], LeCun, 1985). After the discovery of the back-propagation algorithm by Parker, and LeCun, it turned out that the algorithm had been described earlier by Werbos in his Ph.D. thesis in 1974 [25].

In 1988, Broomhead and Lowe [26] described a procedure for the design of layered feedforward networks using radial basis functions (RBF), which provide an alternative to multilayer perceptrons. Many of the important early papers including many of those mentioned here have been collected in

Anderson and Rosenfeld (1988)[27]. Neural networks have certainly come a long way from the early days of McCulloch and Pitts and the development theory, design, and applications will continue.

This chapter contains a brief general introduction to neural networks, followed by a more detailed description of the networks used in this work. The application of the described ANNs, with emphasis on pattern recognition in AMMA-signals, is discussed at the end of this chapter.

3.1.2 Processing elements

The individual computation elements that make up most artificial neural system models are often referred to as neurons or Processing Elements(PEs). Figure 34 shows the general PE model.

Every PE has many inputs, and a single output which can fan out to other PEs in a following layer.

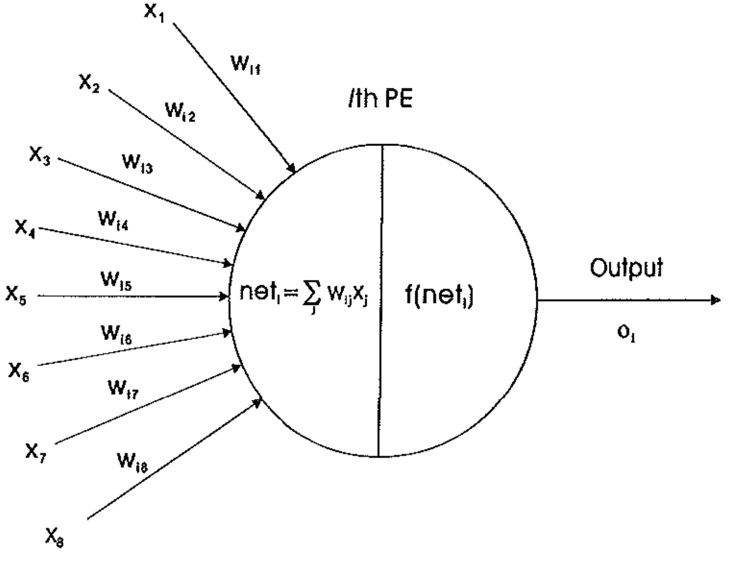


Figure 34: Functional model of an artificial neuron.

Each link between two PEs has a coupling coefficient that assigns a weight to incoming signals.

Each PE determines a net-input value based on all its input connections. The net-input is calculated by summing the input values, multiplied by their corresponding weights. In other words, the net-input to the ith unit can be written as:

$$net_i = \sum_j w_{ij} x_j \qquad 1$$

where the index j runs over all connections to the PE.

Once the net input of the PE is calculated, we can determine the output value by applying an output function (transfer function):

$$o_i = f(\mathrm{net}_i) \qquad 2$$

This results in the following equation for the output of a processing element:

$$o_i = f\left(\sum_j x_j w_{ij}\right) \qquad 3$$

where:

o_i = the output of the *i*th PE.

x_j = the input of the ith PE.

w_{ij} = the weight connection associated with the j.th input.

$f(\,)$ = the threshold function: $f(\mathrm{net}_i) = \begin{cases} 1 & \mathrm{net}_i \geq \theta \\ 0 & \mathrm{net}_i < \theta \end{cases}$

θ = the threshold level.

The basic idea of a weighted summation of inputs to be compared to a treshold to determine the output, is used in most neural networks. However, there are many variations on this basic model. Some networks operate on continuous input signals or use a different output function to calculate the output from the weighted sum.

Examples of such different output functions are the semilinear sigmoid function (i.e., bounded above and below, but differentiable) used in a Back Propagation Network (BPN) and a nonlinear exponential function as used in a Probabilistic Neural Network (PNN). Figure 35 shows a number of different output functions.

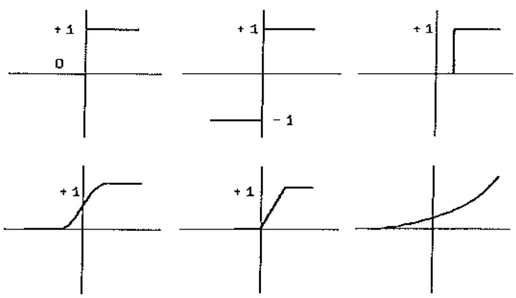


Figure 35: A number of different output functions (activation functions).

3.1.3 Neural Network Topologies

The topology of a neural network is the organization of units into groups and the connections between them, and can be divided in topologies with strictly feed-forward connections and with feedback connections. In a purely feed-forward network, the input simply flows through the connections. As it passes through intermediate PEs, it is transformed until it ultimately reaches its final form at the output PEs. The only time-related factor is that sending PEs must compute their states before the receiving PEs can use them to compute their own states. Once the flow of information reaches the output PEs, processing ends until new input values are fed into the network. Thus, a simple functional relationship exists between inputs and outputs. Two typical examples of a feed-forward network are the Back Propagation network (BPN) and the Probabilistic Neural Network (PNN). These two feed-forward networks topologies are illustrated in Figure 36.

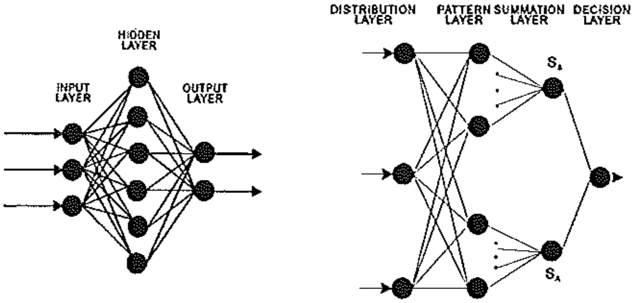


Figure 36: Feedforward topology: (Left) A three-layer BPN, (Right) PNN.

In feedback topologies, the output values of higher level PEs are fed back to lower levels. Figure 37 shows two feedback networks.

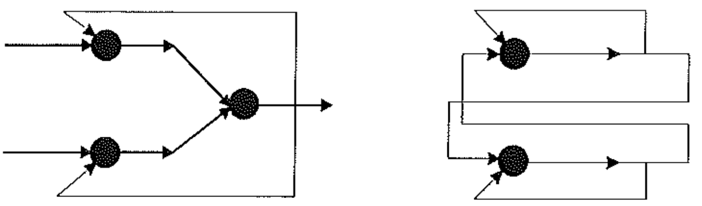


Figure 37: Feedback Network topology.

3.1.4 Learning methods

The learning methods may be categorized as:

-Supervised learning

-Unsupervised learning

-Non learning

3.1.4.1 Supervised learning

In supervised learning, the network is trained on a training set consisting of labeled input vectors. The vector is applied to the input of the network; the label is used as a "target" representing the desired output.

Training is accomplished by adjusting the network weights so as to minimize the difference between the desired and the actual network output. The supervised training process is illustrated schematically in Figure 38.

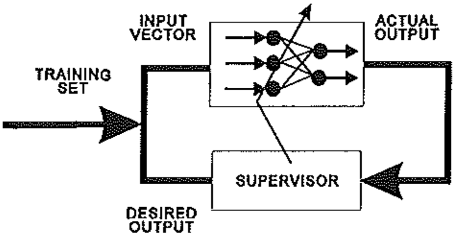


Figure 38: Supervised training process.

3.1.4.2 Unsupervised learning

Unsupervised learning, sometimes called self-organization, requires no labels for the input vectors to train the network. The learning goal is not defined in terms of specific correct examples. During the training process, the network weights are modified so that similar inputs produce similar outputs. The unsupervised training process is illustrated schematically in Figure 39.

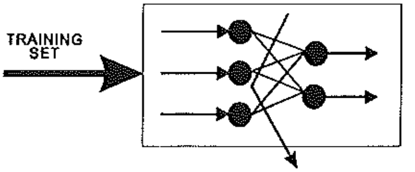


Figure 39: Unsupervised training process.

3.1.4.3 Non learning

Non learning networks simply store the training sets, and perform pattern matching calculations. The non learning training process is illustrated schematically in Figure 40.

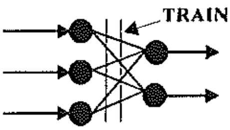


Figure 40: Non learning training Process.

3.1.5 The learning process and neural networks topology

Many of the learning methods are closely connected with a certain (class of) network topology. Below, an overview of some of the different ANNs is given.

-Supervised learning:

i) Feedback Networks:

- a) Mean Field Annealing
- b) Boltzman Machine (BM)
- c) Learning Vector Quantization (LVQ)

ii) Feedforward Networks:

- a) Perceptron
- b) Adaline, Madline
- c) BackPropagation Network (BPN)

-Unsupervised learning:

i) Feedback Networks:

- a) Binary Adaptive Resonance Theory (ART1)
- b) Analog Adaptive Resonance Theory (ART2)
- c) Discrete Hopfield (DH)
- d) Continuous Hopfield (CH)
- e) Discrete Bidirectional Associative Memory (BAM)

ii) Feedforward Networks:

- a) Linear Associative Memory (LAM)
- b) CounterPropagation Network (CPN)

-Non learning:

- a) Probabilistic Neural Network (PNN)
- b) Spatiotemporal Pattern Recognition (SPR)

In our case, in AMMA signal classifications, we know the patterns to be recognized, and we do not want the network to define the classes, so we need a **supervised learning** network, or a **non learning** network. In this thesis we apply the two following artificial neural networks:

- Probabilistic Neural Network(PNN)
- BackPropagation Network (BPN)

to recognize and classify patterns of the activities that were described earlier.

3.2 Probabilistic Neural Network (PNN)

3.2.1 Introduction

The network described here is actually a statistical algorithm proposed several decades ago. It is described in Meisel, [28], and Duda, [29]. Although its theoretical and practical power was known at that time, the state of computer technology precluded its widespread use. Even moderate size problems required memory and CPU speed far beyond what was available at that time. Therefore it fell into disregard until Specht revived it in the form of a neural network [30] which he called a " probabilistic neural network" referring to its roots in probability theory. He showed that by organizing the flow of operations into "layers", and assigning primitive operations to individual "neurons" in each layer, the algorithm can be made to resemble a four-layer feedforward network with exponential activation functions. The following sections present the network in a form closer to its roots.

To understand the basis of the PNN paradigm, it is useful to begin with a discussion of the Bayes decision strategy. It will then be shown that this statistical approach can be mapped into a feedforward neural network structure typified by many neurons that can perform all functions in parallel.

3.2.2 The Bayes strategy for pattern classification

An accepted criterion for decision rules or strategies used to classify patterns is that they do so in such a way that the expected risk is minimized. Such strategies are called "Bayes strategies", and can be applied to problems containing multiple categories.

Consider the two-category situation in which the state of nature θ is known to be either θ_A or θ_B. If it is wanted to decide whether $\theta=\theta_A$ or $\theta=\theta_B$ based on a set of measurements represented by the P-dimensional vector $X^t=[X_1 \cdots X_j \cdots X_p]$, the Bayes decision rule becomes:

$$\begin{aligned} d(X) &= \theta_A \quad \text{if} \quad h_A l_A f_A(X) \geq h_B l_B f_B(X) \\ d(X) &= \theta_B \quad \text{if} \quad h_A l_A f_A(X) < h_B l_B f_B(X) \end{aligned} \qquad 4$$

where $f_A(X)$ and $f_B(X)$ are the probability density functions of the vector X for categories A and B. Variable l_A is the loss function associated with the decision $d(X) = \theta_B$ when $\theta = \theta_A$ and l_B is the loss function associated with the decision $d(X) = \theta_A$ when $\theta = \theta_B$ (the losses associated with correct decisions are taken to be equal to zero); h_A is the a priori probability of occurrence of patterns from category θ_A, and $h_B = 1 - h_A$ is the a priori probability that $\theta = \theta_B$.

The key to using Eq.(4) is the ability to estimate PDFs (Probability Density Functions) based on training patterns. Often a priori probabilities are known or can be estimated; the loss functions require subjective evaluation.

3.2.3 Estimating the PDF using Parzen windows

Bayesian classification requires a PDF for each class. In practice, it is often difficult to determine the PDF with high accuracy. There may be too few training vectors, and the data may be incomplete or it may be partially inaccurate. Some means are required to estimate the PDF from such sparse, real-world data sets. Parzen [31] developed such a technique, commonly called the method of Parzen windows. The following equation expresses the method for finding the needed value for the PDF:

$$f_C(X) = \frac{1}{(2\pi)^{\frac{p}{2}}\sigma^p}\frac{1}{n_c}\sum_{i=1}^{n_c}\exp\left(-\frac{(X-Y_{ci})^t(X-Y_{ci})}{2\sigma^2}\right) \qquad 5$$

where

$f_c(X)$	=	the value of the PDF of class C at point X
n_c	=	number of training vectors in class C
p	=	number of components in the training vector
X	=	the point in feature space at which the PDF is to be evaluated
Y_{ci}	=	*i*th training vector from class C
σ	=	smoothing variable
t	=	vector transpose
i	=	training vector number

While this formula may appear complicated, the idea is simple; $f_c(X)$ is simply the sum of multivariate Gaussian distributions centered at each of the training samples. However, the sum is not limited to being Gaussian. It can in fact approximate any smooth density function. Figure 41 illustrates the effect of different values of the smoothing parameter σ on $f_c(X)$ for the case in which the training patterns are one-dimensional patterns, and Figure 42 illustrates the two-dimensional case. The density is plotted from Eq. (5) for three values of σ with the same training set in each case. A small value of σ causes the estimated density function to have distinct modes corresponding to the locations of the training examples. A larger value of σ, as indicated in Fig. 41b,c and 42b,c produces a larger degree of spread of the contributions of individual cases. Here, values of X close to the training examples are estimated to have about the same probability of occurrence as the given examples. A very large value of σ would cause the estimated density to be Gausian, regardless of the true underlying distribution. As it will be seen, selection of proper values of σ is needed for adequate generalization and classification. With Eq.(5) the Bayes decision rule for the two-category situation in Eq.(4) becomes

$$d(X) = \theta_A$$

$$\text{if} \qquad (6)$$

$$\sum_{i=1}^{n_A} \exp\left[-\frac{(X - Y_{Ai})^t (X - Y_{Ai})}{2\sigma^2}\right] \geq S.\sum_{i=1}^{n_B} \exp\left[-\frac{(X - Y_{Bi})^t (X - Y_{Bi})}{2\sigma^2}\right]$$

and

$$d(X) = \theta_B$$

$$\text{if}$$

$$\sum_{i=1}^{n_A} \exp\left[-\frac{(X - Y_{Ai})^t (X - Y_{Ai})}{2\sigma^2}\right] < S.\sum_{i=1}^{n_B} \exp\left[-\frac{(X - Y_{Bi})^t (X - Y_{Bi})}{2\sigma^2}\right]$$

where

$$S = \frac{h_B l_B}{h_A l_A} \cdot \frac{n_A}{n_B}$$

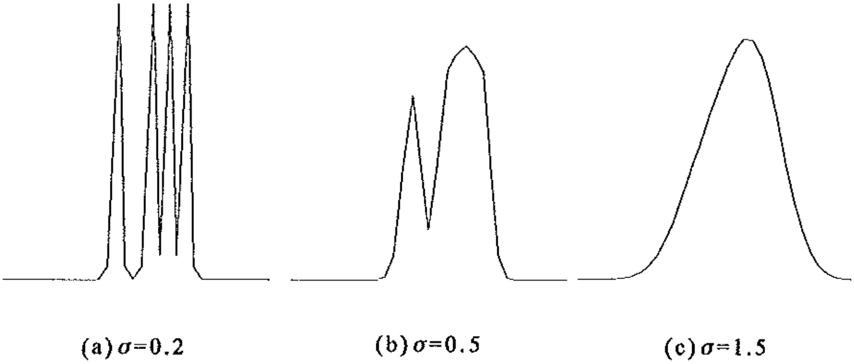


Figure 41: The smoothing effect of different values of σ on a1D PDF estimated from examples.

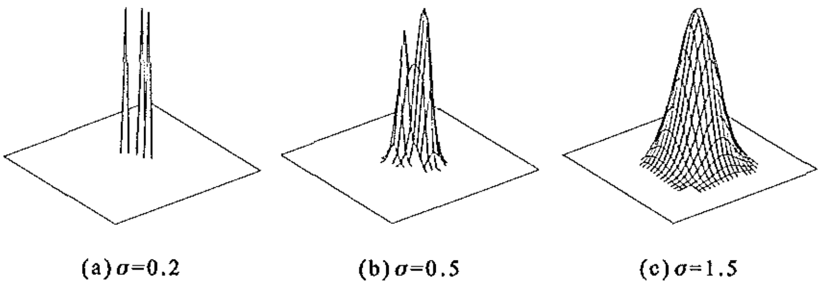


Figure 42: The smoothing effect of different values of σ on a 2D PDF estimated from examples.

3.2.4 The architecture of a Probabilistic Neural Network (PNN)

The many advantages offered by ANNs have prompted an effort to recast the Baysian classifier into the probabilistic neural network (PNN) framework. Figure 43 shows a neural network organization for classification of input patterns X into two categories.

An input vector $X^t = [\, X_1 \; \ldots \; X_p \,]$ to be classified is applied to the PEs of the input layer. This layer is merely a distribution layer which supplies the same input value to all of the patterns PEs. Each pattern PE (shown in more detail in Figure 44) forms a

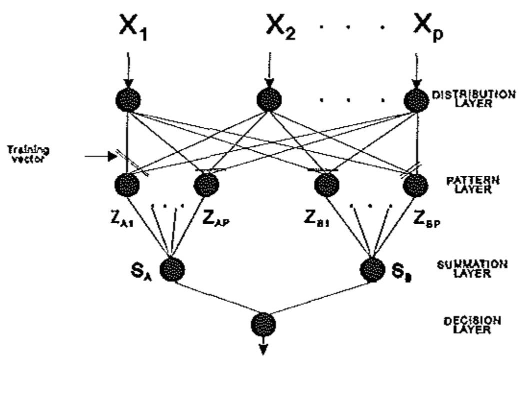


Figure 43: The Probabilistic Neural Network (PNN).

dot product of each input pattern vector X with a weight vector W_i and then performs a nonlinear operation on Z_i ($Z_i = X \cdot W_i$) before outputting its activation level to the summation PE.

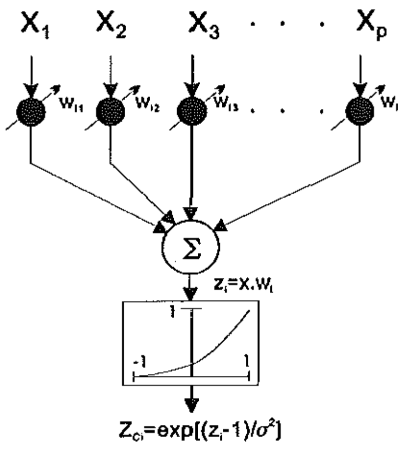


Figure 44: The pattern PE functional model.

Instead of the Sigmoid function (i.e., bounded above and below, but differentiable) commonly used for BPN networks, the nonlinear operation used here is:

$$\exp\left[\frac{(Z_i - 1)}{\sigma^2}\right] \qquad 7$$

Assuming that both X and W_i are normalized to unit length, we have $X^tX = W^tW = 1$ and the expression $(X-W_i)^t(X-W_i) = (X^tX - 2X^tW_i + W^t_iW_i)$ becomes $-2(X^tW_i - 1) = -2(Z_i - 1)$ and Eq.(7) is equivalent to:

$$\exp\left[-\frac{(X - W_i)^t(X - W_i)}{2\sigma^2}\right] \qquad 8$$

The set of weights entering a pattern layer PE represent a specific training vector; each weight has the value of a component of that training vector (i.e, $W_i = Y_{Ai}$, or $W_i[j] = Y_{Ai}[j]$, for class A and $W_i = Y_{Bi}$ or $W_i[j] = Y_{Bi}[j]$ for class B). Thus, the resulting output is :

$$Z_{Ci} = \exp\left[\frac{(Z_i - 1)}{\sigma^2}\right] = \exp\left[-\frac{(X - W_i)^t(X - W_i)}{2\sigma^2}\right] = \exp\left[-\frac{(X - Y_{Ci})^t(X - Y_{Ci})}{2\sigma^2}\right] \qquad 9$$

Each PE in the **summation layer** receives all pattern layer outputs associated with a given class. Thus, the output of each summation layer PE is

$$S_C = \sum_{i=1} Z_{Ci} = \sum_{i=1} \exp\left[\frac{(Z_i - 1)}{\sigma^2}\right] = \sum_{i=1} \exp\left(-\frac{(X - Y_{Ci})^t (X - Y_{Ci})}{2\sigma^2}\right) \qquad 10$$

This is exactly the form needed to implement Eq.(5). The output, or decision PE is a two-input PE as shown in Figure 45. This PE produces binary output. It has only a single variable weight K:

$$K = -S = -\frac{h_B l_B}{h_A l_A} \cdot \frac{n_A}{n_B}$$

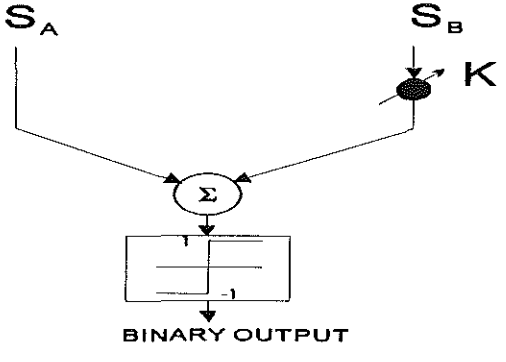


Figure 45: The output or decision PE.

3.3 Advantages and disadvantages of the PNN

The PNN approach offers many advantages. It trains virtually instantaneously (i.e., the training time is zero!), the days or weeks of iterative training of other ANNs are replaced by little more than reading the training set. The PNN paradigm allows data to be added or deleted from the training set without lengthy retraining. This characteristic of the PNN makes it more compatible with many real-world problems, since network learning (as human learning) is often a continuing process. Some disadvantages stem from the fact that the entire training set must be stored, as well as processed, each time an unknown case is to be classified. This means that memory requirements are large, and execution speed is low. This approach is hardly suitable for real-time applications, unless a hardware implementation is available.

3.4 Error Back-Propagation Network (BPN)

3.4.1 Introduction

In this section we study an important class of neural networks, namely, error back-propagation networks. BPNs have been applied successfully to solve some difficult and diverse problems by training them in a supervised manner with an algorithm known as the error back-propagation algorithm. This algorithm is based on the error-correction learning rule and is also referred to in the literature as the back-propagation algorithm. The learning process performed with the algorithm is called back-propagation learning. The conceptual basis of BPNs was first presented in 1974 by Werbos [25], then independently reinvented by Parker in 1985[24], and presented to a wide readership in 1986 by Rumelhart [23]. Back-propagation has been much studied in the past few years, and many extensions and modifications have been considered. Only the basic form of BPN is discussed here.

3.4.2 Network architecture

Typically, back-propagation employs three or more layers of processing elements (units, neurons). Figure 46 shows the architecture of a typical four-layer back-propagation network. The two internal layers are hidden layers (with hidden neurons). The network shown here is fully connected, which means that a neuron in any layer of the network in connected to all the neurons in the previous layer. Signal flow through the network progresses in a forward direction, from left to right and on a layer- by-layer basis.

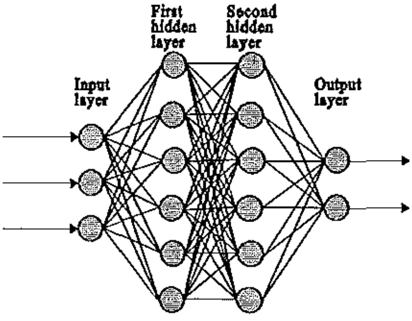


Figure 46: A four-layered back-propagation network.

A back-propagation neural network is trained by supervised learning. The output, o_j, of any neuron j can be expressed as the non-linear transfer function, f, of the input, net_j. Where net_j is the dot product of the output of the previous layer (containing N nodes), o_k, and a weight vector, as shown below:

$$o_j = f(net_j) \quad \text{where:} \quad net_j = \sum_{k=1}^{N} w_{kj} o_k \qquad 11$$

Thus the output of each neuron in the final layer is a non-linear function of the inputs and all the weight matrices. Generally the output functions of the processing elements in a back-propagation network are sigmoid transfer functions:

$$f(net) = \frac{1}{1 + e^{-net+\theta}} \qquad 12$$

This function acts as a soft threshold with the center of the slope at θ. Figure 47 shows the sigmoid function.

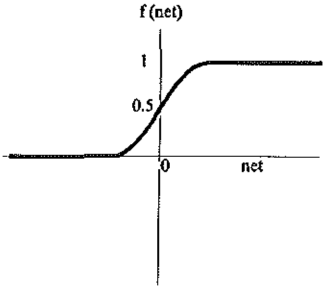


Figure 47: The sigmoid function.

This function has three important properties. Firstly, the sigmoid is non-linear, allowing the network to perform complex mappings of input to output vector spaces. Secondly it is continuous and differentiable which allows the gradient of the error to be used in updating the weights, and thirdly the function satisfies the following differential equality:

$$f'(net) = f(net)[1 - f(net)] \qquad 13$$

3.4.3 The Back-Propagation Training Algorithm

The error back propagation learning rule is probably the most widely used method to train feed-forward networks. The basic idea of this learning rule is to define a measure of the overall performance of the system and then to find a way to optimize that performance. We can define the performance of the system as:

$$E = \sum_{k=1}^{K} E^k = \tfrac{1}{2}\sum_{k=1}^{K}\left\|t^k - o^k\right\|^2 = \tfrac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{m}(t^{ki} - o^{ki})^2 \qquad 14$$

where

o^{ki} = the i.th component of the actual output vector produced by presenting the k.th training input pattern x^k

t^{ki} = the i.th component of the k.th desired output vector

E^k = the square error of the output for training input pattern x^k.

E = the sum of the square error of all training patterns.

m = the number of output neurons.

k = the number of patterns in the training set.

The goal is to minimize this sum. If the system error is zero, all training patterns are mapped on the correct target output pattern. If not, we can assign a particular neuron blame in proportion to the degree to which changes in that neuron's activity lead to changes in the error. That is, we change the weights of the system in proportion to the derivatives of the error with respect to the weights. The rule for changing weights is given by the gradient descent method, i.e. we minimize the error function E by using the following iteration process:

$$w_L^{mn}(new) = w_L^{mn}(old) - \eta \frac{\partial E}{\partial w_L^{mn}} \qquad 15$$

where

w_L^{mn} = the weight of the connection between the n^{th} neuron in layer $L-1$ and the m^{th} neuron in layer L

η = a positive consant that controls the rate of change of the weights (learning rate)
Substituting eq.14 for E in eq.15 gives:

$$w_L^{mn}(new) = w_L^{mn}(old) - \eta \frac{\partial\left\{\sum_{k=1}^{K} E^k\right\}}{\partial w_L^{mn}} = \sum_{k=1}^{K}\left\{-\eta \frac{\partial E^k}{\partial w_L^{mn}}\right\} \qquad 16$$

After some mathematical manipulation, the expression for the updates of the weights of the connections to the neurons of the output layer L is obtained:

$$w_L^{mn}(new) = w_L^{mn}(old) + \sum_{k=1}^{K} \eta \delta_L^{km} o_{L-1}^{kn} = w_L^{mn}(old) + \sum_{k=1}^{K} \eta (t_m^k - o_m^k) f'(net_L^{km}) o_{L-1}^{kn} \qquad 17$$

where:

O_{L-1}^{km} = the output of the m^{th} neuron in layer $L-1$ for an input pattern k

$\delta_L^{km} = (t_m^k - o_m^k) f'(net_L^{km})$,
a local error measure for the m. th neuron in layer L, due to training input pattern x^k

$f'(net_L^{km})$ = the derivative of the activation function $f(net_L^{km})$

The expression for the updates of weights in a hidden layer is:

$$w_L^{mn}(new) = w_L^{mn}(old) + \sum_{k=1}^{K} \eta \delta_L^{km} o_{L-1}^{kn} \qquad 18$$

where the local error measures (δ's) for a processing element m in a hidden layer k can be determined recursively by:

$$\delta_L^{km} = \sum_{l} \delta_{L+1}^{l} w_{L+1}^{lm} f'(net_L^{km}) \qquad 19$$

A complete description and derivation of the gradient descent method is given in Parallel Distributed Processing [32].

For a given training set, back-propagation learning may proceed in one of two basic ways:

1. Batch Mode. In batch mode of back-propagation learning, weight updating is performed after the presentation of all the training examples that constitute an epoch (one complete presentation of the

entire training set during the learning process is called an epoch). The above described weight updating is a batch mode of back-propagation learning.

2. Pattern Mode. In pattern mode (or online-mode) of back-propagation learning, weight updating is performed after presentation of each training example. The weight update for a pattern k becomes:

$$w_L^{kmn}(new) = w_L^{kmn}(old) + \eta\delta_L^{km}o_{L-1}^{kn} \qquad 20$$

The use of pattern mode requires less local storage, and makes the search in weight space stochastic in nature, which, in turn, makes it less likely for the back-propagation algorithm to be trapped in a local minimum. The use of batch mode of training provides a more accurate estimate of the gradient vector. Experiments show that in the general case when the weight updates are small (η sufficiently small), the two training modes yield comparable solutions, and their relative effectiveness depends on the problem at hand.

3.4.4 Advantages and disadvantages of BPNs

The back-propagation neural network offers many disadvantages. It is very sensitive to initial weights, and the back-propagation algorithm often converges very slowly to the solution and can get stuck in local minima of the cost function. For complex problems it may require days or weeks to train the network. The choice of the number of layers and neurons per layer is still an unsolved problem. No general rule exists to determine the required network size for a certain application. In most applications the network size is chosen in a heuristic or empirical way. The back-propagation neural network has no reject option.

A great advantage of the back-propagation neural network is its generalization capability. Another advantage of the BPN is its general applicability. It can be used for both continuous and binary mapping for many different types of problems.

3.5 Application of the selected neural networks to daily life motor activities

3.5.1 Construction of a training set

To train a neural network, one must have labeled examples of input data. These data may come from databases, simulations, expert opinions or reference data (i.e., data-sets obtained from recording under predefined conditions). In our case, the latter type of data was used. To provide the reference data for building the training sets, the instrumented subject follows during 15-30 minutes a protocol consisting of a number of daily life motor activities. A particular training set must be representative for its class and must be unambiguous.

The way in which a training set for a daily life motor activity class is defined depends on the definition of each activity class. Each daily life motor activity class is reflected in specific postures and movements. Each specific posture and movement is reflected in typical waveforms of the signal in each channel. Some of the daily life activity classes can be defined by using one single channel, others need two, three or sometimes all four channels. The use of more channels to define some of the activity classes reduces the classification errors. Figure 48 shows representative signals for some of the activity classes. The class definitions are based on the occurrence of specific waveforms and combinations of the accelerometer signals.

The use of only one single channel (first channel or second channel) is sufficient to define a step (class

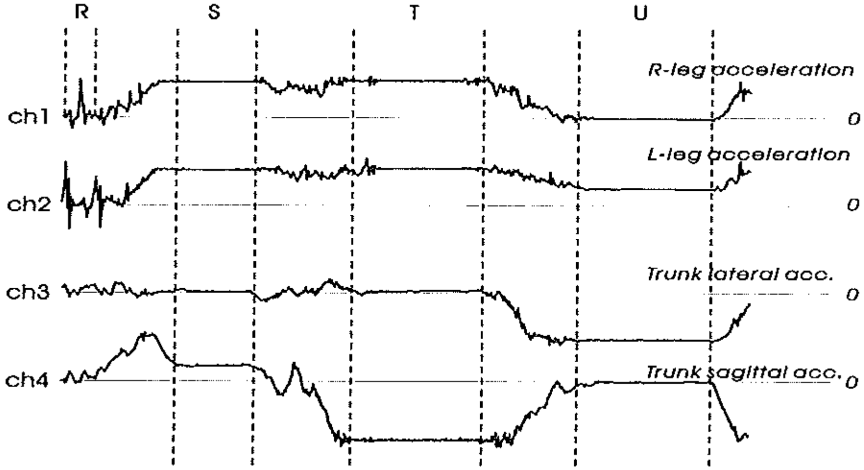


Figure 48: Representative signals for some activity classes recorded during calibration.

R). For the definition of 'sitting' (class S) and 'lying on the back' (class T), we used two channels (first and fourth channel). For lying on the left side (class U) we used the third and the fourth channel. In mathematical notation:

ClassR : {trainsetRch1, Ø, Ø, Ø}

ClassS : {trainsetSch1, Ø, Ø, trainsetSch4}

ClassT : {trainsetTch1, Ø, Ø, trainsetTch4}

ClassU : {Ø, Ø, trainsetUch3, trainsetUch4}

where Ø specifies a non-used channel.

Thus we have to choose one training set for class R and two training sets for each of the classes S, T and U (one for each channel). Each training set which is fed to the ANNs contains a number of class-specific waveforms.

TrainsetRch1={example$_1$-classR-ch1, example$_2$-classR-ch1,..........example$_i$-classR-ch1}

TrainsetSch1={example$_1$-classS-ch1, example$_2$-classS-ch1,..........example$_j$-classS-ch1}

TrainsetSch4={example$_1$-classS-ch4, example$_2$-classS-ch4,..........example$_k$-classS-ch4}

TrainsetTch1={example$_1$-classT-ch1, example$_2$-classT-ch1,..........example$_l$-classT-ch1}

$TrainsetTch4=\{example_1\text{-}classT\text{-}ch4, example_2\text{-}classT\text{-}ch4, ... example_m\text{-}classT\text{-}ch4\}$

$TrainsetUch3=\{example_1\text{-}classU\text{-}ch3, example_2\text{-}classU\text{-}ch3, ... example_n\text{-}classU\text{-}ch3\}$

$TrainsetUch4=\{example_1\text{-}classU\text{-}ch4, example_2\text{-}classU\text{-}ch4, ... example_o\text{-}classU\text{-}ch4\}$

Generally speaking, the more examples one can collect for a training set, the better. Unfortunately, no general rules exist for the calculation of the appropriate number because this depends on the complexity of the application. It should be noted that all examples in each training set must have the same length (dimension), however, two different training sets may contain examples with different dimension. The above training sets can be used either for PNN or BPN.

3.5.2 Activity detection with Probabilistic Neural Network (PNN)

Now we are in a position to design a PNN-classifier to recognize and classify the daily life motor activity. It has already been shown in the previous section how to construct a training set. In the following subsection, we will use the training set in the storing phase, the entire collected data during 15-30 minutes in the adjusting phase and the total data of 10-12 hours in the running phase.

3.5.2.1 Storing phase

Figure 49 shows the PNN-classifier in the storing phase for the above example. In the storing phase, each DataProcessor-Ch* module reads a Trainset*ch* (the training set) and feeds it to a related Class*-ch*-PNN (a PNN for a two-class problem). A peculiar feature in our PNN-classifier is the parallel use of a column of PNNs for a two-class problem instead of using a single PNN for a multi-class problem for each channel. This is so devised, because a PNN for a multi-class problem would require that the dimensionality of the examples in all training set be the same. The fixed size of the input of a neural network has led us not only to use a column of PNNs, but it makes the PNN-classifier a patient-dependent system (i.e. for every patient, we have to build a new training set, thus finding a new smoothing parameter and changing the input dimension of PNN-classifier).

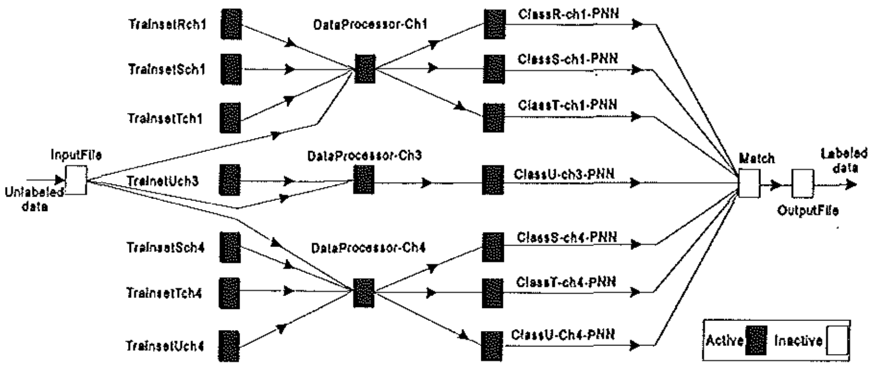


Figure 49: A schematic diagram of PNN-classifier in the storing phase.

Figure 50 illustrates 10 examples of training set TrainsetRch1 which have been fed to a ClassR-ch1-PNN neural network in order to recognize a step pattern. The intra-subject variability of step's pattern can be clearly seen.

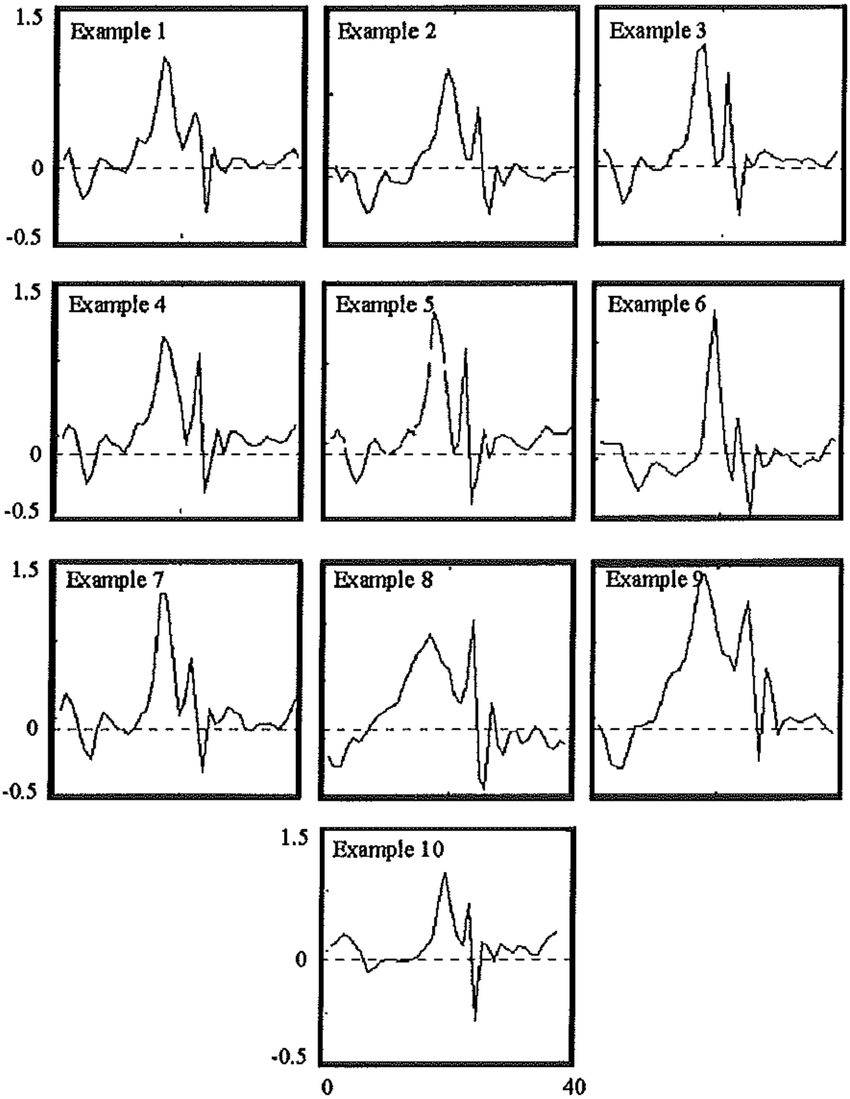


Figure 50: Several examples of step's pattern.

The value of the training pattern "Example 1" in ASCII format looks as follows:

1.0000E-02, 8.0000E-02,-1.2000E-01,-2.4000E-01,-2.1000E-01,-8.0000E-02, 2.0000E-02,-1.0000E-02,-5.0000E-02,-5.0000E-02,-8.0000E-02, 2.0000E-02, 1.4000E-01, 1.2000E-01, 1.6000E-01, 3.8000E-01, 6.7000E-01, 6.2000E-01, 2.3000E-01, 7.0000E-02, 1.8000E-01, 3.1000E-01, 2.2000E-01,-3.3000E-01, 8.0000E-02,-4.0000E-02,-8.0000E-02, 2.0000E-02, 2.0000E-02, 1.0000E-02,-4.0000E-02,-4.0000E-02,-1.0000E-02,-2.0000E-02,-2.0000E-02, 0.0000E+00, 5.0000E-02, 7.0000E-02, 2.0000E-02.

Figure 51 shows a tiled graphical displays of the above 10 examples of the training set TrainsetRch1. Each row represents the value of an example array, and each tile in this row represents the value of a single array element. The size of the tile increases with the value of the element.

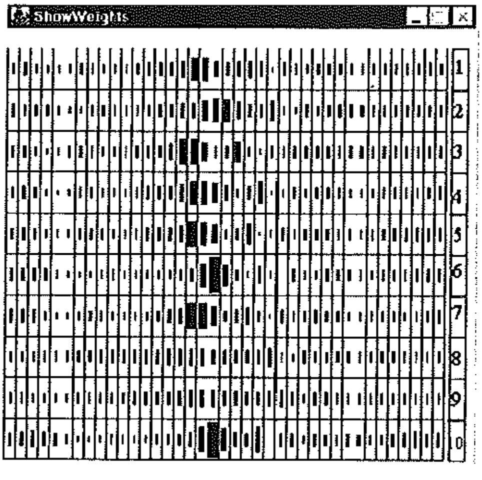


Figure 51: Tiled graphical displays of "training" examples.

3.5.2.2 Adjusting phase

In order to properly evaluate the network, it has to be fed with data which were not used in training and check the results. This can be done by step-wise scrolling a window with the size of the pattern to be recognized one bit each step, over the test file that contains the pattern. If the contents of the window approaches one of the examples from the training set with a probability density that exceeds a given set value, the PNN network outputs a '1'. This means that a more or less matching pattern is found. It gives '-1' if the probability density is lower than the set comparison value in cases that the contents of the window does not properly match the examples from the training set. If the results are correct, the PNN is ready to use. If not, more or better training sets must be obtained, or the smoothing parameter must be adjusted. Figure 52 shows the PNN-classifier in the adjusting phase for the above example, where the Inputfile module reads the unlabeled data set which has to be classified and labeled, the Match module is used to combine and compare the outputs of all PNN modules (this combining and comparing is based on the definition of the activity classes) which results in the assignment (or not) of a label to a part of the inputdata, the OutputFile module writes the results in a file.

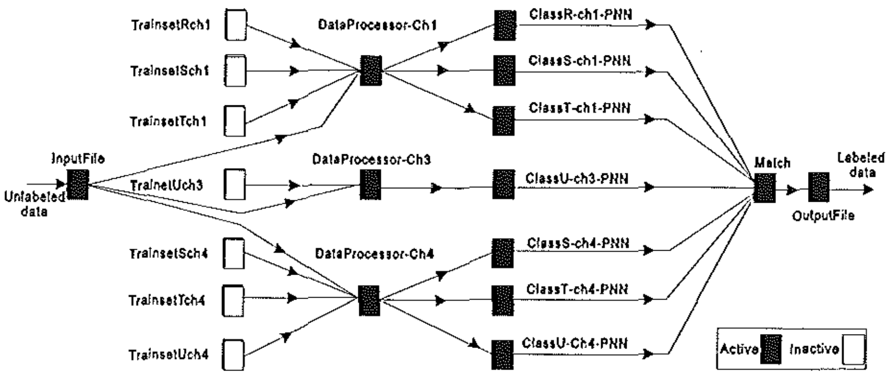


Figure 52: A schematic diagram of PNN-classifier in an adjusting and run phase.

By altering the smoothing parameter, the generalization ability of the PNN can be adjusted. Our experiences for selecting a good smoothing parameter can be summarized in the following simple rules:

- The more examples are available in the training set, the smaller the smoothing parameter value should be.
- The more noisy the data is, the larger the smoothing parameter value should be.
- A small value of the smoothing parameter leads to overlearning (or overfitting or overtraining).
- A large value of the smoothing parameter leads to underfitting (or overgeneralization).
- The best procedure is to try a few values. In our application, the value of the smoothing parameter ranges from 0.1-0.3.

It has been found that in practical problems it is not difficult to find a proper value for the smoothing parameter, and that the misclassification rate does not vary dramatically with small changes in the smoothing parameter.

In the adjusting phase, we used a software package which can visualize the collected data and visually compared the results of the PNN-classifier with the test data set.

3.5.2.3 Running phase

The running the PNN-classifier consists of presenting it with 10-12 hours input data and gathering the results. Figure 53 shows the outputs of four sensors, the step's pattern which has been detected by the Class-ch1-PNN neural network is highlighted by a rectangle. Figure 54 shows the output of the Class-ch1-PNN neural network.

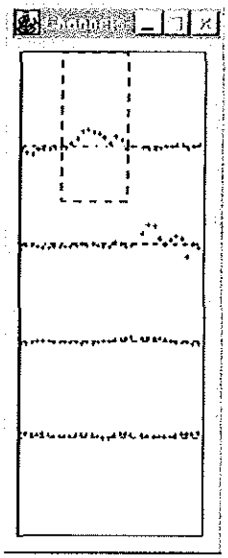


Figure 53: The output
of four sensors.

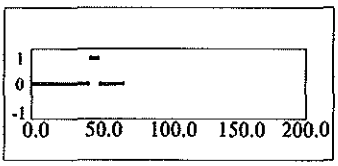


Figure 54: The output of ClassR-ch1-
PNN.

3.5.3 Activity detection with an error back-propagation neural network

The back-propagation neural network was the second neural network which has been applied to the recognition of daily life motor activity. The strategy consists of three phases: configuration, training and running phase. The training sets can easily be constructed as discussed in section 3.5.1.

3.5.3.1 Configuration phase

When a back-propagation neural network is to be applied to solve the recognition of daily life motor activities problem, one is confronted with the following practical considerations:

- number of activity classes;
- number of training examples;

- number of layers;
- number of neurons on each layer;
- initialization of weights;
- batch updating or sample updating;
- setting learning rate η and momentum term α;
- choosing the sort of activation function;
- stopping criterion;
- output representation.

Unfortunately, the above mentioned considerations are not independent of each other. Almost all choices have implications, although some are stronger and some are weaker. To find an optimal setting is almost an optimization problem itself.

3.5.3.2 Training phase

The basics of the training of a BPN were discussed in section 3.4.3. Figure 55 shows the BPN-classifier in a training phase for the classification of the four motor activities as discussed in section 3.5.1. In this phase, each Class*-ch*-BPN (a back-propagation neural network) is trained independently by a Trainset*ch* (the training set). As Figure 55 shows, a column of BPNs for a one-class problem is employed instead of using a single BPN for a multi-class problem. This is so devised, because just like the case of PNN, a BPN for a multi-class problem would require patterns of the same dimensionality.

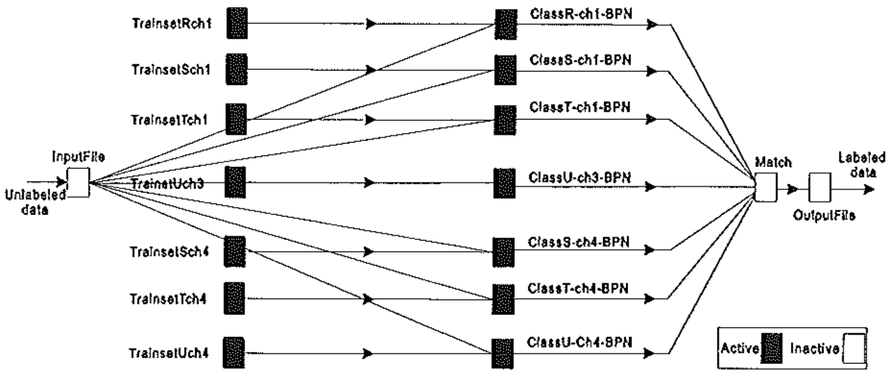


Figure 55: A schematic diagram of BPN-classifier in the training phase.

3.5.3.3 Running phase

Running the BPN-classifier consists of presenting it with 10-12 hours input data and gathering the results. Unlike the PNN which can generate a binary output (found/not found), the BPN is not able to reject a sample (unknown pattern) in a statistically significant way. The unknown pattern is usually classified into a class with low probability (high Mean Squared Error (MSE)). To solve the reject category problem, the error statistics of each BPN output such as MSE can improve the classification performance. Unknown samples can be rejected if the probability is below a certain threshold. Figure 56 shows a schematic diagram of BPN-classifier in the training phase.

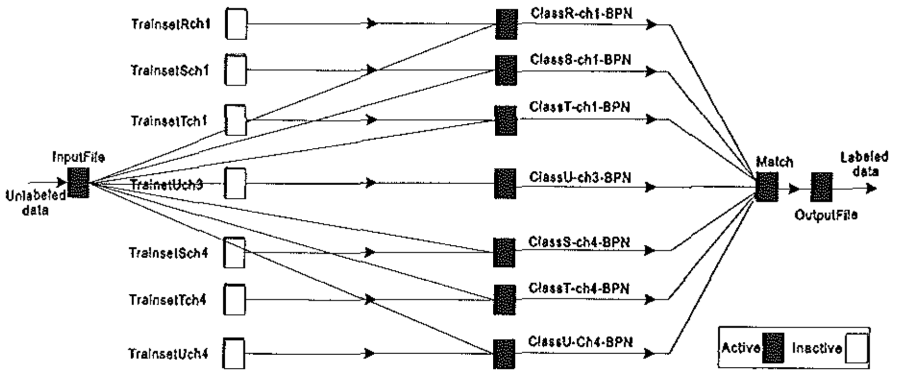


Figure 56: A schematic diagram of BPN-classifier in the Running phase.

Figure 57 shows the probability of classes on the outputs of the above BPN neural networks, when a step pattern [class R] was presented.

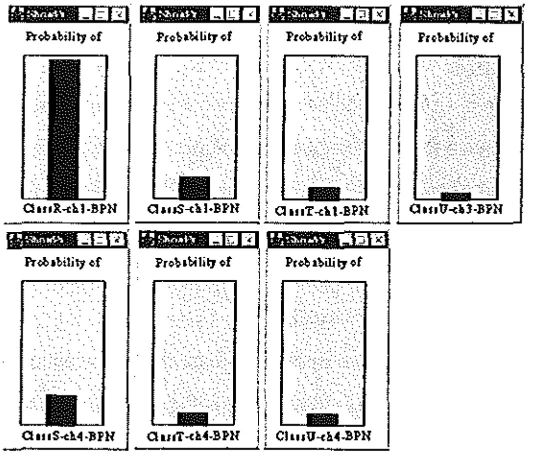


Figure 57: The error statistics of seven BPN outputs.

In this case, the class R [Step] was recognized with high probability, all other classes have low probability.

Figure 58 illustrates the probability of classes on the BPN outputs of the BPN-classifier, when a pattern not belonging to any of the classes is presented. Each BPN assigns such a pattern to a class with a certain probability, but the pattern can be rejected if all output values are below a certain probability threshold. Experiments have shown that by increasing the number of examples in each training set, the BPN outputs become smaller in cases, when such patterns are presented.

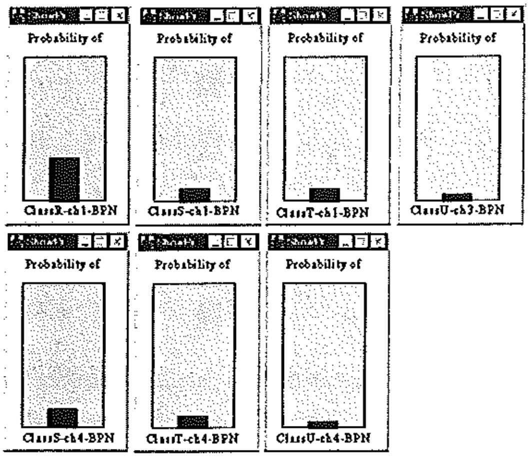


Figure 58: The error statistics of seven BPN outputs.

3.6 Performance estimation

The performance of a system is an indication of what the system accomplishes. In developing an ADL-classifier it is important to estimate its performance during the running phase (using phase). For ADL-classification, the estimation of performance can be obtained by calculating the percentage of correctly classified cases. Alternatively, the number of incorrect classification can be used; this is often referred to as the error rate. Ideally the estimation is based on an unlimited number of pattern exemplars, but, in practical situations limited amount of data is available.

To estimate the performance of the PNN-and BPN-classifiers, the following approach was used:

- divide the entire data collected during 15-30 minutes (collected according to a specific protocol) into two parts: one part (20%) to be used during the development of the classifier and the other part (80%) to be used to test the classifier. If the performance of the ADL-classifier on this test set is considered satisfactory, the ADL-classifier is accepted for use.

In addition to the above approach for obtaining the classification error rate we have also compared the output of the ADL-classifiers with the manually labeled data of about 10 hours of each subject. Since for each subject a different ADL-classifier has been trained, an average classification error is calculated for all subject.

3.7 Post-processing

To refine the onset and end time of each activity as estimated by the two above mentioned ADL-classifiers, one has to use some explicit knowledge to postprocess the output of the used ADL-classifier. Figure 59 shows how the highest peak of a step pattern can be estimated correctly by a postprocessing algorithm. The E indicates the onset time of the detected step pattern and P shows the location of the highest peak of the step pattern. This postprocessing algorithm shifts forward a window from the estimated onset time of a step pattern. The local maximum in this window presents the highest peak of the step pattern. The size of the window should be as large as the pattern size.

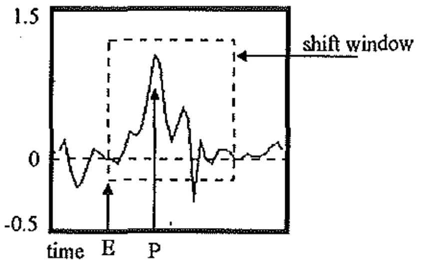


Figure 59:An example of a step pattern in which the highest peak can easily be estimated by using postprocessing algorithm.

To fine-tune the estimated onset and end time of other activities by ADL-classifier, one has to devise a good postprocessing algorithm.

3.8 Subjects

To test and verify our system, the daily life activities of a group of eight male amputees were measured. The subjects were instrumented in the morning and the instruments were removed approximately 11 hours later. As mentioned earlier, the recorded data are downloaded from the RAMCORDER to a computer for further processing. In Table 3 some information about the subjects and the recording time is given.

Subject	Age	Affected (above knee amputation)	Years post-amputation	Recording time
1	60	L	53	10h:25m
2	60	L	37	10h:18m
3	34	R	15	10h:51m
4	62	L	44	11h:18m
5	47	R	18	10h:41m
6	67	R	9	10h:55m
7	56	L	41	11h:21m
8	40	L	20	11h:21m

Table 3: Some characteristics of subjects.

3.9 Results

3.9.1 Results of PNN-classifier

The PNN-classifiers were, on the average, able to recognize 95% of all presented cases of the daily life activity classes of all subjects correctly. Because of a short duration of a number of specific activities, the PNN-classifiers were unable to recognize those activities. Another reason for misclassification(i.e., not recognizing the activity) was the occurrence of a waveform pattern which was not included in the training sets. In the following, we present some activity profiles which have been extracted from the numerical output of our PNN-classifier.

Figure 60 shows a typical example of an 'Activity Profile'. Here the sequence and the duration of the activities walking, sitting and standing are presented.

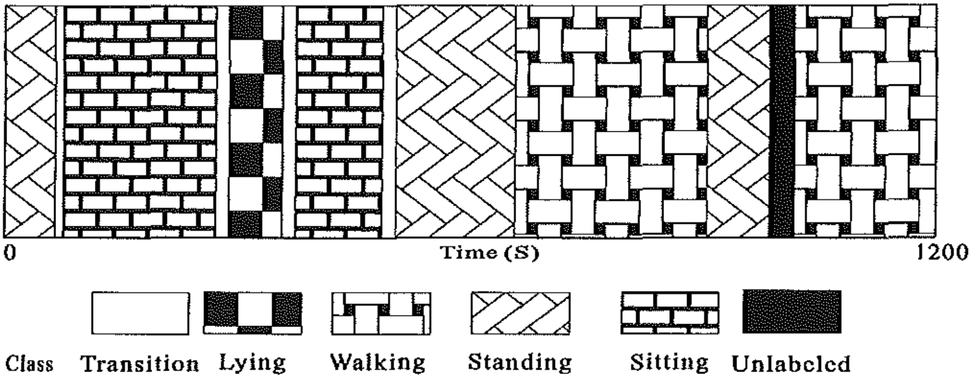


Figure 60: Graphical presentation of an activity profile: sequence and duration of daily life motor activity classes over a period of 20 minutes.

In Figure 60 only 20 minutes of the data-set is shown because displaying a longer recording period would mask some of the recognized activities. The signal waveforms that could not be recognized by the PNN-classifier are indicated as unlabeled. All transition activities, e.g., from standing to sitting, from sitting to lying, etc. are shown as transition.

An overview of the distribution in time of daily-life activity classes of all amputees during a long term recording is presented in the pie graph in Figure 61. This figure shows the contribution of each activity as a percentage of the total recording time. In this figure the 'transition' and 'unclassified' cases are put together. The transition time is the time which lasts between two different activity classes e.g. the transition from sitting to standing , etc.

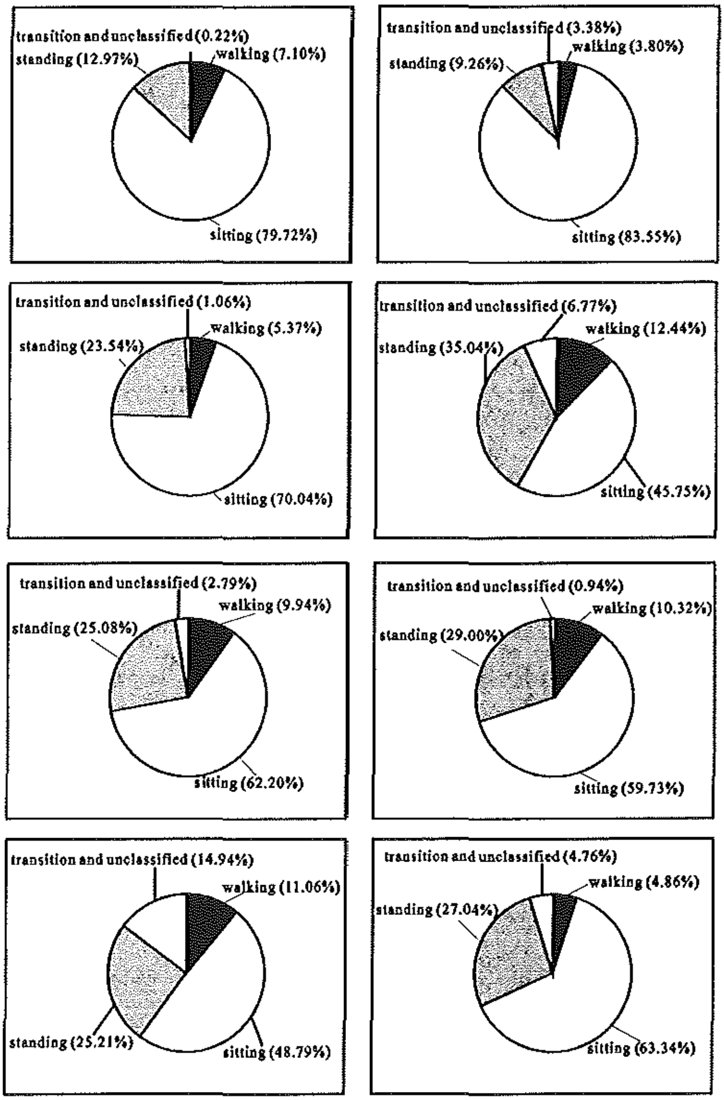


Figure 61: Overview of an activity profile: distribution of activities as a percentage of long term recording time for eight amputees

Figure 62 shows eight histograms in which the horizontal axis displays the duration of the activity 'walking' divided in category-intervals of 10 seconds, and the vertical axis displays the frequency of each category-interval as recorded during the total recording time.

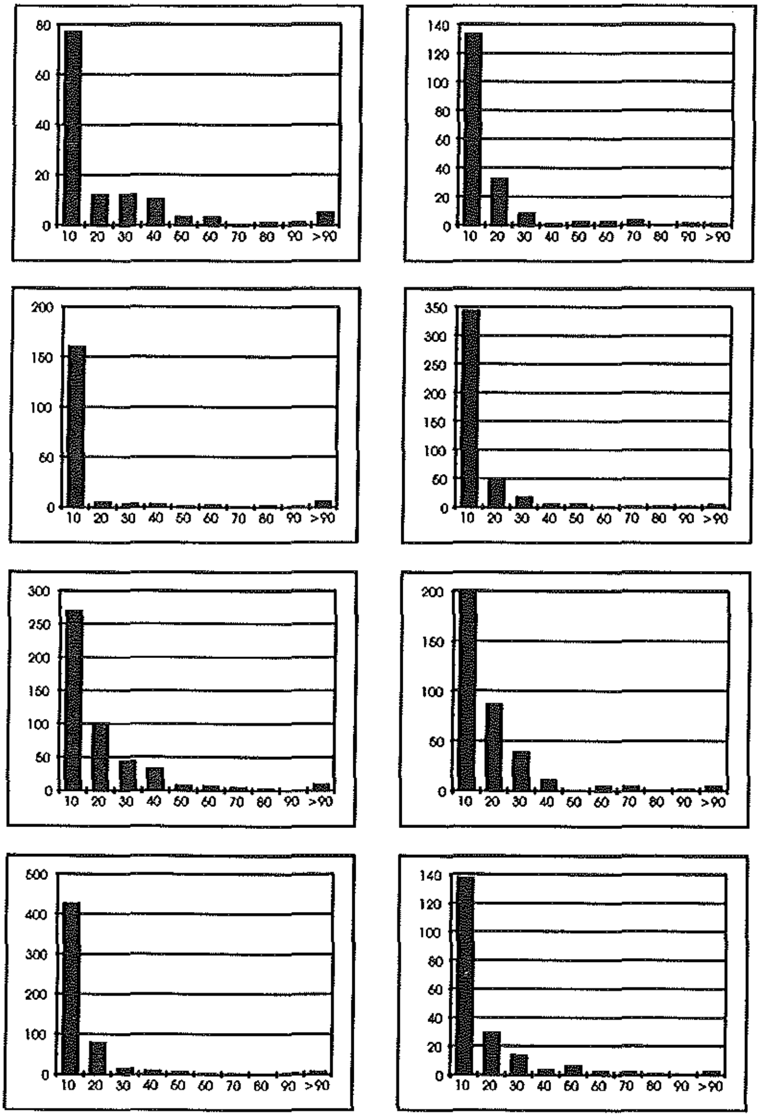


Figure 62: Histogram of an activity profile: the activity walking is divided in category-intervals (walking blocks) of 10 seconds.

3.9.2 Results of BPN-classifier

Experiments using the BPN involve the setting of a large number of parameters that influence the BPN's operation and it is not feasible to investigate the behavior of all possible network configurations.

Various experiments with varying parameters have been made. For the BPN-classifier consisting of a column BPN, only networks with one hidden layer were used. The number of neurons in this layer as well as in the output layer were varied in order to evaluate the performance of such a network in various configurations. The results of these experiments are presented in Table 4 and Table 5.

Number of hidden neurons	Number of output neurons	Percentage of correct recognition
5	1	94%
6	1	93%
7	1	93%
8	1	94%
9	1	91%
10	1	94%

Table 4: Performance of a BPN-classifier on the recognition of the daily life motor activities for various numbers of hidden neurons.

As can be seen from the performance figures in Table 4, increasing the number of nodes in the hidden layer only marginally affects the performance of the BPN-classifier.

Number of hidden neurons	Number of output neurons	Percentage of correct recognition
5	1	94%
5	2	97%
5	3	97%
5	4	96%
5	5	95%
5	6	93%
5	7	94%

Table 5: Performance of a BPN-classifier on the recognition of the daily life motor activities for various numbers of output neurons.

It can be seen from the results in Table 5 that the performance increases with an increasing number of output neurons up to a size of 3 neurons; use of a lager number of output neurons does not show a further positive effect on the BPN-classifier's performance.

As a next step, studying the performance of the BPN-classifier in response to increasing the number of hidden layers was investigated. Results of such an experiment are presented in Table 6. From the results presented in this table it can be seen that an increase of the number of hidden layers decrease the performance of the BPN-classifiers. In all these experiment, the number of example patterns in the training set had been held constant. The number of example patterns in the training set is given in Table 7.

Number of hidden layers	Number of output neurons	Percentage of correct recognition
1	3	97%
2	3	96%
3	3	87%
4	3	89%
5	3	89%
6	3	84%

Table 6: Performance of a BPN-classifier on the recognition of the daily life motor activities for various numbers of hidden layers. The number of hidden neurons in each layer is 5.

Activity	The number of example patterns in the training set
Sitting	30
Standing	30
Step	75
Lying on the back	25
Lying on the left side	25
Lying on the right side	25

Table 7: The number of example patterns in the training set for each activity.

3.10 Discussion and Conclusion

The use of artificial neural networks in the field of rehabilitation is limited [33][34]. The analysis and interpretation of daily life motor activities and related clinical parameters are of importance in clinical applications such as analysis of motor activities in post- and pre-medication. In addition, the performance of a specific operation can be assessed by comparing the motor activities before and after the operation. Also, in ECG applications, the motor activities recorded simultaneous with the ECG signal may help to get a better picture of heart diseases. No satisfactory methods exist to monitor the daily life motor activities from large amounts of data obtained during sessions of 10 hours or more of continuous recordings of ambulatory patients who randomly perform daily life activities at home or at work. This means that the manual analysis of activities of a subject (with huge amounts of data) takes months to be performed. Therefore, an automated approach is necessary.

Although in this study the data of amputees of our previous projects were used, in general, the above approach can be applied to any application, which need the analysis of the daily life motor activities.

Since the daily life motor activities are very complex and show extremely large inter- and intra-individual variation, a simple threshold technique will provide a low classification accuracy. Also, using other techniques including regular signal processing tools such as smoothing, Fourier analysis, etc. requires magic numbers to obtain a reliable accuracy. However, there is no guarantee to find such numbers in a noisy environment such as recorded motor activities.

The accuracy of classification obtained by an ADL-classifier is very reliable, since this network can generalize and recognize similar patterns, even in noisy environments. The networks are well-controlled by many setting parameters and by preprocessing (e.g., filtering and offset correction).

The comparison of the output of the automatic classifier with the classification via visual inspection of the events resulted in 95% conformity. In 5% of the events, automatic classification was not possible because of too short duration of a certain activity or the occurrence of activities not included in the training set.

From this study it can be concluded that the PNN and BPN classifier are potentially useful tools for the classification of daily life motor activities.

By means of several ways of graphical representation we were able to show typical characteristics of the daily life pattern of amputees at work. However, the instrumentation allows for ambulatory recording of motor activities in all possible circumstances like outdoor recreation, transport in vehicles and activities at home. Furthermore other quantities like heart rate, EMG, temperature, light and sound intensity, etc., can be recorded. The sensors are light and small, and are hidden under the clothes and do not hinder normal behavior.

From this study we derived the following conclusions:

- daily life motor activities are complex and show extremely large inter- and intra-individual variation which excludes using regular signal processing tools for recognition;
- for the application under consideration, PNNs and BPNs are potentially useful options;
- a satisfactory conformity of 95% between automatic and visual classification of events can be achieved;
- the automatic classification of 10 hours of activities takes less computation time with special hardware;
- the graphical presentation of the output yields clinically meaningful information;
- the ADL-classifier is patient-dependent which means that for every patient a training set has to be built and optimal parameters settings have to be chosen.
- Application of a postprocessing algorithm does improve the determination of the onset and end time of each activity.
- Because of the many setting parameters and the time consuming training process of BPNs, using a PNN-classifier is much easier than using a BPN-classifier.
- Because of the reject class option on PNN output, the performance of PNN is more reliable than that of a BPN.

References

[1] Gelsema E.S., Kanal L.N. : Pattern recognition in practice IV: Multiple paradigms, Comparative studies and hybrid systems. Proceedings of an International Workshop held on Vlieland, The Netherlands, June 1-3, 1994, North-Holland Amsterdam.

[2] Wong AJ. : Recognition of general patterns using neural networks." Biological Cybernetics, 1988;58:361-372.

[3] Raychaudhuri T, Chieh-hong Yeh J, Harney LGC. Westcott CT : Baked product classification with the use of a self-organizing map. Proceedings of the sixth Australian conference on Neural Networks;1995 Feb 6-8; Sydney. Sydney: Dept. of Electrical Engineering, University of Sydney, Australia,1995;152-155.

[4] Hornik KM, Stinchcombe M, White H. :Multilayer feedforward networks are universal approximators. Neural Networks 1989;2:359-366.

[5] Favata F, Walker R. :A study of the application of Kohonen-type neural networks to t he traveling salesman problem. Biological Cybernetics 1991;64:463-468.

[6] Azoff EM. :Neural network time series forecasting of financial markets. New York: John Wiley & Sons, 1994

[7] Van der Smagt P, Groen FCA., Kröse BJA. :A monocular robot arm can be neurally positioned. Proceedings of International Conference on Intelligent Autonomous Systems (IAS-4);1995,1995;123-130. IOS. Press 1995.

[8] Lippmann. R.P.: Review of Neural Network for Speech Recognition. Neural Computation, 1989;1:1-38.

[9] Back AD, Tsoi AC. : FIR and IIR synapses, a new neural network architecture for times series modeling. Neural Computation 1991;3(3):375-385.

[10] Narendra KS, Parthasarathy K. : Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Network 1990;1(1):4-27.

[11] Rumelhart D.E, Hinton G.E, Williams R.J : Learning internal representation by error propagation, in Parallel distributed processing: explorations of the microstructure of cognition, vol. 1: Foundations (D.E. Rumelhart and J.L. McClelland eds.), MIT Press, 1986.

[12] Kohonen T. :An introduction to neural computing, Neural Network, 1988 1(2):3-16.

[13] McCulloch W.S, Pitts W. : A logical calculus of the ideas imminent in nervous activity, Bulletin of Mathematical Biophysics, 1943; 5:115-133.

[14] Hebb D.O. : The organization of behavior, New York: Wiley, 1949.

[15] Minsky M., Papert S., : Perceptrons: an introduction to computational geometry, MIT Press, 1969.

[16] Grossberg S. : How does a brain build a cognitive code?, Psychological Review, 87:1-51, 1980.

[17] Grossberg S. : Neural expectation: Cerebellar and retinal analogs of cells fired by learnable or unlearned pattern classes, Kybernetik, 1972, 10:49-57..

[18] Grossberg S. : Adaptive pattern classification and universal recording: I. Parallel development and coding of neural detectors, Biological Cybernetics, 1976a, 23:121-134

[19] Grossberg S. : Adaptive pattern classification and universal recording: II. Feedback, expectation, olfaction, illusions, Biological Cybernetics, 1976b, 23:121-134

[20] Kohonen T. : Self-organized formation of topologically correct feature map, Biological Cybernetics, 1982a, 43:59-69.

[21] Cohen M.A. Grossberg S. : Absolute stability of global pattern formation and parallel memory storage by competitive neural network, IEEE Transactions on systems, man and Cybernetics, 1983, 13:815-826.

[22] Ackley D.H, Hinton G.E, and Sejnowski T.J, : A learning algorithm for boltzmann machines, Cognitive Science, 1985, 9:147-169.

[23] Rumelhart D.E, Hinton G.E, and Williams R.J : Learning internal representation by error propagation, In Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1986, chapter 8, MIT Press.

[24] Parker D.B. : Learning Logic, Technical Report TR-47, Center for computational Research in Economics and Management Science, MIT, April 1985.

[25] Werbos P. J., :Beyond regression: New tools for prediction and analysis in the behavioral sciences, PhD dissertation, Committee on Applied Mathematics, Harvard University, Cambridge. MA, November 1974.

[26] Broomhead D.S, and Lowe D. : Multivariable functional interpolation and adaptive networks, Complex System, 1988, 2:321-355.

[27] Anderson J.A. and Rosenfeld, eds : Neurocomputing: Foundation of Research. Cambridge, MA: MIT Press, 1988.

[28] Meisel W.S., : Computer-oriented approaches to pattern recognition, New York and London: Academic Press, 1972.

[29] Duda. R.O. and Hart. P.E. : Pattern classification and Scene Analysis. New York: Wiley-Interscience, 1973

[30] Specht D., : Probabilistic Neural Networks. Neural Networks1990;3:109-118.

[31] Parzan E., : On estimation of a probability density function and mode., Ann. Math. Stat, 1962, 33:1065-1076.

[32] Rumelhart D.E. and McClelland, J.L., :Parallel distributed processing: explorations of the microstructure of cognition, MIT Press, 1986

[33] Holzreiter SH, Köhle ME. Assessment of gait patterns using neural networks. J. Biomechanics, 1993;26:645-651.

[34] Gonzalez LP, Arnaldo CM. Classification of drug-induced behavior using a multi-layer feed-forward neural network. Computer Methods and Programs in Biomedicine 1993;40:167-173.

Chapter 4

Fuzzy Rule-Based Classification

This chapter is a generalization of the following paper:

Computerized analysis of daily life motor activity for ambulatory monitoring

Kourosh. Kiani[*MSc], Chris. J. Snijders[*Phd], Edzard. S. Gelsema[**Phd]

Faculty of Medicine and Allied Health Sciences, Erasmus University Rotterdam

* Department of Biomedical Physics and Technology

** Department of Medical Informatics

Accepted by Technology and Health Care (Official Journal of the European Society for Engineering and Medicine), Elsevier, 1997

4. Fuzzy Rule-Based Classification

4.1 Introduction

Fuzzy logic was introduced in 1956 by Zadeh [1] as a new way to provide a mathematical framework to capture the uncertainties associated with human cognitive systems such as thinking and reasoning. It is a generalization of conventional (Boolean) two-valued logic, and it uses "soft" linguistic (e.g. large, tall, small) values for system variables and a continuous range of values in the interval [0, 1], rather than strict binary (True or False) decisions and assignments. It has been applied very successfully in many areas where conventional based approaches are difficult to implement. Classification is one of those areas. In this chapter, we describe a fuzzy ruled-based approach for the recognition of daily life motor activities.

4.2 Fuzzy Sets and Membership

Let X be a space of objects and x be a generic element of X. A classical set (crisp) A is defined as a collection of elements $x \in X$, such that an element x in the universe X is either a member of set A or it is not. This binary property of membership can be represented mathematically with the characteristic function,

$$\chi_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

where $\chi_A(x)$ indicates membership of element x in set A. For illustration, suppose set A is the crisp set of all people with $35 \leq x \leq 55$ year in the universe of age of people, shown in Figure 63.

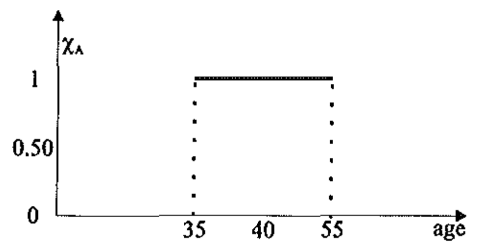


Figure 63: Age membership function for a crisp set A.

A particular individual, x_1, has an age of 40 years. The membership of this individual in crisp set A is equal to 1, or full membership, given symbolically as $\chi_A(x_1) = 1$. Another individual, say, x_2, has an age of 34.99 year. The membership of this individual in crisp set A is equal to 0, or no membership, hence $\chi_A(x_1) = 0$, also seen in Figure 63. In these cases the membership in a set is binary, either an element is a member of a set or it is not.

Unlike the classical (crisp) set mentioned above, a fuzzy set expresses the "degree of membership" to which an element belongs to a set. Hence the characteristic function of a fuzzy set is allowed to have values between 0 and 1, denoting the degree of membership of an element in a given set. If X is a collection of objects denoted generically by x, then a fuzzy set A in X is defined as a set of ordered pairs:

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \mu_A(x_3)/x_3 + \mu_A(x_4)/x_4 + \mu_A(x_5)/x_5 + \ldots$$

$\mu_A(x)$ is called the membership of x in A, which maps X onto [0, 1].

As an example, consider the membership functions for the fuzzy variable height. In Figure 64, the membership functions $\mu_{Tall}(x)$, $\mu_{Average}(x)$ and $\mu_{Short}(x)$ are defined graphically, where height is indicated along the x axis of the graph, and degree of set membership of the corresponding height is given by the y coordinate. Thus, the extent to which a height of 1.79m is "tall" is 0.50, and the extent to which it is "average" is 0.25. These can be presented by the ordered pairs (1.79, 0.50) and (1.79, 0.25) respectively. This shows an important point, namely that an element (in this case a height) can be a member of more than one set.

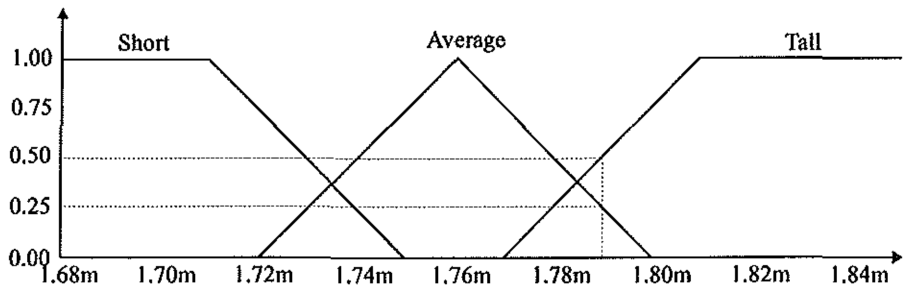


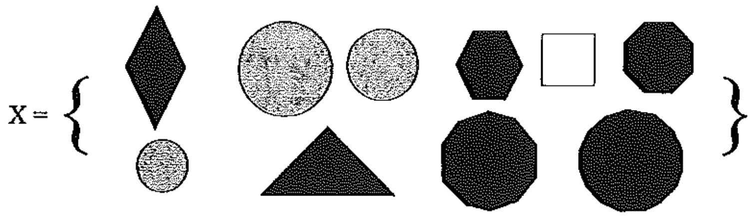
Figure 64: Membership functions for the fuzzy variable "height".

Fuzzy sets can be defined discrete or continuous, or can be defined using examples of set members, in any way desired. It is also possible to define them mathematically (functional representation); for example the set "tall" can be defined as:

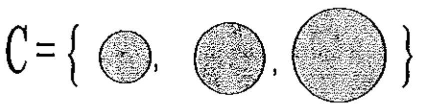
$$\mu_{Tall}(h) = \begin{cases} 0 & h \le 1.77 \\ \dfrac{h-1.77}{0.04} & 1.77 \le h < 1.81 \\ 1 & h \ge 1.81 \end{cases}$$

where h is height.

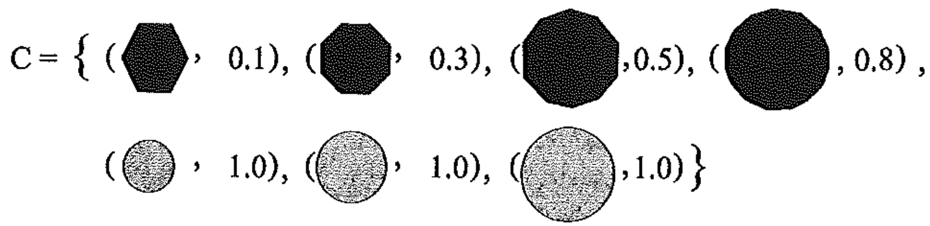
The following example shows a universe of shapes:



One can define the crisp set "circles" as:



The fuzzy set "circles can be defined as:



A simple example of a discrete universe and a discrete fuzzy subset of it, is:

X={-3, -2, -1, 0, 1, 2, 3, 4}

A=0.6/-3 + 0.0/-2 + 0.3/-1 + 0.6/0.0 + 1.0/1 + 0.6/2 + 0.3/3 + 0.5/4

Figure 65 shows the fuzzy set A graphically.

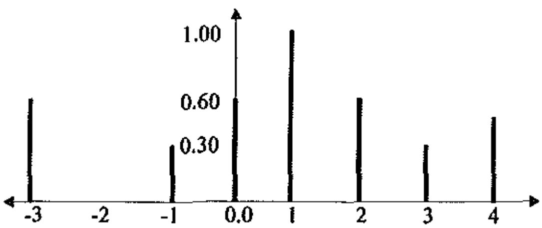


Figure 65: A discrete fuzzy set.

4.3 Operation on Fuzzy Sets

In order to manipulate fuzzy sets, it is necessary to have operations that enable us to combine them. Corresponding to the ordinary logical operations, i.e., AND, OR and COMPLEMENT, fuzzy sets have similar operations. As with the definition of membership functions, there is no single recognized set of fuzzy operations. In the following subsection, the fuzzy versions of the operations AND, OR and COMPLEMENT will be introduced.

4.3.1 Intersection

As said before, there is no unique way to extend classical logical operations. The intersection of two fuzzy sets A and B is defined as [Zadeh 1965]:

$$\mathrm{AND}(\mu_A(x), \mu_B(x)) = \mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}$$

where A and B are fuzzy subsets of a universe X.

The membership function is obviously the crucial component of a fuzzy set. It is therefore not surprising that operations with fuzzy sets are defined via their membership functions.

Let A and B be fuzzy subsets of the universe X={-3, -2, -1, 0, 1, 2, 3, 4}

A=0.6/-3 + 0.0/-2 + 0.3/-1 + 0.6/0.0 + 1.0/1 + 0.6/2 + 0.3/3 + 0.5/4

B=0.2/-3 + 0.6/-2 + 0.4/-1 + 0.6/0.0 + 0.5/1 + 0.4/2 + 0.5/3 + 0.3/4

$\mu_{A\cap B}$= 0.2/-3 + 0.0/-2 + 0.3/-1 + 0.6/0.0 + 0.5/1 + 0.4/2 + 0.3/3 + 0.3/4

The intersection of A and B is shown in Figure 66

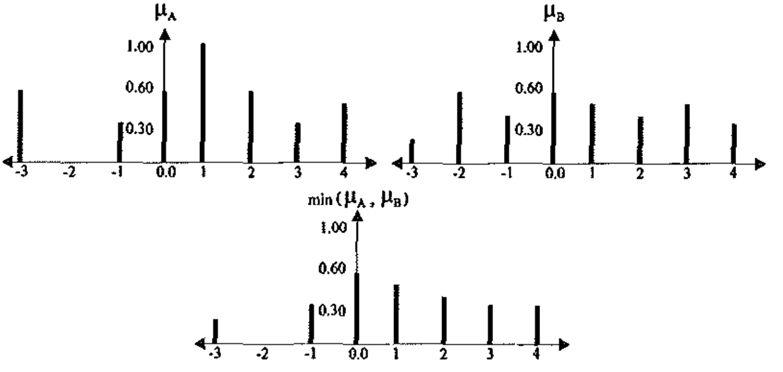


Figure 66: Intersection of fuzzy sets A and B.

Figure 67 shows an intersection of two continuous triangular fuzzy sets with bold lines.

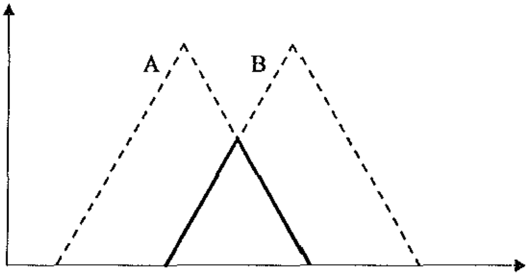


Figure 67: Intersection of two triangular fuzzy sets.

Some other nonparametric AND operators in fuzzy logic are given in Table 8 [2][3][4].

$\mu_A(x)\cdot\mu_B(x)$
$\dfrac{\mu_A(x)\cdot\mu_B(x)}{\mu_A(x)+\mu_B(x)-\mu_A(x)\cdot\mu_B(x)}$
$\dfrac{\mu_A(x)\cdot\mu_B(x)}{2-[\mu_A(x)+\mu_B(x)-\mu_A(x)\cdot\mu_B(x)]}$
$\max\{0,\mu_A(x)+\mu_B(x)-1\}$

Table 8: Possible operators for AND in fuzzy logic.

4.3.2 Union

The union of two fuzzy sets A and B is defined as [Zadeh 1965]:

$$OR(\mu_A(x),\mu_B(x)) = \mu_{A\cup B}(x) = \max\{\mu_A(x),\mu_B(x)\}$$

where A and B are fuzzy subsets of a universe X.

With the subsets A and B as defined in 4.3.1, the union of A and B is:

$\mu_{A\cup B} = 0.6/\text{-}3 + 0.6/\text{-}2 + 0.4/\text{-}1 + 0.6/0.0 + 1.0/1 + 0.6/2 + 0.5/3 + 0.5/4$

This is shown graphically in Figure 68.

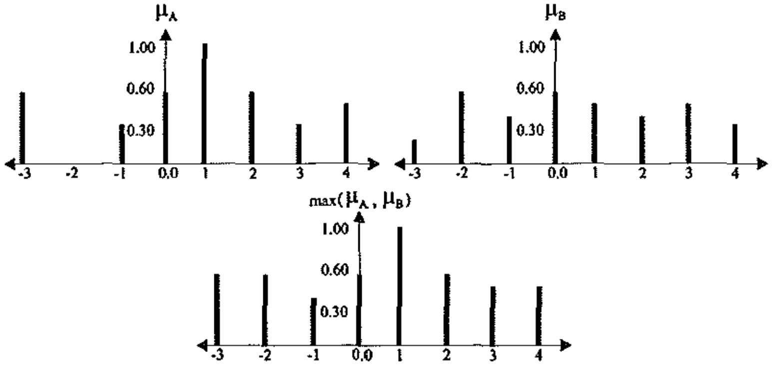


Figure 68: Union of fuzzy sets A and B.

Figure 69 shows the union of two continuous triangular fuzzy sets with bold lines.

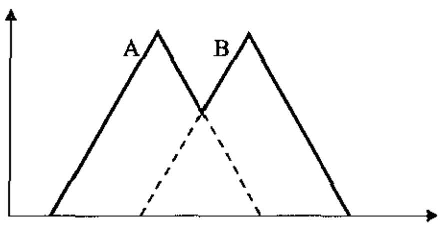


Figure 69: Union of two triangular fuzzy sets.

Some other nonparametric OR operators in fuzzy logic are given in Table 9 [2][3][4]. There are no general guidelines as to which OR or AND operator to choose in a specific situation.

$\mu_A(x)+\mu_B(x)-\mu_A(x)\cdot\mu_B(x)$
$\dfrac{\mu_A(x)+\mu_B(x)-2\mu_A(x)\cdot\mu_B(x)}{1-\mu_A(x)\cdot\mu_B(x)}$
$\dfrac{\mu_A(x)+\mu_B(x)}{1+\mu_A(x)\cdot\mu_B(x)]}$
$\min\{1,\mu_A(x)+\mu_B(x)\}$

Table 9: Possible operators for OR in fuzzy logic.

4.3.3 Complement of fuzzy sets

The Complement of a fuzzy set A is defined as: [Zadeh 1965]

$$\mu_{\bar{A}}(x)=1-\mu_A(x)$$

Let A be a fuzzy subset of universe X={-3, -2, -1, 0, 1, 2, 3, 4}

A=0.6/-3 + 0.0/-2 + 0.3/-1 + 0.6/0.0 + 1.0/1 + 0.6/2 + 0.3/3 + 0.5/4

$\bar{A} = 0.4/-3 + 1/-2 + 0.7/-1 + 0.4/0.0 + 0.0/1 + 0.4/2 + 0.7/3 + 0.5/4$

The complement of A is shown graphically in Figure 70.

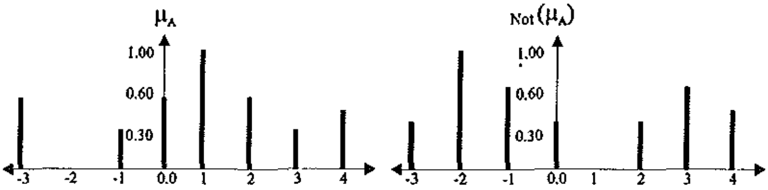


Figure 70: Fuzzy set A [left] and its complement [right].

This operation is shown in Figure 71 for a typical continuous fuzzy set.

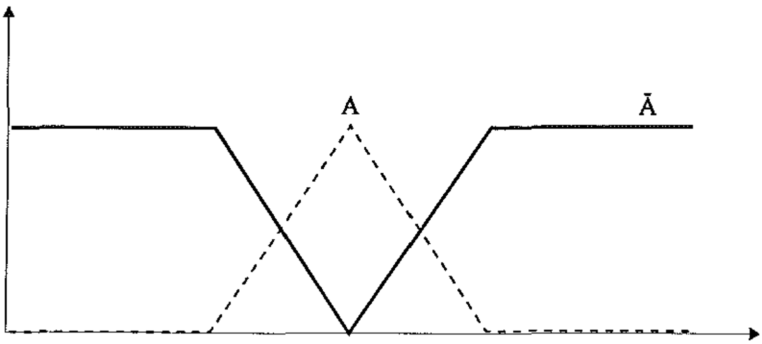


Figure 71: Complement of fuzzy set.

4.4 Fuzzy IF-THEN rules

In the field of artificial intelligence there are various ways to represent knowledge. Perhaps the most common way to represent human knowledge is in the form of natural language expressions of the type,

$$\text{IF } \underbrace{\text{input1 is A AND input2 is B}}_{\text{antecedent part}} \text{ THEN } \underbrace{\text{output is C}}_{\text{consequent part}}$$

where input1, input2 and output are linguistic variables [3], A, B and C are linguistic values that are characterized by membership functions. It typically expresses an inference such that if we know a fact (antecedent), then we can infer, or derive, another fact called a conclusion (consequent). Due to their concise form, fuzzy IF-THEN rules are often employed to capture the imprecise modes of reasoning that play an essential role in the human ability to make decisions in an environment of uncertainty and imprecision.

4.5 Fuzzy inference systems

Fuzzy inference systems perform fuzzy reasoning. Basically a fuzzy inference system is composed of five functional blocks, as depicted in Figure 72.

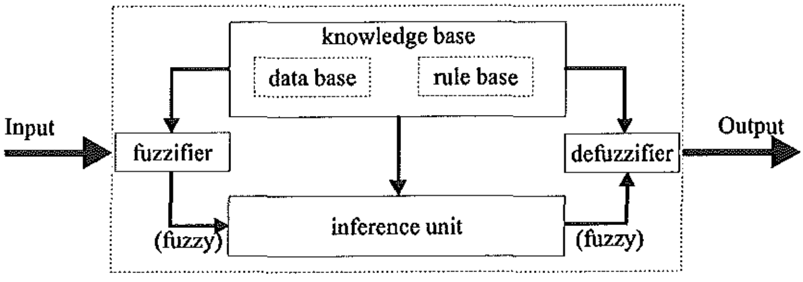


Figure 72: Fuzzy inference system.

- a **data base** contains information about the membership functions of the fuzzy sets used in fuzzy rules, the domains of the variables and kinds of normalization.
- a **rule base** contains a number of fuzzy IF-THEN rules;
- a **fuzzifier** receives the current crisp values of the input variables and transforms them into degrees of match with linguistic values;
- an **inference unit** performs the inference operation on the fuzzy rules;
- a **defuzzifier** transforms the fuzzy results of the inference into a crisp output by using a suitable transformation.

Usually, the rule base and data base are jointly referred to as the knowledge base.

Several types of fuzzy inference systems have been proposed in the past[5][6][7]. They differ in the types of fuzzy reasoning and fuzzy IF-THEN rules employed. In the following, we present two well-known inference mechanisms in fuzzy rule-based systems.

Mamdani uses the following architecture:

Rule 1: IF input1 is A_{11} and input2 is A_{12} THEN output is C_1

also

Rule 2: IF input1 is A_{21} and input2 is A_{22} THEN output is C_2

fact: input1 is x_0 and input2 is y_0

---

consequence: output is C

The fuzzy implication is modeled by Mamdani as:

$$A \text{ and } B \rightarrow C = (A \cap B) \cap C$$

and the operator **also** is interpreted as **Oring** the output of the rules by the **max** operator.

The firing levels of the rules, denoted by α_i, i=1,2, are computed by

$$\alpha_1 = A_{11}(x_0) \cap A_{12}(y_0) = \min\{A_{11}(x_0), A_{12}(y_0)\}$$
$$\alpha_2 = A_{21}(x_0) \cap A_{22}(y_0) = \min\{A_{21}(x_0), A_{22}(y_0)\}$$

The individual rule outputs are computed by

$$C_1'(z) = \alpha_1 \cap C_1(z) = \min\{\alpha_1, C_1(z)\}$$
$$C_2'(z) = \alpha_2 \cap C_2(z) = \min\{\alpha_2, C_2(z)\}$$

Then the overall system output is computed by Oring the individual rule outputs

$$C(z) = C_1'(z) \cup C_2'(z) = \max\{C_1'(z), C_2'(z)\}$$

Figure 73 illustrates the graphical analysis of two rules, where the A_{11} and A_{12} refer to the first and second fuzzy antecedents of the first rule, respectively, and the C_1 refers to the fuzzy consequent of the first rule; the A_{21} and A_{22} refer to the first and second fuzzy antecedents of the second rule, respectively, and the C_2 refers to the fuzzy consequent of the second rule. The minimum membership value for the antecedents propagates through to the consequent and truncates the membership function for the consequent of each rule. This is done for each rule. Then the truncated membership functions for each rule are aggregated. The aggregation operation *max* results in an aggregated membership function. If one wishes to find a crisp value for the aggregated output, some suitable defuzzification technique could be applied to the aggregated membership function, and a value such as z^* shown in Figure 73 would result.

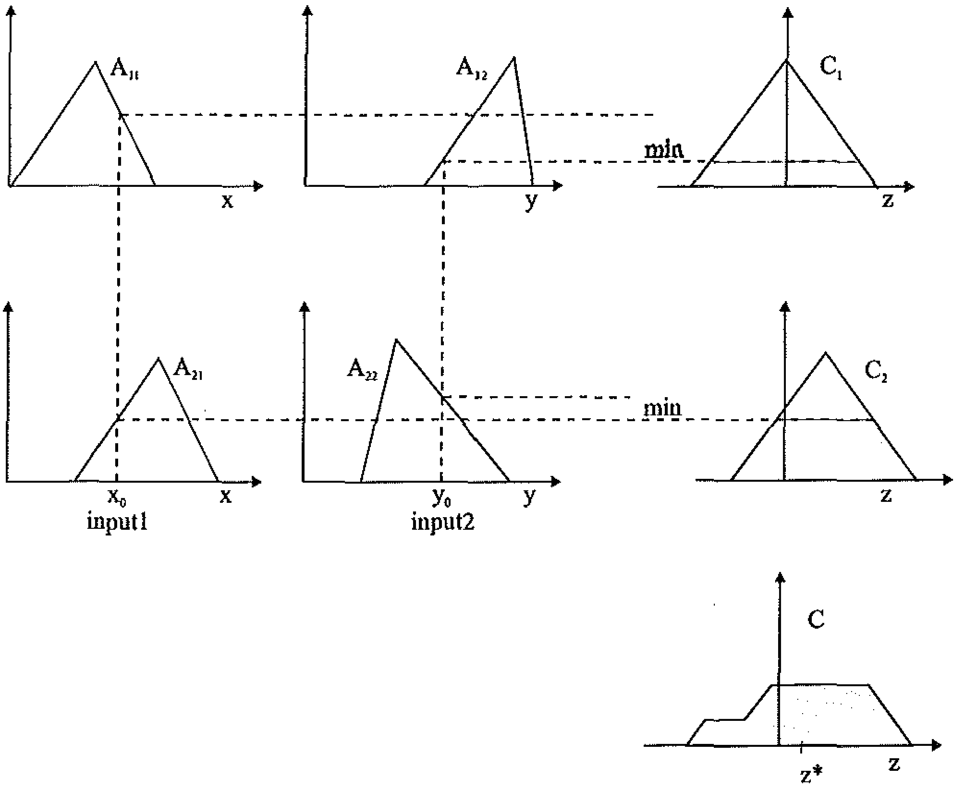


Figure 73: Mamdani's inference mechanism.

Tsukamoto uses the following architecture:

Rule 1: IF input1 is A_{11} and input2 is A_{12} THEN output is C_1

also

Rule 2: IF input1 is A_{21} and input2 is A_{22} THEN output is C_2

fact: input1 is x_0 and input2 is y_0

---

consequence: output is C

All linguistic variables are supposed to have monotonic linguistic value (membership functions).

The firing levels of the rules, denoted by α_i, i=1,2, are computed by

$$\alpha_1 = A_{11}(x_0) \cap A_{12}(y_0) = \min\{A_{11}(x_0), A_{12}(y_0)\}$$
$$\alpha_2 = A_{21}(x_0) \cap A_{22}(y_0) = \min\{A_{21}(x_0), A_{22}(y_0)\}$$

then the individual rule outputs z_1 and z_2 are computed from the equations

$\alpha_1 = C_1(z_1), \qquad \alpha_2 = C_2(z_2)$

and the overall system output is computed by

$$z_0 = \frac{\alpha_1 * z_1 + \alpha_2 * z_2}{\alpha_1 + \alpha_2}$$

Graphically, this is illustrated in Figure 74. With the numeric data given in the figure, we can compute the overall system output as follow:

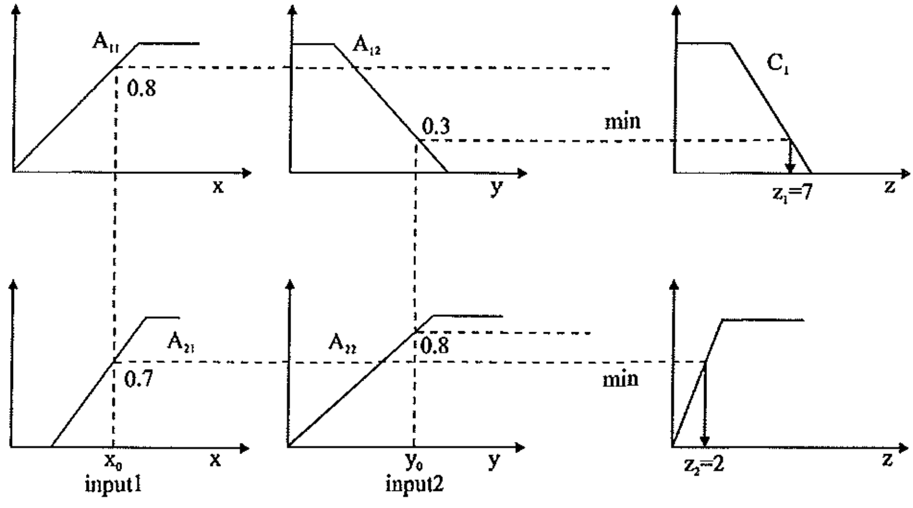


Figure 74: Tsukamoto's inference mechanism.

According to the figure we see that

$$A_{11}(x_0) = 0.8 \qquad \text{and} \qquad A_{12}(y_0) = 0.3$$

therefore, the firing level of the first rule is

$$\alpha_1 = \min\{A_{11}(x_0),\ A_{12}(y_0)\} = \min\{0.8, 0.3\} = 0.3$$

and from

$$A_{21}(x_0) = 0.7 \qquad \text{and} \qquad A_{22}(y_0) = 0.8$$

it follows that the firing level of the second rule is

$$\alpha_2 = \min\{A_{21}(x_0),\ A_{22}(y_0)\} = \min\{0.7, 0.8\} = 0.7$$

the individual rule outputs $z_1 = 7$ and $z_2 = 2$ are derived from the equations

$$C_1(z_1) = 0.3 \qquad \text{and} \qquad C_2(z_2) = 0.7$$

and the overall system output is

$$z_0 = \frac{0.3*7 + 0.7*2}{0.3 + 0.7} = \frac{2.1 + 1.4}{1} = 3.5$$

These two fuzzy inference mechanisms provide a good foundation for a discussion on fuzzy rule-based classification and on hybrid systems.

4.6 Fuzzy classification

Fuzzy classification systems, based on fuzzy logic[8][9][10] are capable to deal with cognitive uncertainties such as the vagueness and ambiguity involved in classification problems. In a fuzzy classification system, an object can be classified by applying a set of fuzzy rules based on the linguistic values of its attributes. Unlike conventional approaches of pattern classification, fuzzy classification assumes the boundary between two neighboring classes as an overlapping area within which a pattern (an object) has partial membership in each class. This viewpoint not only reflects the reality of many applications in which categories have fuzzy boundaries, but also provides a simple representation of the potentially complex partition of the feature space. The classifier is described by fuzzy IF-THEN rules. An example of fuzzy classification rules for a 2-dimensional feature space is:

R_1: *IF* x_1 is **small** *AND* x_2 is **very large** *THEN* $x = (x_1, x_2)$ belongs to class C_1

R_2: *IF* x_1 is **large** *AND* x_2 is **small** *THEN* $x = (x_1, x_2)$ belongs to class C_2

R_3: *IF* x_1 is **small** *AND* x_2 is **large** *THEN* $x = (x_1, x_2)$ belongs to class C_3

R_4: *IF* x_1 is **very small** *AND* x_2 is **very large** *THEN* $x = (x_1, x_2)$ belongs to class C_4

where R_i is the i.th classification rule, C_k indicates an output class, x_1 and x_2 are the features of a pattern (or object), **very small**, **small**, **large** and **very large** are linguistic terms characterized by appropriate membership functions and *AND* is a fuzzy logical operation.

To build a fuzzy classification system, the most difficult task is to find a set of fuzzy rules connected with the specific classification problem. This task can be accomplished in two ways:

1. to acquire knowledge from experts and then translate their knowledge into fuzzy rules[2][3][11].
2. to generate the fuzzy rules automatically from sample data (training set) without expert help[12][13][14][15][16].

In a fuzzy classification system, a classification rule takes the same format as a non-fuzzy classification rule but the inferencing is based on fuzzy logic.

4.6.1 Inference of fuzzy rule based classifiers

For simplicity, let the rule base contain 9 fuzzy IF-THEN rules and have two inputs x and y as follows:

Rule 1: IF input1 is A_1 and input2 is B_1 THEN output is C_1

Rule 2: IF input1 is A_1 and input2 is B_2 THEN output is C_2

Rule 3: IF input1 is A_1 and input2 is B_3 THEN output is C_3

Rule 4: IF input1 is A_2 and input2 is B_1 THEN output is C_4

Rule 5: IF input1 is A_2 and input2 is B_2 THEN output is C_1

Rule 6:	IF input1 is A_2 and input2 is B_3 THEN output is C_6
Rule 7:	IF input1 is A_3 and input2 is B_1 THEN output is C_7
Rule 8:	IF input1 is A_3 and input2 is B_2 THEN output is C_8
Rule 9:	IF input1 is A_3 and input2 is B_3 THEN output is C_6
fact:	input1 is x_0 and input2 is y_0
consequence:	output is C

The firing levels of the rules, denoted by α_i, i=1,9, are computed by

$\alpha_1 = \min\{A_1(x_0), B_1(y_0)\}$
$\alpha_2 = \min\{A_1(x_0), B_2(y_0)\}$
$\alpha_3 = \min\{A_1(x_0), B_3(y_0)\}$
$\alpha_4 = \min\{A_2(x_0), B_1(y_0)\}$
$\alpha_5 = \min\{A_2(x_0), B_2(y_0)\}$
$\alpha_6 = \min\{A_2(x_0), B_3(y0)\}$
$\alpha_7 = \min\{A_3(x_0), B_1(y_0)\}$
$\alpha_8 = \min\{A_3(x_0), B_2(y_0)\}$
$\alpha_9 = \min\{A_3(x_0), B_3(y_0)\}$

If several fuzzy rules have the same consequence class, their firing strengths have to be combined. Usually, the OR operation is used.

The individual rule outputs are computed by

$C_1 = \alpha_1 \text{ OR } \alpha_5 = \max\{\min\{A_1(x_0), B_1(y_0)\}, \min\{A_2(x_0), B_2(y_0)\}\}$
$C_2 = \alpha_2 = \min\{A_1(x_0), B_2(y_0)\}$
$C_3 = \alpha_3 = \min\{A_1(x_0), B_3(y_0)\}$
$C_4 = \alpha_4 = \min\{A_2(x_0), B_1(y_0)\}$
$C_6 = \alpha_6 \text{ OR } \alpha_9 = \max\{\min\{A_2(x_0), B_3(y0)\}, \min\{A_3(x_0), B_3(y_0)\}$
$C_7 = \alpha_7 = \min\{A_3(x_0), B_1(y_0)\}$
$C_8 = \alpha_8 = \min\{A_3(x_0), B_2(y_0)\}$

the overall classifier output is selected by

$C = \max\{C_1, C_2, C_3, C_4, C_6, C_7, C_8\}$

Figure 75 illustrates above the 2-input fuzzy rule base with 9 rules.

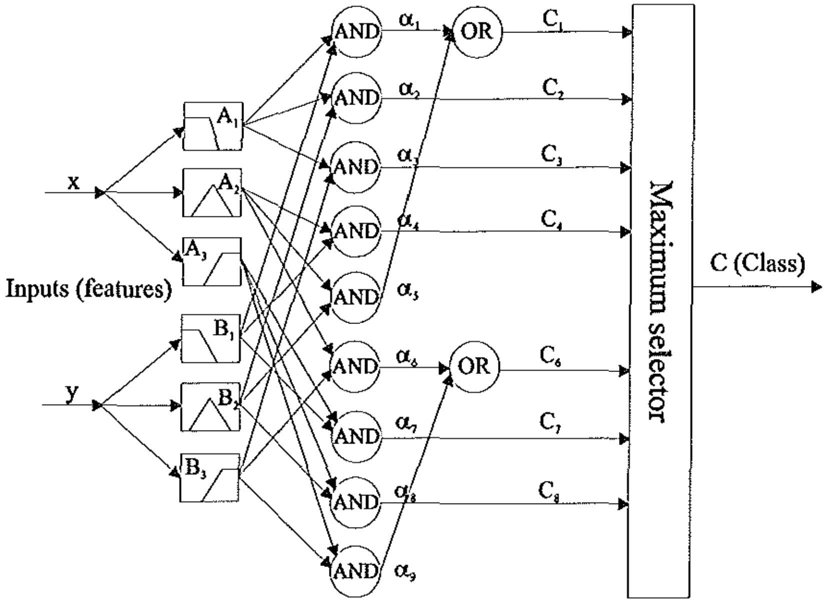


Figure 75: Graphical representation of 2-input fuzzy rule base with 9 rules.

Since each input feature is associated with three memberships, the input space (feature space) is partitioned into 9 fuzzy subspace, each of which is governed by a fuzzy IF-THEN rule. The antecedent part of a rule defines a fuzzy subspace, while the consequent part specifies the output within this fuzzy subspace.

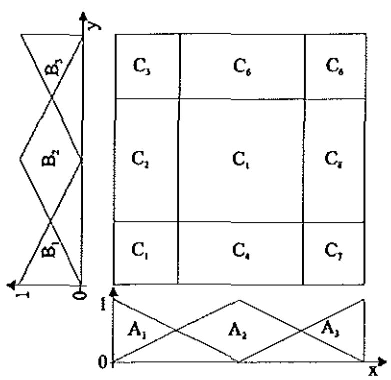


Figure 76: Fuzzy partitioning by 9 rules.

4.7 Application with fuzzy rule based classification

In this section, we explain how the fuzzy IF-THEN rule classifier has been applied to the AMMA-signal for classification of the daily life motor activities. We used the modified pattern recognition system which was discussed in section 2.11, and also shown in Figure 77 as our general model.

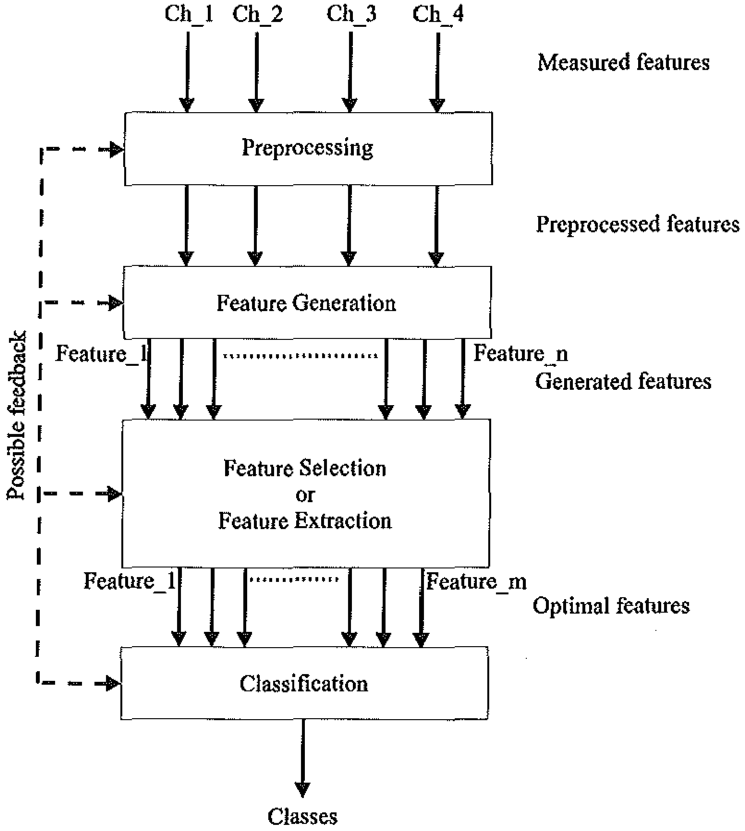


Figure 77: The modified pattern recognition.

Several steps were taken to implement the modified pattern recognition system which uses a fuzzy rule based classifier. In the following subsection, these steps will be discussed.

4.7.1 Feature generation

As mentioned before, using only the four continuous features (the outputs of the four accelerometers) is not enough to design a patient independent classifier, for the recognition of activities (different waves in signals), and the detection of onsets and endpoints of the waves. In the feature generation part, the preprocessed data are transformed into some new representations (new features) in order to maximize the pattern recognition ability and minimize the misclassification rate. In our application, 160 features were generated by using the two algorithms discussed in chapter 2 for each channel. The following figures illustrate some generated features. Figure 78 shows a part of preprocessed activities as recorded by channel 1.

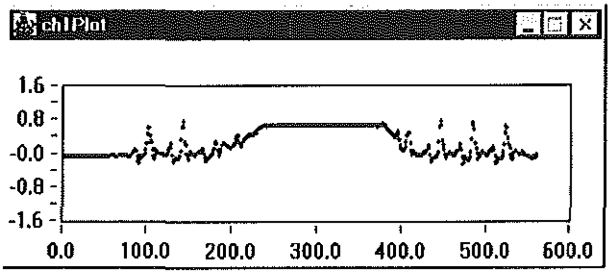


Figure 78: Part of activities recorded by channel 1.

Figure 79 illustrates a generated feature from channel 1 by using the function:
" Power(Norm(ch1[x_1..x_{16}], 2),2)".

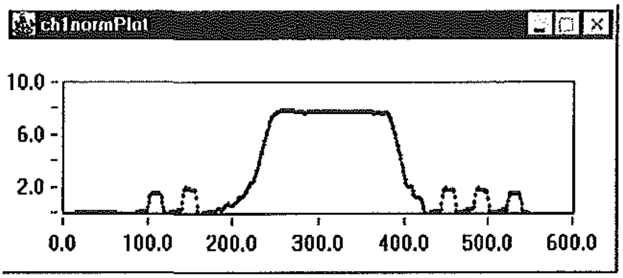


Figure 79: New generated feature from channel 1.

Figure 80 illustrates another generated feature from channel 1 by using the function:
"Norm(Cum_sum(ch1[x_1..x_{16}]),2).

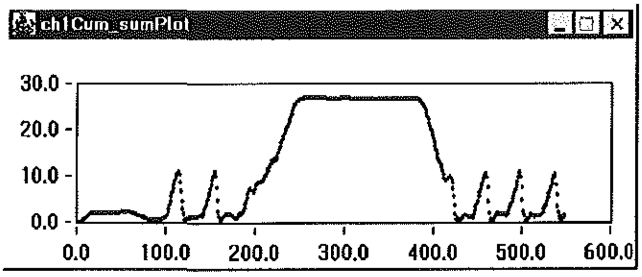


Figure 80: New generated feature from channel 1.

Figure 81 shows a part of preprocessed activities as recorded by channel 2.

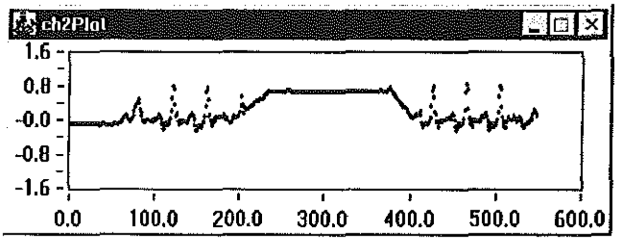


Figure 81: Part of activities recorded by channel 2.

Figure 82 illustrates a generated feature from channel 2 by using the function: "Power(Norm(ch2[x_1..x_{16}], 2),2)".

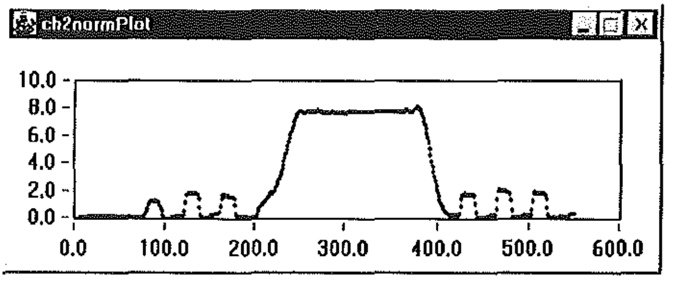


Figure 82: New generated feature from channel 2.

Figure 83 illustrates another generated feature from channel 2 by using the function: "Norm(Cum_sum(ch2[x_1..x_{16}]),2).

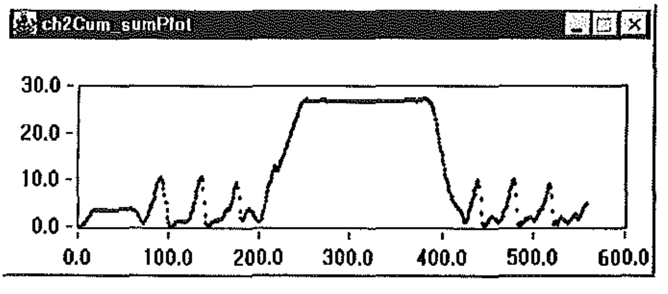


Figure 83: New generated feature from channel 2.

Figure 84 shows an other part of preprocessed activities as recorded by channel 1 for two periods of lying on the back and same other postures.

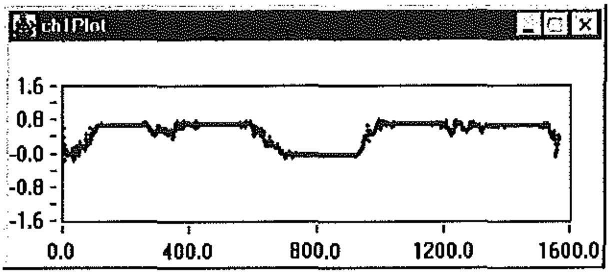


Figure 84: Part of activities recorded by channel 1.

Figure 85 illustrates a generated feature from channel 1 by using the function: "Power(Norm(ch1[$x_1..x_{16}$], 2),2)".

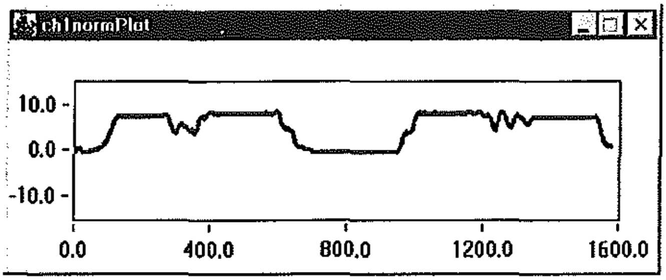


Figure 85: New generated feature from channel 1.

Figure 86 illustrates a generated feature from for channel 1 by using the function: "Standard_deviation(Norm[$x_1..x_8$])".

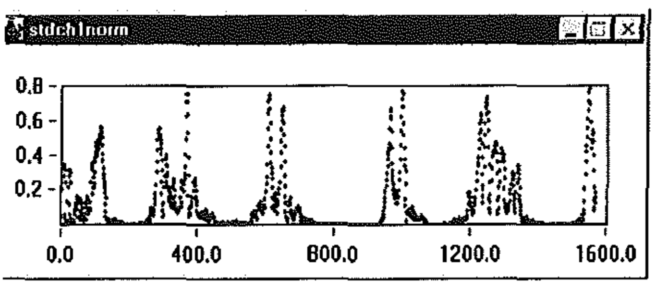


Figure 86: New generated feature from other feature.

Figure 87 illustrates another generated feature from channel 1 by using the function: "Norm(Cum_sum(ch1[x_1..x_{16}]),2)".

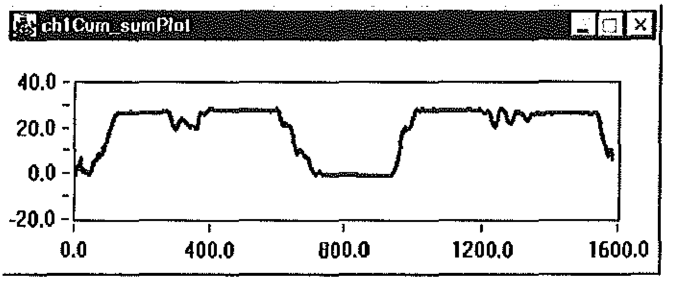


Figure 87: New generated feature from channel 1.

Figure 88 illustrates a generated feature for channel 1 by using the function: "Standard_deviation(Cum_sum[x_1..x_8])".

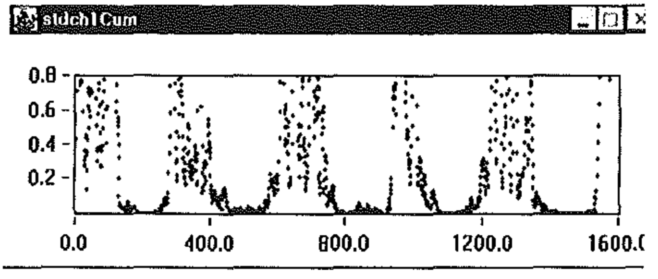


Figure 88: New generated feature from other feature.

Figure 89 shows a part of preprocessed activities as recorded by channel 4 for two periods of lying on the back and same other postures.

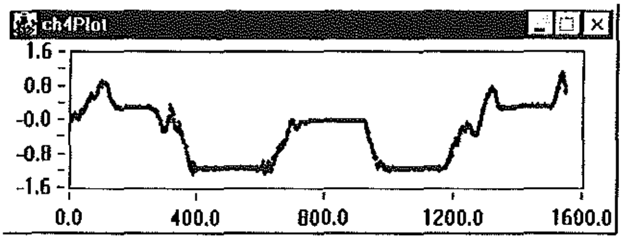


Figure 89: Part of activities recorded by channel 4.

Figure 90 illustrates a generated feature from channel 4 by using the function: "Power(Norm(ch4[$x_1..x_{16}$], 2),2)".

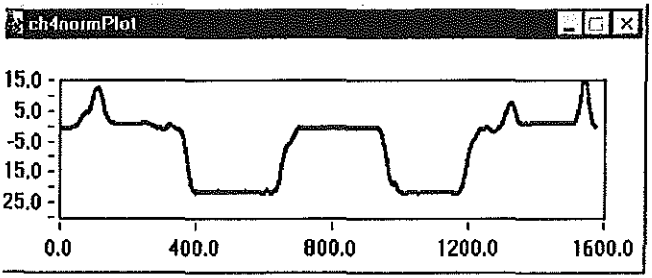


Figure 90: New generated feature from channel 4.

Figure 91 illustrates a generated feature from for channel 4 by using the function: "Standard_deviation(Norm[$x_1..x_8$])".

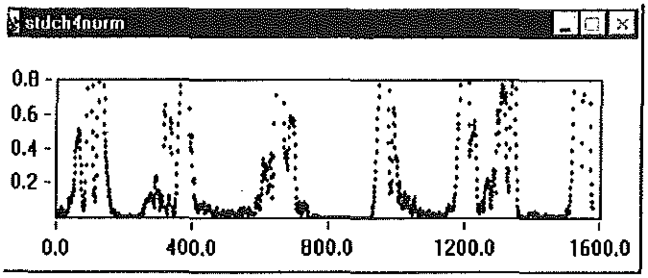


Figure 91: New generated feature from other feature.

Figure 92 illustrates another generated feature from channel 4 by using the function: "Norm(Cum_sum(ch4[$x_1..x_{16}$]),2)".

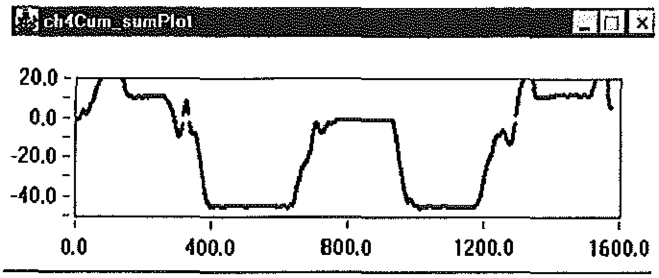


Figure 92: New generated feature from channel 4.

Figure 93 illustrates a generated feature from for channel 4 by using the function: "Standard_deviation (Cum_sum[x_1..x_8])".

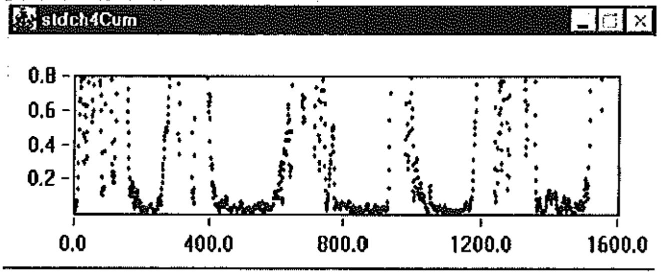


Figure 93: New generated feature from other feature.

4.7.2 Feature Selection

In this study, 160 continuous features were generated by using the two developed algorithms. From these generated features the following eight features were selected (by using a trial and error approach):

1. Standard_deviation (Cumulative_sum([x_1,....x_8])),
2. Power(Norm ([x_1,....x_{16}],2),2),
3. Norm (Cumulative_Sum([x_1,....x_{16}])),
4. Standard_deviation(Norm ([x_1,....x_8])),
5. Standard_deviation([x_1,....x_8]),
6. Average([x_1,....x_{16}]).
7. Average([x_1,....x_{16}]). $\times$ Norm(Cumulative_Sum([x_1,....x_{16}]),2)
8. Slope of Power(Norm([x_1,....x_{16}],2),2)

Still, many other subsets could be selected. In chapter 5, we introduce a neuro-fuzzy network that is capable to select the best features automatically.

4.7.3 Fuzzy rule based classifier

We choose the fuzzy rule based classifier for classification in the modified pattern recognition system. In the following subsection, we discuss some steps in building a fuzzy rule based classifier.

4.7.3.1 Generation of fuzzy sets

The first step in building a fuzzy rule based classifier is the definition of fuzzy sets which will be used in the rule. It is necessary to decompose a feature (variable) into two or more fuzzy sets. Each fuzzy set describes some range of the feature's (variable's) values and attaches a linguistic meaning to that range.

The choice of the number of fuzzy sets and how those fuzzy sets are distributed over the universe of discourse requires knowledge of how the classifier output should be related to the classifier inputs. There is no standard design procedure that can be employed to choose the number and positions of the fuzzy sets. Figure 94 illustrates how input feature Ch1Norm is decomposed into a set of fuzzy regions.

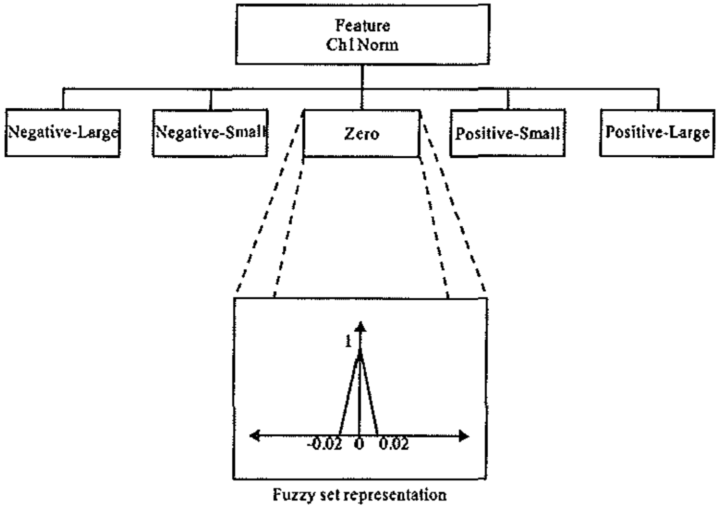


Figure 94: Five partitions for the input feature, Ch1Norm

In Figure 95 through 96, the feature Ch1Norm is decomposed (partitioned) into a collection of fuzzy sets. The universe of discourse for the feature Ch1Norm is the interval [-25, 25].

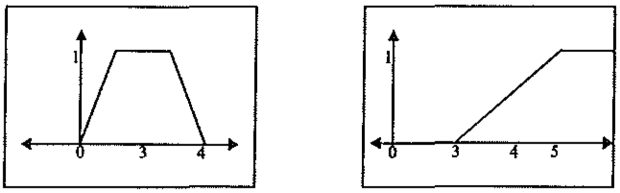


Figure 95: Membership functions for (left) fuzzy set Positive-Small and (right) fuzzy set Positive-Large.

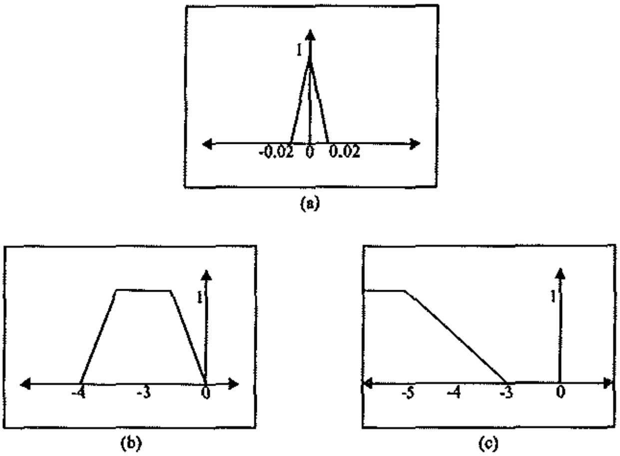


Figure 96: Membership functions for (a) fuzzy set Zero (b) fuzzy set Negative-Small and (c) fuzzy set Negative-Large.

Figure 97 shows how the input space Ch1Norm appears after partitioning.

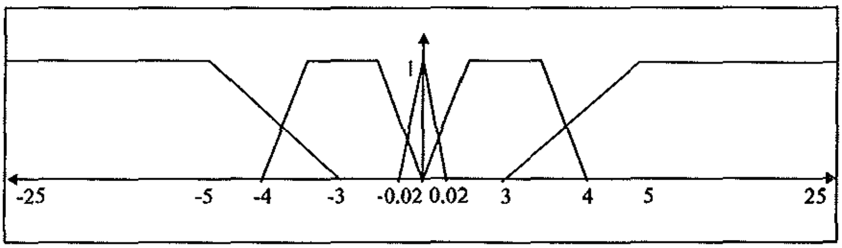


Figure 97: Partitioning for the input feature, Ch1Norm.

Other features must also be partitioned into a set of fuzzy regions. Proper partitioning of a feature into a complete set of fuzzy regions is an important aspect of building a robust and flexible classifier.

4.7.3.2 Writing the rules

The second step in building a fuzzy rule based classifier is the writing of the rules that describe how the classifier operates. If each of the n features is partitioned into a different number of fuzzy partitions, say, X_1 (feature_1's universe of discourse) is partitioned into k_1 partitions and X_2 (feature_2's universe of discourse) is partitioned into k_2 partitions and so forth, then the maximum number of rules is given by The actual used number of rules, necessary for classification is much less than N_R. The following eight

$$N_R = k_1 k_2 k_3 \cdots k_n$$

rules are typical rules which have been used to classify daily life motor activities:

RULR_1	*IF* Ch1PowerNorm is **Positive-Large** *AND* Ch1NormCum_Sum is **Positive-Large** *AND* Ch4NormCum_Sum is *NOT* **Negative-Large** *THEN* activity is <u>sitting</u>

RULR_2	*IF* Ch1PowerNorm is **Positive-Large** *AND* Ch1NormCum_Sum is **Positive-Large** *AND* Ch4NormCum_Sum is **Negative-Large** *AND* Ch4PowerNorm is **Negative-Large** *THEN* activity is <u>lying on the back side</u>

RULR_3	*IF* Ch1PowerNorm is **Zero** *AND* Ch1NormCum_Sum is **Zero** *AND* Ch3NormCum_Sum is *NOT* **Negative-Large** *AND* Ch3NormCum-Sum is *NOT* **Positive-Large** *AND* Ch2PowerNorm is **Zero** *AND* Ch2NormCum_Sum is **Zero** *THEN* activity is <u>standing</u>

RULR_4	*IF* Ch1PowerNorm is **Zero** *AND* Ch1NormCum_Sum is **Zero** *AND* Ch2PowerNorm is **Zero** *AND* Ch2NormCum_Sum is **Zero** *AND* Ch3NormCum_Sum is **Negative-Large** *AND* Ch4NormCum_Sum is **Zero** *AND* Ch4PowerNorm is **Zero** *THEN* activity is <u>lying on the left side</u>

RULR_5	*IF* Ch1PowerNorm is **Zero** *AND* Ch1NormCum_Sum is **Zero** *AND* Ch2PowerNorm is **Zero** *AND* Ch2PowerCum_Sum is **Zero** *AND* Ch3NormCum_Sum is **Positive-Large** *AND* Ch4NormCum_Sum is **Zero** *AND* Ch4PowerNorm is **Zero** *THEN* activity is <u>lying on the right side</u>

RULR_6	*IF* Ch1PowerNorm is **Zero** *AND* Ch1NormCum_Sum is **Zero** *AND* Ch2PowerNorm is **Zero** *AND* Ch2NormCum_Sum is **Zero** *AND* Ch3NormCum_Sum is **Zero** *AND* Ch3PowerNorm is **Zero** *THEN* activity is <u>standing</u>

RULR_7	*IF* Ch1Average × Ch1NormCum_Sum is **Positive-Large** *AND* h1SlopOFPowerNorm is **One** *THEN* activity is <u>walking</u>
RULR_8	*IF* Ch2Average×Ch2NormCum_Sum is **Positive-Large** *AND* h2SlopOFPowerNorm is **One** *THEN* activity is <u>walking</u>

As mentioned before, to refine the estimated onset and end time of each activity, to detect the highest peak of a step pattern, and for computing the transition time, one has to devise a good post-processing algorithm. For computing the transition time, we have used the features Ch*Standard-deviation(**) which has been shown in section 4.7.1.

4.7.4 Results

To illustrate the applicability of the fuzzy rule based classification technique for the classification of daily life motor activities, we applied our fuzzy rule based classifier to the recorded data of eight amputees and three other recorded data of healthy subjects. The fuzzy rule based classifiers were, on the average, able to recognize 99% of the presented cases of daily life activity classes of all subjects correctly. In an experiment where we applied the classifier to 12000 step patterns, the classifier was able to recognize with more than 99.5% accuracy. Figure 98 illustrates a part of the recorded data during walking activity.

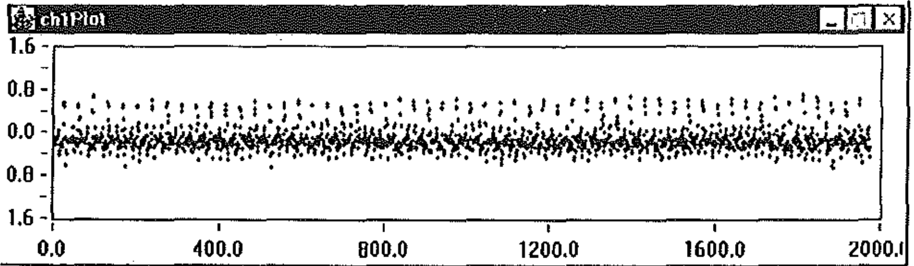


Figure 98: A part of recorded data during walking.

Figure 99 shows the output of our classifier in response to the data presented in Figure 98.

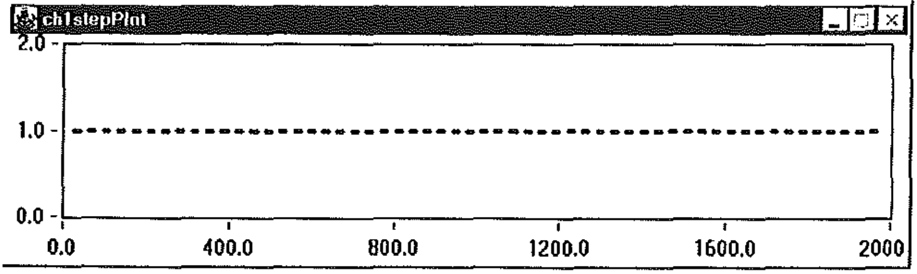


Figure 99: The output of classifier in response to above data.

In Figure 100, we present an activity profile which has been extracted from the numerical output of our classifier. It shows a 3-D bar graph of the mean footstep time, as a function of walking block interval time and monitoring time.

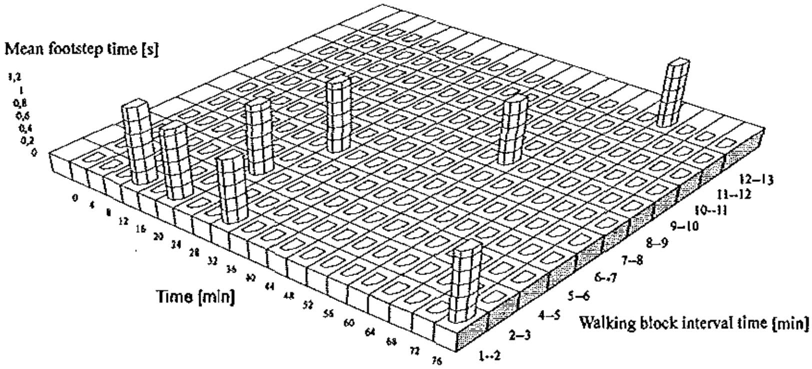


Figure 100: 3-D bar graph of an activity profile: the mean footstep time as a function of monitoring time and walking block interval time. subject 11

To be certain about patient-independency of the classifier and its performance, we did another experiment. In this extra verification experiment we applied the classifier to a set of 1.5h data (with sampling rate of 25 per second, our classifier was designed for data with sampling rate of 32 per second) which included 3830 step patterns. From this data set the classifier was able to recognize 3812 steps (99.53%). This result shows the same high performance of recognition as the finding in the first experiment.

4.8 Conclusion

In this chapter we have described a fuzzy rule based classifier and its application to the recognition of daily life motor activities. We have described with examples some of the important basic concepts in fuzzy logic. Several new features and their membership functions have been described. The comparison of the output of the fuzzy rule based classifier with the classification via visual inspection of the events resulted in 99% conformity. In contrast with the ANN based classifier, the fuzzy rule based classifier is a patient independent classifier, and the results indicate that its performance is superior to that of the PNN and BPN classifiers.

References

[1] Zadeh L.A., : Fuzzy sets, Information and Control, 1965, 8:338-353.

[2] Zimmermann H.J., : Fuzzy set theory and its application, Boston, Kluwer, Dordrecht, 1991

[3] Ross T.J., : Fuzzy logic with engineering applications, New York, McGraw-Hill, Inc. 1995.

[4] Kaufmann A, Gupta M.M., : Introduction to fuzzy arithmetic theory and application, New York, Van Nostrand Reinhold, 1991

[5] Lee C.C, : Fuzzy logic in control system : fuzzy logic controller-Part I, IEEE Transactions Syst., Man, Cybern, 1990; 20(2):404-418.

[6] Lee C.C, : Fuzzy logic in control system : fuzzy logic controller-Part II, IEEE Transactions Syst., Man, Cybern, 1990; 20(2):419-435.

[7] Tsukamoto Y., : An approach to fuzzy reasoning method, In Gupta M.M, Rammohan K.R, Yager R.R,, editors, Advances in fuzzy set theory and application, Amsterdam, North-Holland, 1979; pp. 137-149.

[8] Bezdek J., : Pattern recognition with fuzzy objective function algorithms, New York, Plenum, 1981.

[9] Pal S. and King R., : Image enhancement using fuzzy sets, Electro. Lett., 1980; 16:376-378.

[10] Bezdek J.C and Pal S.K., : Fuzzy models for pattern recognition, IEEE Press, USA, 1992.

[11] Cox E., : The fuzzy systems handbook, a practitioner's guide to building, using, and maintaining fuzzy systems., London, Academic Press, Inc, 1994.

[12] Wang L.X and Mendel J.M, : Generating fuzzy rules from numerical data, with applications., Technical report USC-SIPI-169, Signal and Image Processing Institute, University of Southern California, Los Angeles, CA, 1991.

[13] D'alché-Buc F., Anderés V. and Nadal J.P., : Rule extraction with fuzzy neural networks, International journal of neural system, 1994; 5(1):1-11.

[14] Fu L., : Rule learning by searching on adapted nets, in Proceedings of AAAI, pp. 590-595, 1991.

[15] Halgamuge S.K and Glesner M., : Neural networks in designing fuzzy systems for real world application. International journal for fuzzy sets and systems, North Holland, 1994; 65(1):1-12.

[16] Halgamuge S.K, Pöchmüller and Glesner M., : An alternative approach for generation of membership functions and fuzzy rules based on Radial and Cubic Basic Function Networks. International journal of approximate reasoning, Elsevier, 1995

Chapter 5

Hybrid System

5. Neuro-fuzzy Systems

5.1 Introduction

This hybrid system combines the advantages of both fuzzy systems and neural networks. The advantage of fuzzy systems is that they deal with explicit knowledge which can be explained and understood. Neural networks deal with implicit knowledge which can be acquired by learning. Unlike in most ANN in which knowledge is not transparent (the black-box characteristic of ANN), the knowledge in a neuro-fuzzy system is transparent like in fuzzy systems[1][2]. The architecture of a neuro-fuzzy system is such that the trained network can be translated into a number of fuzzy IF-THEN rules. The strength of a classification rule such as "IF X is small AND Y is large THEN class A" is determined by the interconnection weight which can be learned by a learning process. In addition, a number of neuro-fuzzy systems can analyze the features (inputs to the network) so that superfluous features can be removed. **FuNe-I** [3] is one such Neuro-fuzzy system which has been implemented as a mutilayer perceptron architecture, and has the following advantages:

1. Knowledge incorporation: explicit knowledge acquired from experts can be easily incorporated as new rules to the FuNe system;
2. Rules extraction: the modified and new rules can be extracted from a properly trained FuNe, to explain how the results are derived;
3. Feature selection: after the extraction of rules, superfluous input features which do not appear in rules or appear in weaker rules can be removed;
4. Generalization capability.

The main purpose of this chapter is to introduce the neuro-fuzzy system briefly and to describe its applicability in our proposed methodology for future improvement of daily life motor activity classification.

5.2 Proposed methodology

Figure 101 shows a diagram of our proposed methodology for classification of AMMA-signals.

For the features generation part, an interactive software package will have to be developed to generate a number of features from preprocessed data, as discussed in chapter 2. In addition, the software package must generate the training and test sets interactively.

In the feature selection part, the features with high discrimination ability are selected automatically by the FuNe network. Also, FuNe will generate a set of classification fuzzy rules which can be implemented in a fuzzy classifier.

Since our proposed methodology is general in nature, it can be applied in other fields of pattern recognition of one-dimensional medical signals such as ECG, EEG, EMG. We believe that this systematic approach is a solution to other related problems in industry as well.

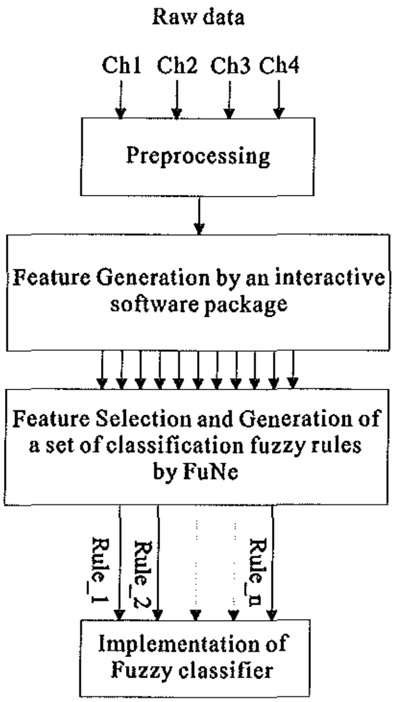


Figure 101: schematic description of the proposed methodology for classification.

5.3 Neuro-Fuzzy system

The basic idea in neuro-fuzzy systems (inference or classifier system) is to incorporate learning capability into fuzzy systems. By training such a system on training data using the back-propagation (or any other) learning algorithm, one can extract a suitable number of fuzzy rules and find a proper partitioning of input and output space (structure estimation) or one can adjust the system parameters, such as membership functions and other possible parameters (parameter estimation). Several methods for the fusion of fuzzy systems and neural networks are reported in literature [4][5][6][7][8][9][10]. Different learning strategies are used in these systems, e.g., unsupervised learning, supervised learning and differential competitive learning. In order to familiarize with neuro-fuzzy systems, in the following paragraphs, we consider the Tsukamoto fuzzy inference system which is implemented as a neuro-fuzzy system.

For simplicity, we assume that the fuzzy inference system under consideration has two inputs x and y and only one output z. For a Tsukamoto fuzzy model [11], a typical rule set with two fuzzy IF-THEN rules is:

Rule 1: IF x is A_1 and y is B_1 THEN z is C_1

Rule 2: IF x is A_2 and y is B_2 THEN z is C_2

Figure 102 is an illustration of how a two-rule inference system of the Tsukamoto type derives the overall output when subjected to two crisp inputs x and y.

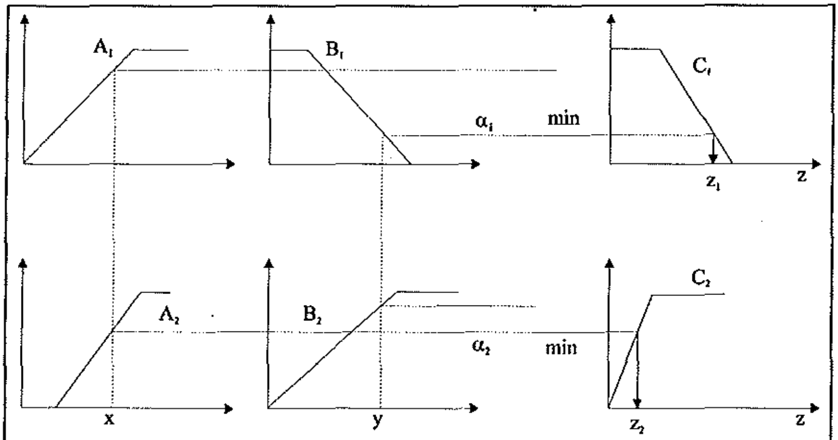


Figure 102: The Tsukamoto fuzzy model.

$$z = \frac{\alpha_1 z_1 + \alpha_2 z_2}{\alpha_1 + \alpha_2}$$

The equivalent Neural network based architecture(Neuro-fuzzy architecture) of this model is shown Figure 103.

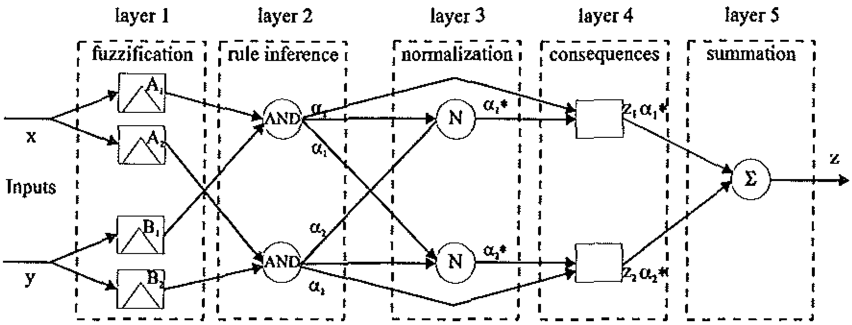


Figure 103: An equivalent neural network based fuzzy inference system for Tsukamoto fuzzy model.

Looking at Figure 103, the operation of a neuro-fuzzy system can be described as follows:

Layer 1. Fuzzification layer

Each node (neuron) in this layer represents an input membership function of the antecedent of a fuzzy rule. The crisp inputs x and y are fuzzified by using these membership functions. There is no restriction to the form of the membership functions (except that they must be differentiable with respect to their parameters). Usually, trapezoid, triangular, or bell-shaped membership functions are used. For example, the bell-shaped function (Gaussian function) is defined as follows:

$$mf_j(x_i) = e^{-\frac{(x_i-\mu_j)^2}{\sigma_j^2}}$$

The parameters of the neurons of this layer can be trained to fine tune the final shape and location of the membership functions. In the case of the Gaussian membership function, the parameters μ_j and σ_j may be interpreted as the weights of a link between the i.th neuron in layer 1 and the j.th neuron in layer 2. In most neuro-fuzzy architectures, the number of neurons in this layer is fixed, but it is possible to add or remove these neurons during training, according to the outputs produced on the training samples.

Layer 2. Fuzzy rule layer

Each rule neuron performs the fuzzy logical AND operation between antecedents (IF-part). This layer contains one neuron for each fuzzy IF-THEN rule. Each neuron corresponding to the antecedent of a fuzzy rule computes the firing level α_i of this antecedent. In our example case the firing levels α_i of the fuzzy rules are computed by

$$\alpha_1 = \min(mf_{A_1}(x), mf_{B_1}(y))$$
$$\alpha_2 = \min(mf_{A_2}(x), mf_{B_2}(y))$$

Layer 3. Normalization layer

The firing levels of the fuzzy rules are normalized. The i.th neuron compute the ratio of the i.th rule's firing level and the sum of all rule's firing levels:

$$\alpha_1^* = \frac{\alpha_1}{\alpha_1 + \alpha_2}$$
$$\alpha_2^* = \frac{\alpha_2}{\alpha_1 + \alpha_2}$$

Layer 4. Consequence layer

The function of this layer is rule evaluation. Each neuron in this layer represents a consequent proposition "THEN z is C_i"; it contains the membership function representing the output variable. Each neuron derives the rule output z_i from the equation $C_i(z_i) = \alpha_i$ and multiplies its value with the corresponding normalized firing level. In our example case:

$C_1(z_1)= \alpha_1$ the output of rule 1 is $\alpha_1^* \cdot z_1$

$C_2(z_2)= \alpha_2$ the output of rule 1 is $\alpha_2^* \cdot z_2$

Layer 5. Summation layer

This layer calculates the overall output as the summation of all incoming signals:

$$z = \alpha_1^* \cdot z_1 + \alpha_2^* \cdot z_2 = \frac{z_1 \cdot \alpha_1 + z_2 \cdot \alpha_2}{\alpha_1 + \alpha_2}$$

Training of Tsukamoto fuzzy inference system

First the parameters of the membership functions are initialized. After that, the fuzzy rules are updated by using a training algorithm such as back-propagation as follows:

Given are k training samples arranged in the training set:

$\{ ([x^1, y^1], z^1), ([x^2, y^2], z^2), \ldots, ([x^k, y^k], z^k) \}$

1. Present an input data sample, and compute the corresponding output
2. Calculate the error between the desired output and the actual output(The error is defined by a cost function)
3. The membership functions are updated
4. If Error > Tolerance then goto step 1 else stop.

The shape and position of the membership functions in the fuzzification and consequence layers can be fine tuned by adjusting the parameters of the neurons in these layers, during the training process.

Table 10 Shows some learning schemes used in several currently proposed neuro-fuzzy inference systems.

Neuro-fuzzy system	Premise learning	Consequent learning
Kosko [12]	AVQ	AVQ
Berenji [8]	gradient descent	gradient descent
Lin [13]	SOM	gradient descent
Horikawa [7]	gradient descent	gradient descent
Nie [4]	modified SOM	gradient descent

Table 10: Learning of neuro-fuzzy inference systems

Their training algorithms differ very much from each other and no comparison of those has been presented in literature.

5.3.1 FuNe-I neuro-fuzzy architecture

A special multilayer perceptron architecture is successfully used for generating the fuzzy-neural system FuNe-I. This special architecture, trained with supervised learning can be used to generate a fuzzy classifier system from a given representative input/output data set (training set) without expert help. The system (FuNe-I) extracts an untuned knowledge base in the first phase. The extracted fuzzy system is tuned in the second phase. Figure 104 shows the structure of FuNe-I in the first phase. Only the connections in the fuzzification and defuzzification blocks in Figure 104 represent variable weights, other connections have fixed unity weights. The dark outlined circles represent neurons with sigmoid transfer functions. In FuNe-I both IF and IF NOT rules are considered.

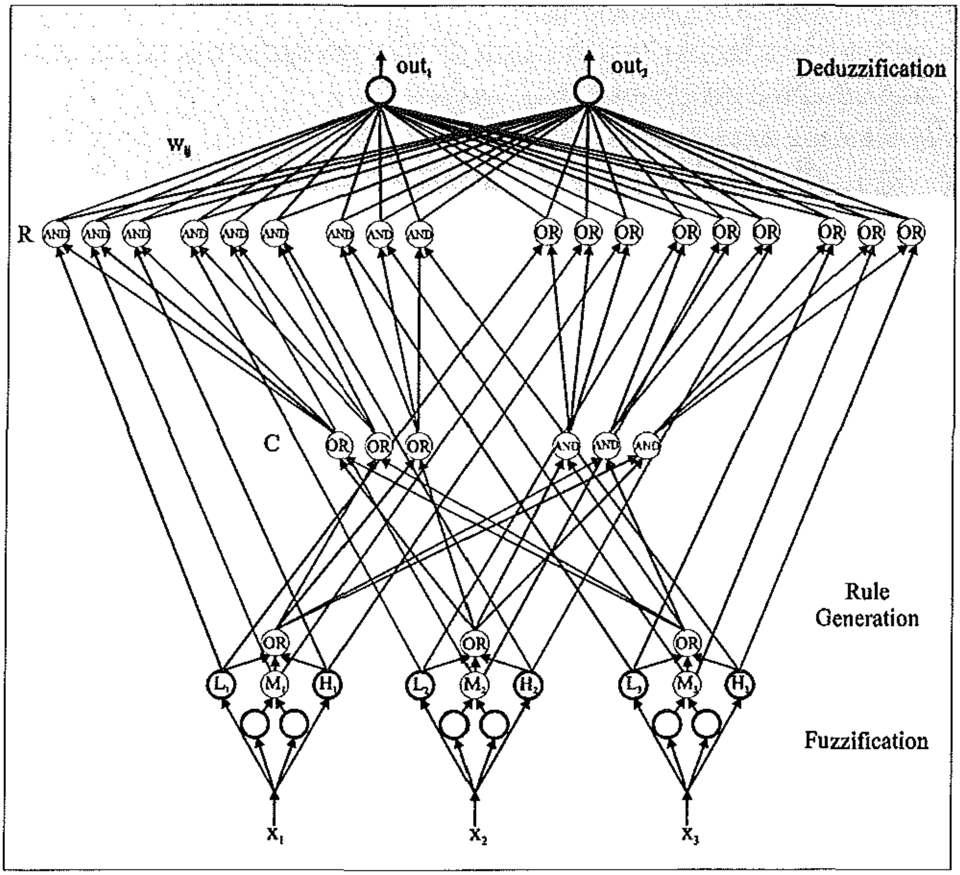


Figure 104: The structure of FuNe-I in first phase.

The FuNe-I model differs from the conventional fuzzy model in that it transfers the weighted sum of the fired rules into the crisp output as follows:

$$out_i = Sig(\sum_{j=1}^{r} w_{ij} \cdot \alpha_j)$$

where r is the number of rules, w_{ij} represents the weight of the connection from the j.th rule neuron to the i.th output. Sig (sigmoid function) is the transfer function of the output neuron.

FuNe-I employs a different approach for finding the initial rule base. In the first phase, it identifies rule relevant neurons for conjunctive and disjunctive rules for each output. The network is trained with "frozen" membership functions; the membership functions are not adjusted during training.

Let us consider an example with 3 inputs, where each of them is partitioned into 3 fuzzy regions with the fuzzy sets of Low (L), Medium (M) and High (H). To find whether the i.th input has any influence on a conjunctive rule, the next steps are to be taken:

1. Connect the fuzzification layer to the neuron C_i (layer C in Figure 104), that selects the maximum from the strongest membership values from all the inputs but the input i :

   $$C_i = Max_{[j \neq i]}^{\forall j}[Max_j(L_j, M_j, H_j)]$$

2. Connect the neuron C_i to corresponding neurons , R_{L_i}, R_{M_i} and R_{H_i} (layer R in Figure 104)
3. connect the neurons R_{L_i}, R_{M_i} and R_{H_i} to the corresponding neurons out_i and initialize the weights W_{L_i}, W_{M_i} and W_{H_i}
4. After the training process, connecting weights are analyzed to extract the Min-rule relevant neurons

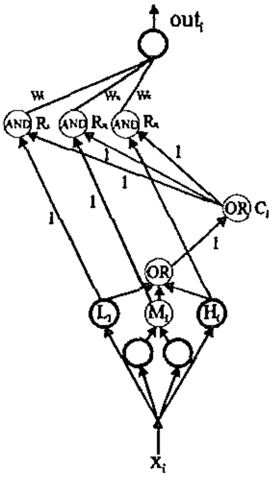


Figure 105: The i.th conjunctive rule.

Extraction of the Max-rule relevant neurons is performed in analogy to the above steps, but C_i is computed as follows:

$$C_i = Min_{[j \neq i]}^{\forall j}[Max_j(L_j, M_j, H_j)]$$

All extracted Min and Max rule relevant nodes can be considered as the initial rule base for FuNe-I-FS. The modified fuzzy system generated from FuNe-I training with a gradient descent learning algorithm (e.g. backpropagation algorithm) is called a FuNe-I fuzzy system (FuNe-I-FS). Figure 106 shows a typical Fuzzy system which has been extracted from a trained FuNe-I model.

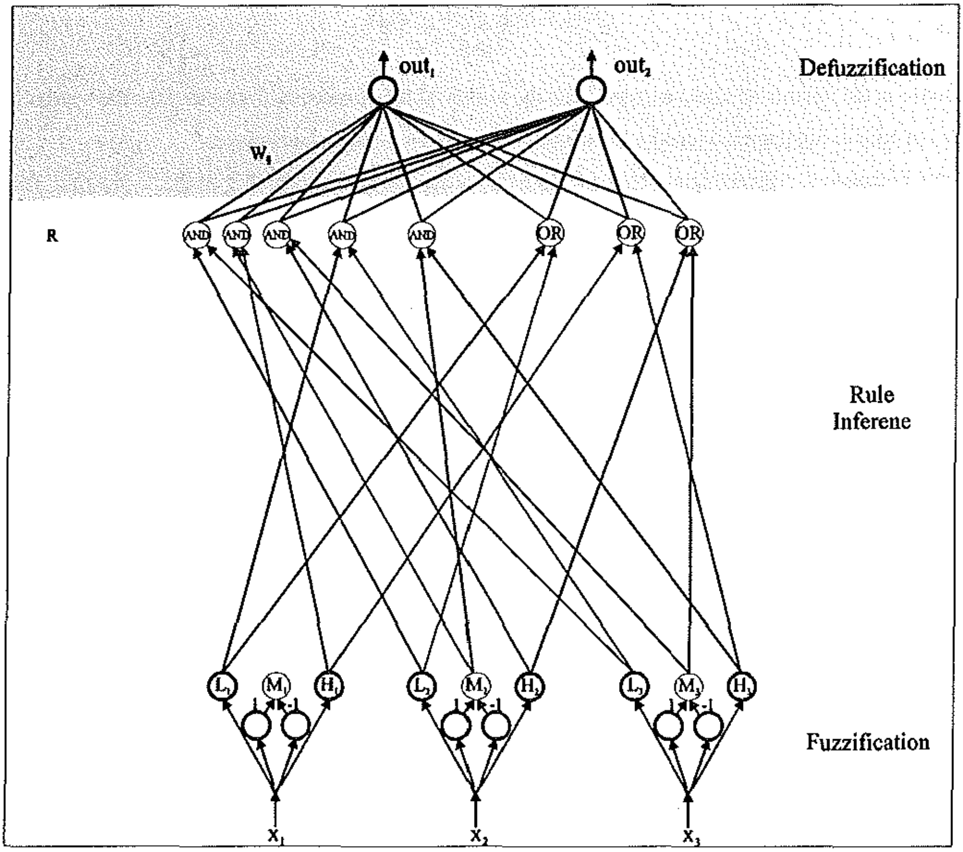


Figure 106: FuNe I Fuzzy System.

After generating the FuNe-I-FS, i.e., after the creation of an initial knowledge base is accomplished, the optimization can be started by using the same training data set. The initial antecedent membership functions can be tuned. An already extracted rule base by FuNe-I can be reduced effectively by training the FuNe-I-FS with the training data set. This is done by the analysis of the connecting weights. Also, superfluous inputs (features) which do not appear in rules can be removed.

5.4 Application Example

Figure 107 shows an example of a multichannel recorded data set. Every channel contains many different patterns (waveforms), and each event class is defined by combining the patterns from different channels. Here, we have four event classes, A, B, C and D.

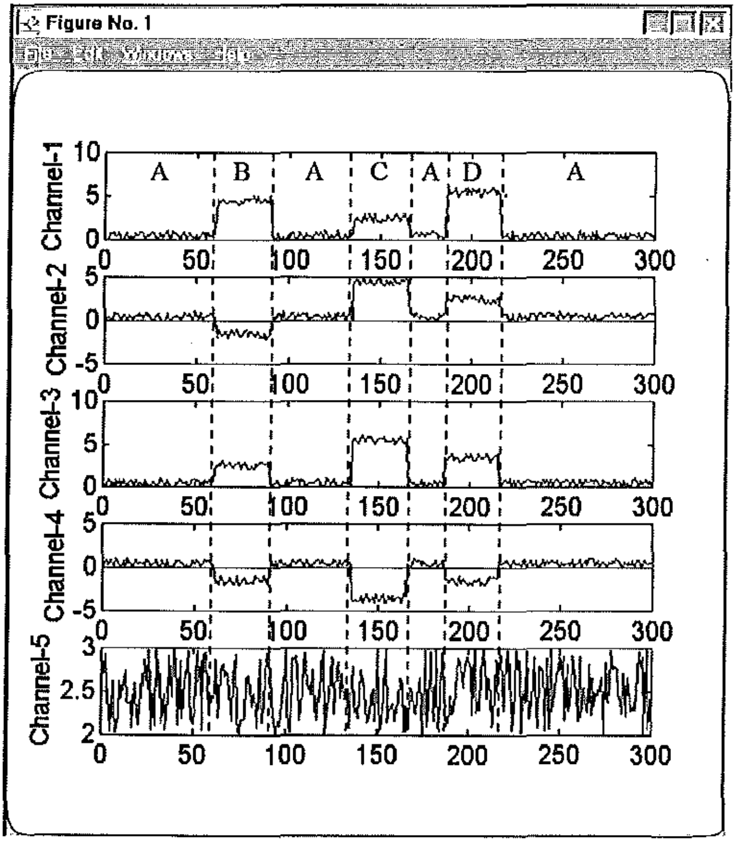


Figure 107: An example of a multichannel recorded data.

In order to apply FuNe-I to this example, firstly, we built a training set of 60 examples, 15 examples for each event class as follows:

CH1	CH2	CH3	CH4	CH5	CLASS LABEL				
0.2190	0.0043	0.2303	0.2724	2.6145	1.0	0.0	0.0	0.0	For class A
0.9347	0.9033	0.6515	0.4770	2.0397	1.0	0.0	0.0	0.0	For class A
4.5285	-1.8775	2.5869	-1.8300	2.4308	0.0	1.0	0.0	0.0	For class B
4.5881	-1.8117	2.8119	-1.9268	2.1986	0.0	1.0	0.0	0.0	For class B
2.2478	4.4416	5.3823	-3.6263	2.4037	0.0	0.0	1.0	0.0	For class C
2.3893	4.7719	5.4295	-3.7381	2.2648	0.0	0.0	1.0	0.0	For class C
5.1504	2.2321	3.4156	-1.2061	2.7853	0.0	0.0	0.0	1.0	For class D
5.3438	2.4306	3.5146	-1.6682	2.1015	0.0	0.0	0.0	1.0	For class D

The training of FuNe-I classifier was started with the following setting :

- Number of inputs: 5
- Number of outputs: 4
- Number of rules to extract: 12
- Number of adjectives (linguistic terms) per input: 3
- Momentum: 0.600
- Learning rate: 0.400

After the first phase (training of FuNe-I), the FuNe-I generated a fuzzy system (FuNe-I-FS) with 12 rules which is shown in Figure 108 (the screen output of FuNe-I program).

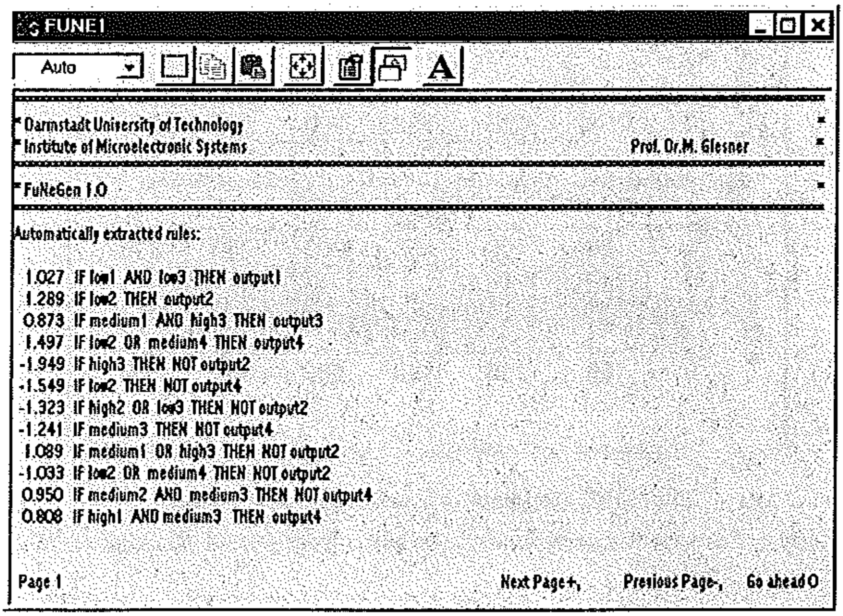


Figure 108: The extracted rules from the trained FuNe I.

As can be seen, input 5 (channel-5) did not appear in rules and can be removed. For the second phase, we used the same training data set to train the FuNe-I-FS (generated fuzzy classifier system by FuNe-I), in order to optimize the generated rules and fine tune the initial membership functions. Figure 109 shows the membership functions of input-1(CH1) before and after fine tuning.

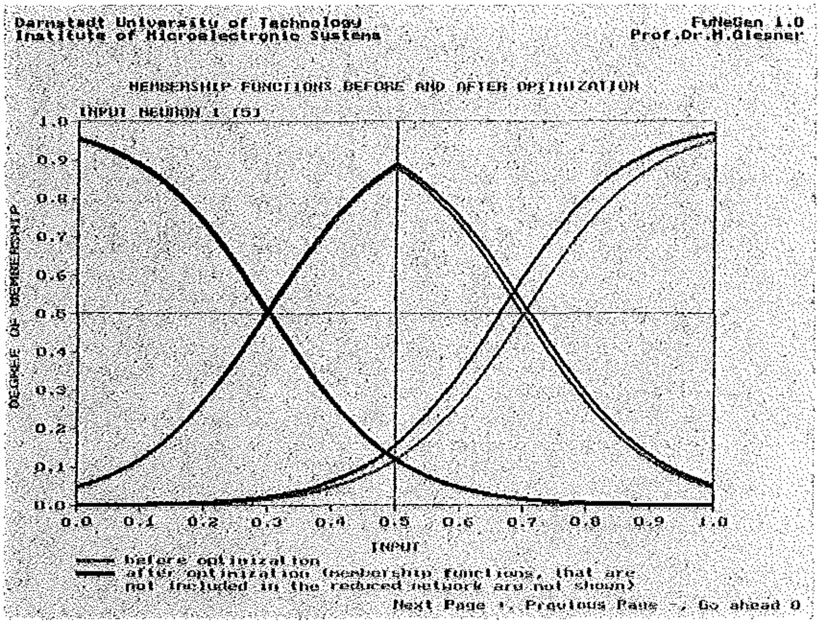


Figure 109: Initial and fine tuned membership functions of input-1.

Figure 110 shows the membership functions of input-2(CH2) before and after fine tuning.

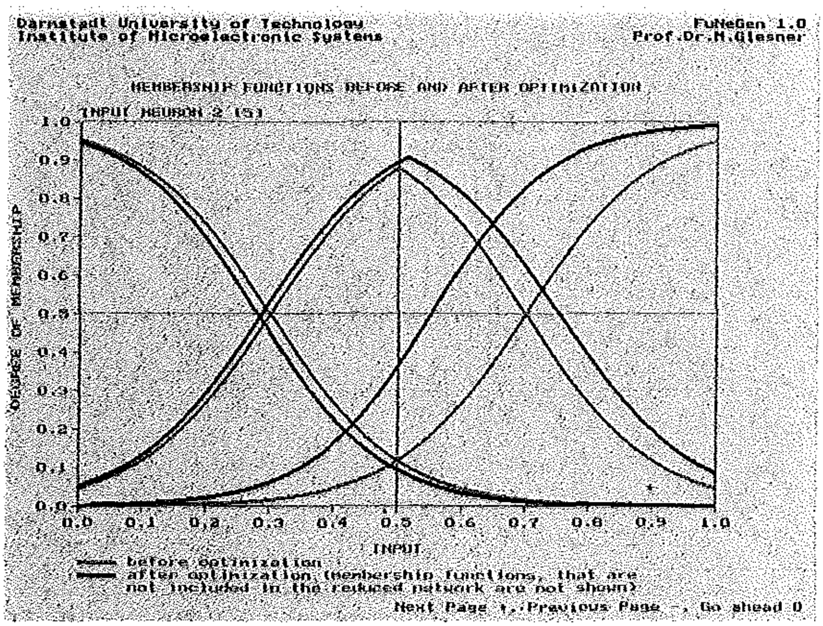


Figure 110: Initial and fine tuned membership functions of input-2.

Figure 111 shows the membership functions of input-3(CH3) before and after fine tuning.

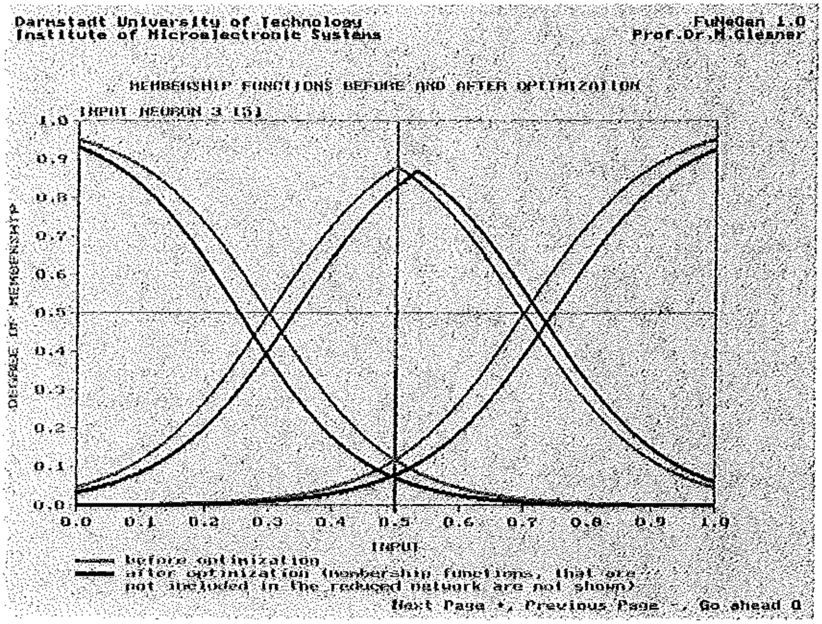


Figure 111: Initial and fine tuned membership functions of input-3.

Figure 112 shows the membership functions of input-4(CH4) before and after fine tuning.

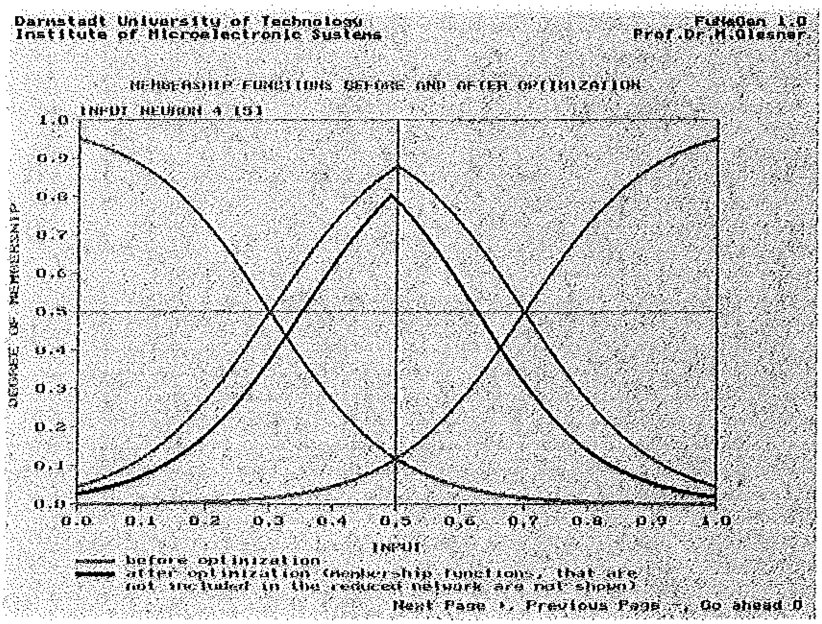


Figure 112: Initial and fine tuned membership functions of input-4.

Error! Reference source not found. shows the updated and optimized find rule set by FuNe-I-FS.

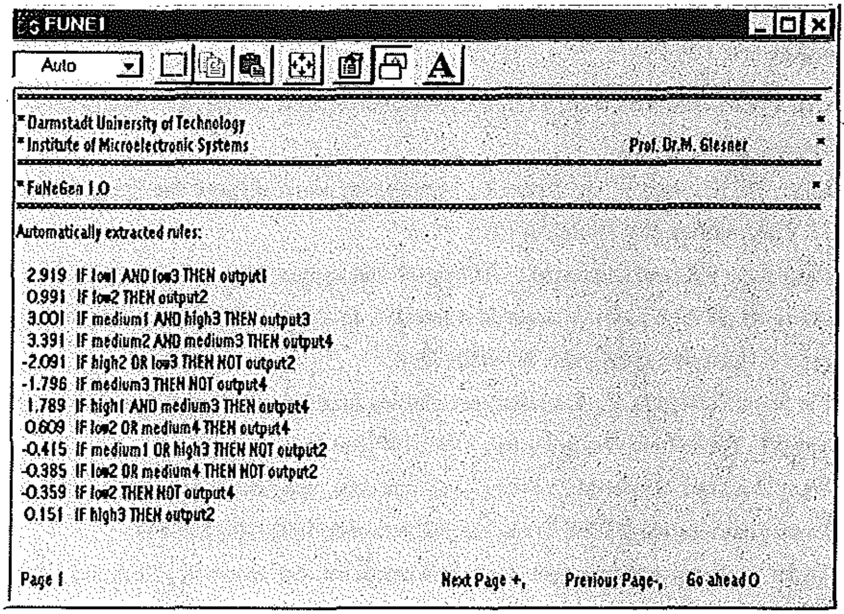


Figure 113: The updated and optimized rules by FuNe-I-FS.

To evaluate the updated and optimized rules by FuNe-I-FS, we built a test file with 300 examples pattern, and applied optimized FuNe-I-FS to it. The result shows 100% correct classification.

The current version of FuNe-I has several limitations which make it unrealistic for a real application. These can be listed as follows:

- The maximal number of inputs is 7.
- The maximal number of linguistic terms is 3.

5.5 Conclusions

In this chapter, we briefly presented an introduction to neuro-fuzzy systems which play a big role in our proposed methodology for the recognition of patterns in multichannel recorded data. Several methods for fusion of fuzzy systems and neural networks are reported in literature. Some of them can only fine tune the membership functions, some others can generate fuzzy rules from a training data set without expert help. We described the FuNe-I model which can generate a fuzzy classifier and remove superfluous input features. Applying FuNe-I to an artificial multichannel data example, has shown that Neuro-fuzzy classifiers like FuNe-I can be potentially useful in classification. The author believes that

Neuro-Fuzzy systems will eventually replace conventional fuzzy decision systems and neural networks in variety of applications.

References

[1] Gupta M.M. and Rao D.H., : On the principles of fuzzy neural networks., Fuzzy Sets and System, 1994; 16:1-18.

[2] Nauck D., Klawonn F., and Kruse R., : Combining neural network and fuzzy control., FLAI' 93, Linz, Austria, Jun. 28- Jul. 2, 1990.

[3] Halgamuge S.K. and Glesner M., : A fuzzy-Neural approach for pattern classification with the generation of rules based on supervised learning. In Neuro- Nimes 92, pp 165-173, Nanterre, France, November 1992. ISBN 2-906899-79-8.

[4] Nie J. and Linkens D.A., : Learning control using fuzzified self-organizing radial basis function network., IEEE Trans. Fuzzy System, 1993; 1(4):280-287.

[5] Wang L.X. and Mendel M., : Fuzzy basis functions, universal approximation and orthogonal least-squares learning., IEEE Trans. Neural Networks, 1992; 3(5):807-814.

[6] Thaler S. : Fuzzy rule generation based on a neural network approach., Electronic Engineer, pp. 43-50, July, 1993.

[7] Horikawa S., Furahashi T., and Uchikawa Y., : On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm., IEEE Trans. Neural Networks, 1992; 3:801-806.

[8] Berenji H.R. and Khedkar P., : Learning and tuning fuzzy logic controllers through reinforcements., IEEE Trans. Neural Networks, 1993; 3(5):724-740.

[9] Vuorimma P., : Fuzzy self-organizing map., Fuzzy Sets and Systems, 1994; 66:223-231.

[10] Kwon T.M. and Zervakis M.E, : A self-organizing KNN fuzzy controller and its neural networks structure., Int. J. Adaptive Control and Signal Processing, 1994; 8:407-431.

[11] Tsukamoto Y., : An approach to fuzzy reasoning method, In Gupta M.M, Rammohan K.R, Yager R.R,, editors, Advances in fuzzy set theory and application, Amsterdam, North-Holland, 1979; pp. 137-149.

[12] Kosko B., : Neural networks and Fuzzy systems: A dynamical approach to machine intelligence, Englewood Cliffs, NJ: Prentice Hall, 1992

[13] Lin C.T., and Lee C.S.G., : Neural-Network-based fuzzy logic control and decision system., IEEE Trans. Computers, 1991; 40(12):1320-1336.

Chapter 6

Summary and future directions

6. Summary and future directions

6.1 Summary

This thesis deals with a study on the potential usefulness of artificial neural networks, fuzzy rule based systems and neural fuzzy systems for recognizing daily life motor activities from the multichannel recorded data in ambulatory monitoring. This thesis can be summarized as follows:

Chapter 1

In chapter 1, a general introduction about the background of the study is given. The different methods of monitoring of daily life motor activities are discussed. The structure of the system for ambulatory monitoring of motor activities is described. Some suggested numerical and graphical representations of results data are presented. The detected relevant motor activities and related clinical parameters by our classifiers are highlighted.

Chapter 2

Chapter 2 briefly reviews various pattern recognition techniques that can be used to perform pattern recognition tasks in AMMA-signals. The manner that the feature space can be partitioned by these various techniques are described. After that, various types of daily life motor activities were defined on the basis of the output of four accelerometers. Further, this chapter addresses a new developed method for generating features. This method is critical for solving patient independent automated pattern recognition systems for the AMMA-system.

Chapter 3

Chapter 3 begins with a short introduction to the field of neural networks, followed by a history of neural networks. Examples of learning types and neural network topologies are discussed. Special attention is paid to the two types of neural networks, Probabilistic Neural Networks (PNN) and BackPropagation Networks (BPN) that were applied to AMMA-signal. The topology and learning rule of these two neural networks are addressed. Construction of a training set is discussed. Further, the way of implementing PNN and BPN based classifiers for recognition of daily life motor activities and their performance estimations are discussed. Finally, this chapter presents the results of application of two artificial neural network based classifiers to eight recorded data bases which were obtained from monitoring of eight amputees during their daily life. From this Chapter we derived the following conclusions:

- daily life motor activities are complex and show extremely large inter- and intra-individual variation which excludes using regular signal processing tools for recognition;
- for the application under consideration, PNNs and BPNs are potentially useful options;
- a satisfactory conformity of 95% between automatic and visual classification of events can be achieved;
- the automatic classification of 10 hours of activities takes less computation time with special hardware;
- the graphical presentation of the output yields clinically meaningful information;
- the ADL-classifier is patient-dependent which means that for every patient a training set has to be built and optimal parameters settings have to be chosen.
- Application of a postprocessing algorithm does improve the determination of the onset and end time of each activity.
- Because of the many setting parameters and the time consuming training process of BPNs, using a PNN-classifier is much easier than using a BPN-classifier.
- Because of the reject class option on PNN output, the performance of PNN is more reliable than that of a BPN.

Chapter 4

Chapter 4 introduces the basic concept of fuzzy sets and membership functions. Various fuzzy operators are illustrated. This chapter introduces the fuzzy IF-THEN rules format and discusses the fuzzy inference systems, and presents two well-known inference mechanisms in fuzzy rule-based systems: Mamdani and Tsukamoto's inference mechanism. This chapter also addresses the fuzzy rule-based classification. Finally, the application of a fuzzy rule based system to daily life motor activities is introduced. To illustrate the applicability of the fuzzy rule based classification technique for the classification of daily life motor activities, we applied our fuzzy rule based classifier to the recorded data of eight amputees and three other recorded data of healthy subjects. The fuzzy rule based classifiers were, on the average, able to recognize 99% of the presented cases of daily life activity classes of all subjects correctly. In an experiment where we applied the classifier to 12000 step patterns, the classifier was able to recognize with more than 99.5% accuracy, which is verified by visual inspection. To be certain about patient-independency of the classifier and its performance, we did another experiment. In this extra verification experiment we applied the classifier to a set of 1.5h data (with sampling rate of 25 per second, our classifier was designed for data with sampling rate of 32 per second) which included 3830 step patterns. From this data set the classifier was able to recognize 3812 steps (99.53%). This result shows the same high performance of recognition as the finding in the first experiment.

Chapter 5

Chapter 5 presents an introduction in the field of Neural-fuzzy systems. This chapter also presents our proposed methodology for recognition of patterns in multichannel recorded data. Further, chapter addresses the **FuNe** which is one of Neural-Fuzzy system in literature. Finally, the capability of **FuNe** to remove superfluous features and extract fuzzy rule is illustrated by an artificial multichannel recorded data example.

6.2 Future Directions

A further improvement of the fuzzy rule based classifier for the recognition of daily life motor activities is practically impossible. A very high level of accuracy is already achieved and errors occur only in very special cases. But, for the recognition of some classes of activities, we expect that it is not possible to find a high discriminating feature. The combination of a fuzzy rule based classifier and the PNN based classifier which operates on the raw data seems to be the only solution. Developing a software package that provides such combination is a future work.

The author believes that his suggested methodology for the recognition of patterns in multichannel recorded data is a systematic approach. In order to implement that, the following should be done:

- developing an interactive software package that generates a number of features from preprocessed data, and has capability for building the training and test sets.
- removing the limitation regarding the number of inputs and membership functions for each input feature in the FuNe-I system.
- using methods for the clustering of input space for initialization of the membership functions in FuNe-I system.

6.3 Samenvatting

Dit proefschrift behandelt een onderzoek naar de potentiële bruikbaarheid van artificiële neurale netwerken, fuzzy rule based systemen en neurale fuzzy systemen voor het herkennen van dagelijkse motorische activiteiten van de multi-kanale geregistreerde gegevens. Dit proefschrift kan als volgt worden samengevat:

Hoofdstuk 1

In hoofdstuk 1 wordt een algemene introductie gegeven over de achtergrond van de studie. De verschillende methodes van het registreren van dagelijkse motorische activiteiten worden beschreven. De structuur van het systeem van ambulante registratie wordt beschreven. Een aantal numerieke en grafische representaties van de resultaten worden gepresenteerd. De herkende relevante motorische activiteiten en gerelateerde klinische parameters door onze herkenningsystemen worden benadrukt.

Hoofdstuk 2

In hoofdstuk 2 worden de verschillende patroonherkenning technieken behandeld die kunnen worden gebruikt om AMMA-signalen te klassificeren. Daarnaast word er de verschillende technieken beschreven waarmee de kenmerkruimte verdeeld word. Op basis van de uitgangen van vier versnellingsopnemers worden diverse typen van dagelijkse motorische activiteiten gedefinieerd. Vervolgens, gaat dit hoofdstuk in op een nieuw ontwikkelde methode om nieuwe kenmerken te genereren. Deze methode is cruciaal voor het oplossen van patiënt onafhankelijke geautomatiseerde patroon herkenningsystemen voor het AMMA-systeem.

Hoofdstuk 3

Hoofdstuk 3 begint met een korte introductie over neurale netwerken, gevolgd door een geschiedenis ervan. Diverse voorbeelden van leermethoden en neurale netwerk topologiën worden besproken. Speciale aandacht wordt besteed aan twee typen van neurale netwerken, Probabilistic Neural Network (PNN) en BackPropagation Network (BPN) die toegepast werden op het AMMA-signaal. De structuur en leer regel van deze twee neurale netwerken worden geadresseerd en constructie van een trainings set wordt bediscussieerd. Vervolgens, worden de manieren van implementatie van op PNN en BPN gebaseerde herkenningsystemen voor het herkennen van dagelijkse motorische activiteiten en hun "performance estimations" gediscussieerd. Tenslotte, presenteert dit hoofdstuk de resultaten van toepassingen van twee op kunstmatige neurale netwerken gebaseerde herkenningsystemen aan acht geregistreerde data-bases die waren verkregen door het registreren van acht personen met been amputaties tijdens hun dagelijks leven. Uit dit hoofdstuk trekken we de volgende conclusies:

- dagelijkse motorische activiteiten zijn complex en laten extreem grote inter- en intra-individuele variaties zien. Daardoor het gebruik van reguliere signaal bewerkings technieken voor het herkenen van activiteiten zijn uitgesloten.
- voor de bedoelde toepassingen, zijn PNNs en BPNs nuttige opties.
- een bevredigende overeenkomst van 95% tussen automatische en visuele classificatie van gebeurtenissen kunnen worden bereikt.
- de automatische classificatie van activiteiten gedurende 10 uur neemt minder tijd in beslag met speciale hardware.
- een grafische presentatie van de uitkomsten levert relevante klinische informatie op.
- het neurale netwerk gebaseerde herkenningsysteem is patiënt afhankelijk. Dit betekent dat voor iedere patiënt een trainings set moet worden opgebouwd en opnieuw de optimale parameter instellingen moeten worden gekozen.
- het uitvoeren van "postprocessing" algoritme geeft een verbetering van het bepalen van de begin- en eind-tijd van iedere activiteit.
- vanwege het aantal parameters en het tijd consumerende "training process" van BPNs, is het gebruik van een PNN-herkennings systeem veel makkelijker dan het gebruik van een BPN-herkennings systeem.
- vanwege de "reject class" optie bij PNN uitgang, is de PNN prestatie meer betrouwbaar dan die van de BPN.

Hoofdstuk 4

In hoofdstuk 4 wordt een algemene introductie gegeven over de basis concepten van vage verzameling en lidmaatschap functies. Dit hoofdstuk beschrijft de fuzzy "IF-THEN" regels. Er wordt een beschrijving gegeven van de "fuzzy inference" systeem. Daarnaast word er twee algemeen bekende inference mechanismes in "fuzzy rule-based" systemen gepresenteerd namelijk het Mamdani en het Tsukamoto inference mechanisme. Dit hoofdstuk adresseert ook de op "fuzzy rule" gebaseerde classificatie. Tenslotte, wordt de toepassing van "fuzzy rule" systeem op dagelijkse motorische activiteiten geïntroduceerd. Om de toepasbaarheid van dergelijke classificatie techniek voor de te illustreren, werd zo'n klassificatie toegepast op de geregistreerde data van acht patiënten met been amputaties en drie gezonde personen. Het fuzzy rule herkenningsystemen kon gemiddeld 99% van de gepresenteerde gevallen van dagelijkse motorische activiteit klassen correct herkennen. In een experiment waar het herkenningsysteem op 12000 stappen patroon werd toegepast, kon het systeem die patronen meer dan 99.5% gevallen nauwkeurigheid herkennen. Om zeker te zijn van patiënt onafhankelijkheid van het systeem en zijn prestaties, werd er een ander experiment uitgevoerd. In dit extra verificatie experiment, werd het systeem toegepast op een geregistreerd data van 1.5 uur (met sampling rate van 25 per seconde, waarbij het systeem was ontworpen voor data met sampling rate van 32 per seconde). Uit 3830 stappen patroon kon het systeem 3812 gevallen herkennen (99.53%), hetgeen werd geverifieerd langs de weg van visuele inspectie. Dit hoge herkennings prestatie komt overeen met de bevindingen uit het eerste experiment.

Hoofdstuk 5

Hoofdstuk 5 beschrijft het veld van "Neural-fuzzy" systemen. Dit hoofdstuk presenteert ook onze gesuggereerde methodologie voor het herkennen van patronen in multi-kanaal geregistreerde data. Vervolgens, adresseert dit hoofdstuk de **FuNe**, één van de "Neural-fuzzy" systemen in de literatuur. Tenslotte, wordt het vermogen van **FuNe** om overbodige kenmerken te verwijderen en "fuzzy rule" te onttrekken, geïllustreerd door het toepassen van dit systeem op een voorbeeld van kunstmatige multi-kanaal geregistreerde data.

6.4 Toekomstige tendensen

Een verdere verbetering van het op "fuzzy rule" gebaseerde herkenningsysteem voor het herkennen van dagelijkse motorische activiteiten is praktisch onmogelijk. Een zeer hoog niveau van nauwkeurigheid is reeds bereikt en fouten komen slechts voor uitzonderlijke gevallen. Voor het herkennen van nieuwe activiteiten kan het erg moeilijk zijn om een hoge discriminerende kenmerken te vinden. De combinatie van het op "fuzzy rule" gebaseerde herkenningsysteem met het PNN systeem lijkt de enige oplossing.

Het ontwikkelen van een software tool dat deze combinatie realiseert is een toekomstig werk.

The auteur gelooft dat zijn gesuggereerde methodologie voor het herkennen van patronen in multi-kanaal geregistreerde data is een systematische benadering. Om dit te implementeren, moet het volgende worden gedaan:

- het ontwikkelen van een interactief software pakket dat een aantal nieuwe kenmerken uit voorbewerkte data genereert, en dat capabel is om de training en test verzamelingen te construeren.
- het verwijderen van de beperkingen betreffende het aantal ingangen en lidmaatschap functies voor ieder ingangs kenmerk in het **FuNe-I** systeem.
- het gebruik maken van methoden voor het clusteren van ingangs ruimte voor het initialiseren van de lidmaatschap functies in het **FuNe-I** systeem.

Glossary

WHO	World Health Organization.
ICIDH	International Classification of Impairment, Diseases and Handicaps.
CAMARC	Computer aided Movement Analysis in a Rehabilitation Context.
EMG	ElectroMyoGraphy is the detection and recording of muscle activity potential using either surface electrodes on the skin or by inserting a needle electrode into the muscle.
ECG	ElectroCardioGraphy is used to measure the heart rate kinetics as a cheaper and more practical alternative to direct oxygen consumption measurement.
Handicap	A Handicap is a disadvantage for a given individual, resulting from an impairment or disability, that limits or prevents the fulfillment of a role that is normal (depending on age, sex, and social and cultural factors) for that individual.
Impairment	An Impairment is any loss or abnormality of psychological, or anatomical structure or function.
Disability	A disability is characterized by excesses or deficiencies of customarily expected activity performance and behavior, and these may be temporary or permanent, reversible or irreversible, and progressive or regressive. Disabilities may arise as a direct consequence of impairment or as a response by the individual, particularly psychologically, to a physical, sensory, or other impairment. Disability represents objectification of an impairment, and as such it reflects disturbances at the level of the person.

Acknowledgment

On the completion of this thesis I would like to express my appreciation to all those who have contributed in one or another way.

First I would like to thank prof.dr.ir. Snijders and prof.dr. Gelsema for their support and for their precise supervision of this dissertation. I enjoyed many instructive and inspiring conversations that have given me deeper insight into the field of ambulatory monitoring and pattern recognition. Their patient and competent supervision has given me the chance to learn many new things. It has been a great pleasure working with them.

I would like to thank all my colleagues working on the 16th floor, for their advise and help, and especially to our secretaries, Ria Van Kruining for administrative assisting where needed.

I would like to thank people of UAF (University Assistance Fund) for their advise and help.

I would like to thank my wife Mandana, my parents, my brothers and all my friends for their support.

Finally, I would like to thank this country (the Netherlands) for making it possible for me to get a good education. I am very grateful that I was always able to follow my interests.

Curriculum vitae

Kourosh Kiani was born on November 12, 1960, in Pol Sefid, Iran. From 1978 to 1979 he studied physics in Mashhad (Iran). In 1993 he received his MSc. degree in Electrical Engineering from the Delft University of Technology in Delft, the Netherlands (1987-1993). In August 1993, he joined the Department of Biomedical Physics and Technology at Erasmus University Rotterdam, to work as a scientific researcher under the supervision of Prof.dr.ir. Snijders and Prof.dr. Gelsema.