

THE IBY AND ALADAR FLEISCHMAN FACULTY OF ENGINEERING

Diagnosis of OSA Syndrome from PAT and Oximetry Recordings Using Fuzzy Decision Trees

A thesis submitted toward the degree of
Master of Science in Engineering Sciences

by
Michal Betito

August 2004

Diagnosis of OSA Syndrome from PAT and Oximetry Recordings Using Fuzzy Decision Trees

A thesis submitted toward the degree of
Master of Science in Engineering Sciences

by
Michal Betito

This research was carried out in the Department of Interdisciplinary Studies
under the supervision of Dr. Dana Ron

August 2004

Abstract

Along with increased awareness to obesity related health problems and the importance of sleep quality, the need for an ambulatory device that will help in diagnosing obstructive sleep apnea syndrome has increased significantly. In this thesis we investigate the problem of diagnosing Obstructive Sleep Apnea Syndrome (OSAS) from the Peripheral Arterial Tonometry (PAT) signal and two additional channels (Oximetry and Actigraphy).

We aim at automatically generating a diagnostic algorithm that will be precise in terms of the RDI (Respiratory Disturbance Index) and will also estimate apnea events time locations. The algorithm should also give us insight to the markers of apnea events in these signals.

We describe the signal processing algorithms used for extracting features from the recorded channels. We suggest a way to create discrete time examples from the channels and discuss ways to match features and classify the examples according to the gold standard. We suggest a way to optimize memory consumption and computational complexity while generating fuzzy decision trees.

We also suggest a way to incorporate a-priory expert knowledge by determining a construction of attribute hierarchies and show that this mechanism of attribute hierarchies can be used for handling missing attributes, where the fact that an attribute is missing can be significant to the classification of an example.

The algorithm presented in this thesis was trained on a set of recordings performed at a sleep laboratory (Rambam Medical Center; Israel) and synchronized with recordings of PAT, oximetry and actigraphy. It was then tested on a set of validation files. The results obtained are compared with those of two heuristic algorithms that were developed. The comparison shows that our algorithm gives superior results in terms of patients diagnosis for treatment. The thesis algorithm achieved sensitivity of 89% and specificity of 83%, where the two former algorithms gained sensitivity of 73% and 93% and specificity of 67% and 63% respectfully.

Furthermore, the algorithm managed to achieve good statistical grades with very little tuning of parameters and human intervention. We also believe that it may be useful for other problems that have similar characteristics. In particular, this may be the case when there is access to labelled data, but the labelling was done by an expert that was provided with additional information. The expert can possibly design a rule using the additional information, but cannot do so without it. In such a case, we can use our automated algorithm that, given labelled data, learns a rule that is not based on the additional information.

Contents

1	Introduction	1
1.1	Related work	2
1.1.1	The relationship between the PAT and Sleep Apnea . . .	2
1.1.2	Estimating Apnea events existence from non-PSG signals	3
1.1.3	Artificial intelligence in similar Bio-medical problems . .	4
1.1.4	Organization	4
2	Background	6
2.1	Physiological Background	6
2.1.1	Sleep	6
2.1.2	Sleep disorders	7
2.1.3	The PAT technology	9
2.2	Machine learning	11
2.2.1	Decision Trees	11
2.2.2	Fuzzy Logic	14
2.2.3	Fuzzy Decision Trees	20
3	Methods	27
3.1	General	27
3.2	Experimental setup	28
3.2.1	Polysomnography	28
3.2.2	PAT-Watch	29
3.2.3	ZZPAT pre-processing	29
3.2.4	PAT Amplitude and Heart Rate Extraction	29
3.3	Feature Extraction	30
3.3.1	Signals Pre-Processing	30
3.3.2	Extracting Physiological Attributes	31
3.3.3	Defining negative and positive examples.	42
3.4	Learning From Examples.	46
3.4.1	Algorithm flow	47
3.4.2	Attribute hierarchy	47
3.4.3	Preliminary tree building	48

3.4.4	Considerations of computational complexity and memory allocations.	49
3.4.5	Pruning.	50
3.4.6	Accuracy testing.	51
3.4.7	Saving and loading the tree's structure using a binary file	52
4	Experimental Results.	54
4.1	Basic algorithm	54
4.2	Rule base algorithm	55
4.3	Results of the suggested algorithm	57
4.4	Comparative analysis	59
4.4.1	measures for comparison	59
4.4.2	Statistical results	60
5	Summary	62
	References	64
A	Data file format	67
B	List Of attributes and their use in the algorithm.	69

Medical Glossary

- *Autonomous Nervous System* Neurons that are not under conscious control, it regulates key functions including the activity of the cardiac (heart) muscle, smooth muscles (e.g., of the gut), and glands. The autonomic nervous system has two divisions: The sympathetic nervous system that accelerates the heart rate, constricts blood vessels, and raises blood pressure. The parasympathetic nervous system slows the heart rate, increases intestinal and gland activity, and relaxes sphincter muscles.
- *Sleep* A natural body function during which we are relatively unconscious and the muscles that we normally control are relaxed. The result of sleep is a refreshment of the nervous system (including the brain) and of the muscles.
- *Apnea* Cessation of breathing during sleep for a period of 10-60 seconds.
- *Hypopnea* A partial cessation of breathing.
- *Respiratory Disturbance Index (RDI)* The number of apnea and hypopnea events per hour.
- *Electroencephalogram (EEG)* A diagnostic test which measures the electrical activity of the brain using high sensitive recording equipment attached to the scalp by fine electrodes.
- *NREM (Non-REM) Sleep* Comprised of Stages 1-4 and lasts from 90 to 120 minutes, each stage lasting anywhere from 5 to 15 minutes. Stage 1 - drowsiness, stage 2- light sleep, stage 3, 4 - deep sleep
- *REM Sleep (Rapid Eye Movements)* Also called paradoxical sleep due to the contradiction between the similarity to stage 1 in the EEG and the immobility of the body muscles. This is the stage in which dreaming takes place.
- *Polysomnography* A sleep study in which physiological monitors are attached to the patient to record nighttime breathing, brain activity, and physical activity.
- *PAT (Peripheral Arterial Tone)* The PAT signal measures the circulatory responses based on the magnitude and time course of changes in arterial blood flow in the fingertip.

- *CHF (Congestive Heart Failure)* A condition where there is ineffective pumping of the heart leading to an accumulation of fluid in the lungs. Typical symptoms include shortness of breath with exertion, difficulty breathing when lying flat and leg or ankle swelling. Causes include chronic hypertension, cardiomyopathy and myocardial infarction.
- *ROC (Receiver Operating Characteristic) Curve* A plot of the true positive rate against the false positive rate for the different possible cutpoints of a diagnostic test.
- *QRS complex* The deflections in an electrocardiographic tracing. It represent ventricular activity of the heart.

List of Figures

2.1	PAT probe pneumatic structure	10
2.2	Pressure-Volume compliance curve	11
2.3	Crisp (left) and fuzzy (right) sets of short, average and tall man.	15
2.4	Examples of Triangular, Trapezoid, Sigmoid and Gaussian membership functions.	16
2.5	A graphical illustration for applying a fuzzy rule.	18
2.6	Fuzzy diagram inference flow	19
2.7	Solving the tipping problem via a Mamdani style fuzzy inference system.	20
2.8	Zero Order Sugeno style fuzzy inference system for the tipping problem.	21
2.9	An example of a fuzzy decision tree for the tipping problem.	22
3.1	PAT amplitude and heart rate noise reduction and filtering to avoid miss-identified complexes and artifacts.	31
3.2	A demonstration of detecting change events in the amplitude and the HR extracted from the PAT signal.	34
3.3	A demonstration of detecting change events in the Oximetry channel.	35
3.4	Detection of the background level on the actigraph channel in the presence of respiratory modulations.	37
3.5	Actigraph movement detection example	38
3.6	Local re-saturation index calculations.	39
3.7	A demonstration of finding examples locations.	45
4.1	A scatter plot of the basic algorithm results.	55
4.2	A scatter plot of the rule base algorithm results.	57
4.3	A scatter plot of the thesis suggested algorithm results.	58

Chapter 1

Introduction

In recent years there is an increasing understanding of the effect sleep quality on the risk of exposure to heart diseases, blood pressure problems and excessive daytime sleepiness. As a consequence, there is greater need for an easy, non-invasive, automatic procedure for determining the sleep quality of a patient. The measurement of the rate of apnea events (cessations of breathing periods) per hour of sleep is an important measure of the sleep quality.

The common procedure nowadays usually involves spending a night at a sleep laboratory. The patient is attached to a PSG (polysomnography) system that records many signals; among them are two respiratory belts, a thermistor that measures the nasal airflow, and sometimes also a pressure canula, some EEG (electroencephalogram) channels and an EMG (Sub mental tension) channel. From the recorded channels, apnea and arousal events are scored visually.

This procedure suffers from some disadvantages: The number of patients that can be tested in one night is limited; the discomfort of sleeping in a strange environment affects the tested sleep quality. Furthermore the task of visually analyzing the recordings from the PSG is time consuming, expensive and suffers from great inter-scorer (and even intra-scorer) variability.

An alternative procedure for detecting apnea was suggested by Scnall et. al [1]. It is based on the fact that an apnea usually ends with either an arousal from sleep or with a desaturation. Since one of the by products of an arousal is peripheral vasoconstriction caused by the sympathetic nervous system, this vasoconstriction can be a significant marker for the occurrence of an apnea event.

This work is about defining a method to identify the apnea/hypopnea index (i.e., the number of apnea/hypopnea events per hour of sleep) of a patient from recordings of the PAT (Peripheral Arterial Tone), oximetry and the actigraphy signals only. This is done using an automated learning algorithm, that is customized for the special characteristics of this problem.

While working on the thesis, a better understanding of the physiological

reflections of this phenomena was gained. Signal processing techniques were used for artifact filtration and for the extraction of relevant features. The question of how to divide the extracted features (which are time-based) into discrete examples was addressed, and found its solution in the form of detecting location of local maximal probability for occurrence of an apnea event, this local probability was estimated from the locations of events extracted from the recorded signals, while taking into consideration the typical time delays caused by the physiology and the measurement instruments.

Several methods from the field of machine learning were tested; eventually the algorithm was chosen to be generated using Fuzzy Decision Trees. This algorithm is modified for the needs of our problem, including a special handling for cases in which feature values of certain events could not be measured, due to the fact that the event did not exist or did not pass a minimal threshold.

Attribute hierarchy was introduced to solve this problem, as well as for allowing different rules to be extracted according to a-priory physiological knowledge.

This algorithm is compared with previous algorithms that were developed for the same problem, and run on the same set of data. We have found that our algorithm showed better results, mainly in terms of patients screening for treatment.

We believe that by choosing the membership functions on the node in a proper automatic manner, the improvement can be much more significant.

1.1 Related work

1.1.1 The relationship between the PAT and Sleep Apnea

The first publication that introduces the possibility to learn about the existence of obstructive sleep apnea from peripheral vasoconstriction was reported in 1999 [1]. The authors showed a very high correlation ($r = 0.92$) between the total number of events of transient peripheral vasoconstriction that coincide with events of tachycardia (increase of heart rate), and the total apnea-hypopnea scoring in patients that suffer from severe OSA syndrome. These markers of apnea were also found to be useful in estimating the arousals from sleep index, as was first suggested by Lavie et. al [2]. Devices and algorithms based on these concepts were developed and assessed [3, 4, 5]. In Chapter 4 we describe some of them and compare their results with that of our algorithm.

Grote et al [4] had studied the effect of changes in the finger blood flow versus changes in fore-arm vascular flow after infusion of nor-epinephrine. The authors had reached the conclusion, that the finger blood flow is correlated with stimulation of the alpha receptors, rather than with beta receptors, whereas

in the wrist activation of both alpha and beta receptors were present. This means that we can separate the influence of the sympathetic activity from that of the parasympathetic activity by measuring the blood flow in the fingertip.

1.1.2 Estimating Apnea events existence from non-PSG signals

Other publications focused on attempting to identify the existence of apnea events from non-PSG signals. Another algorithm for the evaluation of the RDI (Respiratory Disturbance Index) from the Heart rate and Oximetry channels is described [6]. It is based on finding events of changes in the heart beat that coincide with a rise in the oximetry level. It is found that an index of those events (termed CODI) correlated significantly with the RDI. The reported results are very good (a correlation coefficient of 0.88 and an Area under curve of the ROC of 0.94). However, these results were only obtained after removal of recordings in which the index of the oximetry changes differed by more than 24 from the index of changes in heart beat, resulting in the removal of over 30% of the tested cases. The agreement of the indices when all the tested examples are taken into consideration is significantly lower (Correlation of $75 \pm 0.07\%$).

Pensel et. al [7] had tested the possibility of estimating the AHI from the pulse transient time (PTT) and from the PAT, and had shown that these two measures can indicate the existence of an arousal and of an apnea event. They stated that these two signals mirror completely different aspects of the cardiovascular and the autonomic nervous system; The PTT being closely linked to blood pressure and the PAT closely linked to peripheral arterial tone. Therefore we cannot be substitute one by the other. The authors show a connection between these two phenomena and the existence of apnea events, but no qualitative analysis was made.

In [8] the authors suggest a method to define the respiration frequency and the presence of an apnea from calculating the EDR (i.e. the energy of the QRS waves) values from QRS area variations. The method was to first choose the most characteristic one or two ECG channels out of the 8 recorded channels, then to analyze the interpolated curve of the area under each QRS complex, and search for meaningful decreases on this curve, which implies for the existence of an apnea event. Though they had a very poor statistic (only 3 patients showed apnea events) they show high values of sensitivity and specificity (of 87% and 85% respectfully).

1.1.3 Artificial intelligence in similar Bio-medical problems

Many publications describe the use of artificial intelligence for scoring medical algorithmic problems. Though no reports were found on the use of artificial intelligence for apnea detection, some interesting reports that involve detection of alpha waves from EEG signals, and ECG patterns classifications were found. We will briefly describe two articles that deal with these issues through use of fuzzy logic and of decision trees:

Detection of alpha activity via use of fuzzy logic is described in [9]. In this article the authors have found a way to declare the existence of alpha waves activity in the EEG, without determining a constant threshold for the energy of the alpha waves. They use FFT to determine the ratio of energy in the alpha range to the energy in the delta range (0.8-3Hz), in the Theta range (3.5-7.5Hz) and in the Beta range (20-40Hz). These ratios are calculated twice (once for a window size of 2.5 seconds and again for a window size of 10 seconds). The ratios from each window size are inferred via a fuzzy logic system to give a score for the alpha activity in the window. These two scores are combined (again via Fuzzy logic inference) to give the final result.

Bensaid et. al [10] derive fuzzy rules from ID3-decision trees in order to classify ECG patterns. They first build an ID3 decision tree from the training examples, then conjunctive rules are extracted from the tree. The rules are then fuzzified by using membership functions chosen by cardiologists, and using a neural network for optimizing the defuzzification process.

1.1.4 Organization

The following Chapter (Chapter. 2) describes the physiological background for our problem. Section 2.1 starts with a short description of sleep and sleep stages, through sleep disorders and the PAT signal and its connection with sleep disorders. Section 2.2 describes the relevant methods from machine learning. It starts with description of decision trees algorithms, then describes the concepts of fuzzy logic. The combination of these two methods into fuzzy decision trees is explained in Section 2.2.3.

The implementation of the algorithm is described in Chapter 3. The experimental setup is described in Section 3.2. Section 3.3 describes the signal processing and feature extraction phases that were developed for generating the examples. In Section 3.4 there is a description of the program that builds the fuzzy decision trees. The input parameters and data files are described, the algorithm flow and special considerations are detailed in this section also.

Chapter 4 describes the experimental results of our research work. The algorithm's performance is tested and compared with the performance of two earlier algorithms. Chapter 5 summarizes the conclusions from our work.

After the list of references, there are two appendixes: The first one describes the format of the files used as input for the learning algorithm. The second one summarizes the list of attributes inserted into the learning algorithm and describes their contribution to the generated fuzzy tree.

Chapter 2

Background

2.1 Physiological Background

2.1.1 Sleep

People spend almost one third of their lifetime sleeping. Although it may seem like a uniform one passive state, sleep is rather an active physiological process, with unique dynamics and various different sleep stages.

The normal sleep pattern consists of five sleep stages (i.e., stages 1 to 4 and Rapid Eyes Movement (REM) sleep), which are organized in sequential 90 minutes cycles. In general, there are normal values for the time-proportions between the sleep stages. In adults, stage 1 lasts for up to 5% of the sleep time, stage 2 is the longest (50%), stages 3 and 4 are defined as the Slow Wave Sleep (SWS) last about 20-25% and REM sleep (20-25%). It is interesting to note that, in the neonate there is a relative REM sleep predominance (50%), and then, through infancy and childhood there is a gradual progression to the above-mentioned adult value (25%). The sleep stages are not equally distributed along the sleep cycles; REM sleep is getting relatively longer from the first cycle to the last one, while SWS is getting shorter from the first to the last cycle.

The various sleep stages differ in their physiological characteristics. A Polysomnography (standard sleep study; PSG) test can be used in order to evaluate the sleep architecture and sleep stage-dependent pathological conditions. It is considered a “gold standard” for evaluation of normal versus disturbed sleep.

During the night study, the PSG records various physiological parameters. Three of those are used for scoring of sleep stages for determination of sleep fragmentation, these are: electrical activity of the brain cortex (measured by Electroencephalogram (EEG)), Eye movements (measured by Electroculogram (EOG)), and muscle tonus (measured by chin Electromyogram (EMG)).

In general, in the Non-REM sleep stages there is a gradual progression

of the EEG pattern: from “high frequency-low amplitude” EEG pattern of wakefulness, in stages 1 and 2, to “low frequency-high amplitude” waves of SWS. Eye movements in Non-REM sleep stage are only seen during stage 1, in the characteristic form of slow eyes movements. During REM sleep, both the cortical activity pattern and the eyes movements are very similar to wakefulness (i.e., “high frequency-low amplitude” EEG and rapid eyes movements). The muscles’ tonus are reduced during Non-REM sleep (in comparison with wakefulness), and further reduction is seen in the transition from Non-REM sleep to REM sleep. This muscle tonus differences helps to distinguish REM sleep from wakefulness (which are very similar in the two other measures).

2.1.2 Sleep disorders

Sleep disorders are diverse by their origins and manifestations, although they frequently share signs and symptoms. For example, Sleep Apnea Syndrome (SAS) is a respiratory disturbance that is manifested during sleep by snoring and recurrent cessations of breathing, whereas, Periodic Limbs Movements Syndrome (PLMS) is a neurological disorder, which is characterized by rhythmical limbs’ jerks. Both of these disorders can lead to sleep fragmentation, and as a consequent, clinical signs and symptoms of daytime sleepiness.

For clinical purposes, sleep disorders can be classified into four categories:

1. Insomnias - Despite the fact that the patient is tired at bedtime, he has one or any combination of the following three difficulties:
 - To fall asleep.
 - To maintain sleep.
 - Suffers from early morning awakening.
2. Hypersomnias - The patient is sleepy although he has slept adequate time.
3. Parasomnias - Undesired events during sleep (such as nightmares).
4. Circadian rhythm abnormalities - Undesired sleep-wake schedule.

Sleep Apnea Syndrome (hypersomnia)

SAS is the most common sleep disorder, a well-known major public health problem. The estimated prevalence of the syndrome in otherwise healthy adults is 4% and 9% in men and women respectively, while in otherwise healthy children it is about 2-5%. In adults, the severe long-term complications originate from the cardiovascular system, i.e., cerebro-vascular accidents, pulmonary and systemic hypertension, arrhythmias, myocardial ischemia and congestive

heart failure (CHF). The most common significant consequences of SAS in children are neuro-cognitive impairments, while cardiovascular complications such as systemic and pulmonary hypertension, are less frequent than in adults.

The syndrome is characterized by recurrent events of cessation (Apnea) or reduction (Hypopnea) of breathing, with the consequences of hypoxia (reduction of oxygen level in the blood), hypercapnia (excess of carbon dioxide in the blood), arousals from sleep, and sympathetic nervous system outflow surges.

In general, disordered-breathing during sleep can be further sub-classified as obstructive, central or mixed, based on the events' pathophysiology. Obstructive events are caused by the partial/complete occlusion of the upper airways (normal respiratory efforts), central apnea is defined as cessation of ventilation due to the lack of respiratory drive, and mixed event consists of both components, obstructive and central.

The vast majority of sleep apnea cases have the obstructive form, and the typical OSAS patient is an otherwise healthy middle-age adult, who snores and complains about daytime sleepiness. In many cases, snoring is the primary complaint that brings the patient to the physician. Thus, beyond the social problem, that annoying noise assists in the diagnosis process. The sound of snoring is generated by the vibration of the walls of Upper Airways (UA), and is caused by the turbulence of the airflow against the narrowed UA.

In OSAS patients the UA can be narrowed either anatomically (e.g., deviated nasal septum, lymphoid tissue hypertrophy), or functionally by the collapse of the UA walls. The later occurs when the intra-luminal inspiratory negative pressure exceeds the muscular tone of the UA walls.

In most of the sleep-labs, patients with predominantly CSAS (Central Sleep Apnea Syndrome) constitute less than 10% of the sleep apnea patients, and the majority of the cases have the obstructive form. The "pure" CSAS patients are older and rarely otherwise healthy, and most of them have an underlying cardiac disease such as CHF. About 50% of the patients with moderate-to-severe CHF, exhibit Cheyne-Stoke Breathing (CSB), a unique "diamond" pattern of ventilation. It has been shown that patients with moderate to severe CHF with CSB have a poorer prognosis than CHF patients without CSB. Since in CSB the tidal volumes' "diamond" shapes are aligned with the same pattern of ventilatory efforts, it is classified as a central apnea variant. The pathophysiology of central sleep apnea and CSB in CHF patients is a vicious cycle; the heart failure leads to pulmonary congestion, which leads to hypoxia and activates lung vagal receptors, both of them trigger hyperventilation (the hyperventilation phase). A hyperventilation result in hypocapnia that eventually reaches a level below the threshold for ventilation, thus, central apnea is induced (hypoventilation phase).

The severe complications of SAS accentuate the importance of early diagnosis and treatment of the syndrome. The gold standard for SAS diagnosis is in-lab Polysomnography (PSG), and other ambulatory systems have incom-

plete montage of channels. The latter group of devices is in clinical use due to the limited number of in-lab beds. Various therapeutic modalities with different indications and efficacies are available, from conservative treatment such as weight reduction and “tennis ball”, via Continuous Positive Airway Pressure (CPAP) to oral appliances devices and surgical procedures.

2.1.3 The PAT technology

The PAT signal measures the pulsatile volumetric changes at the fingertip’s arterial vasculature, a reflection of the vascular smooth muscle tone changes.

The signal reflects changes in the autonomic nerve system: When the body experiences an episode of physiological stress, the autonomic nervous system acts in order to route the blood to vital organs - thus causing less blood to arrive at the peripheral organs - the limbs. The activation of the sympathetic nervous system leads to digital vasoconstriction via alpha-adrenergic receptors (the effect of the system on the β receptors results in vasodilatation, but, since only α receptors exist in the fingertip’s arteries, the vasodilatation is not apparent in the PAT measurement [4]). Therefore, as sleep respiratory events lead to sympathetic activations (manifested as vasoconstrictions), they can be detected by PAT amplitude attenuations.

Previous works (e.g., [1]) show, that the PAT signal decreases in episodes of desaturation and of arousals. In [11] the authors attempt to separately evaluate the contribution of the airflow limitation and of the arousals to the PAT attenuation, resulting in the conclusions that severe airflow limitations cause significant PAT attenuations whereas mild airflow limitations do not. The response is accentuated in the presence of an arousal.

The PAT probe of the WatchPAT100 is a “pneumo-optical” plethysmograph: the measurement of the change in volume is done via an optical density measurement, which is executed under specific physical condition that is generated by the probe’s pneumatic apparatus.

Principally, the pneumatic apparatus applies a uniform sub-diastolic external pressure field around the two distal phalanges (bones) of finger, including the very tip of the finger. An illustration of the probe’s pneumatic structure is shown on Figure 2.1.

This structure is aimed mainly at increasing the dynamic range of the signal by reducing the transmural pressure of the finger’s arteries. This allows for a greater arterial pulsatile volume changes in response to any given pulsatile blood pressure changes (See Figure 2.2). Therefore, the dynamic range of the measurement is increased, and the sensitivity, robustness and reliability of the acquired signal are improved.

The prevention of venous pooling is also a critical feature of the PAT probe, since:

- Venous pooling for more than a few minutes causes pain that leads to

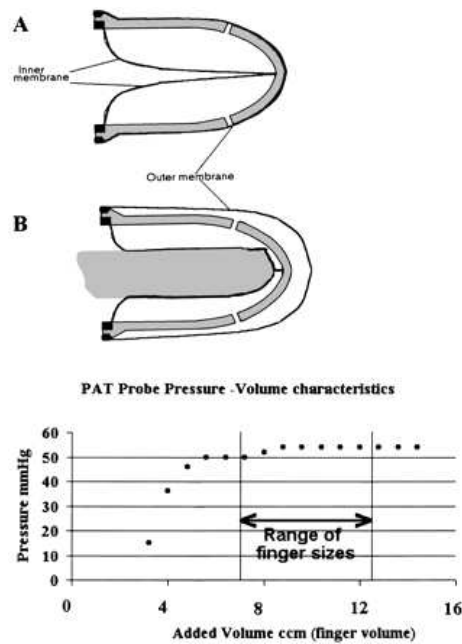


Figure 2.1: PAT probe pneumatic structure. A. shows the probe before the finger is inserted, B. shows the probe after the finger is inserted. The lower plot shows the indifference of the applied pressure to the volume of the inserted finger.

arousals from sleep, which obviously, interfere with our measurement of the sleep quality.

- Venous pooling increases the venous pressure, which leads to veno-arteriole vasoconstrictor reflex; a “local artificial vasoconstriction” rather than “systemic physiological vasoconstriction”, which is also undesired. Other features of the probe are:
 1. Indifference to the inserted-finger’s volume, the same optimal sub-diastolic pressure will be applied around the finger.
 2. The measurement site of the optical element is located distally from the blood turbulence area caused by the change of pressure near the open edge of the probe.

The information from the PAT signal is further combined with the heart rate, pulse oximetry and actigraphy data in order to approximate the index of respiratory events, as will be explained on Sections 3 and 4.

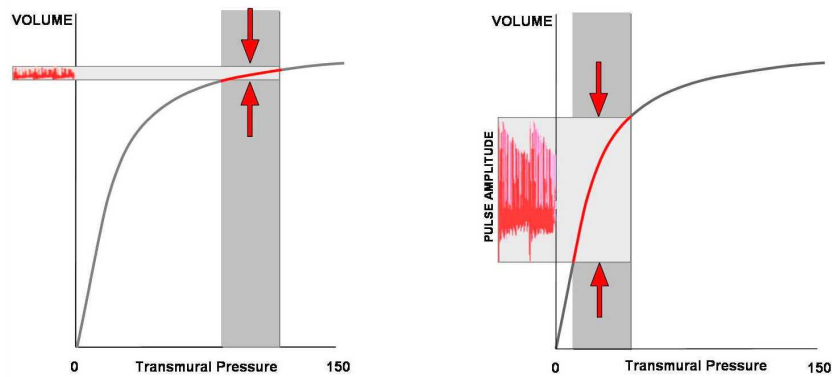


Figure 2.2: Pressure-Volume compliance curve. The figure shows the compliance curve of arterial pressure to volume. The left/right figure shows the work-load range of compliance curve without/with activation of exterior pressure of the PAT probe. The applied pressure reduces the transmural pressure of the blood vessel, thus leading to an increased response in terms of change in volume VS change in pressure.

2.2 Machine learning

2.2.1 Decision Trees

Decision trees form a method for building knowledge-based classification systems by inductive inference from examples.

A decision tree consists of decision nodes and leaves, connected by branches. Each node specifies some test to be carried out on a single attribute value, with one branch and sub-tree for each possible outcome of the test. Each node that is not split into branches and sub-trees is considered a leaf, and indicates an output class.

One major advantage of decision trees, is that they form a classification process that enables good comprehensiveness of the rules extracted by it, and thus allows for better domain understanding (which is not possible in some other machine learning methods, e.g., Neural Networks).

The tree is built by non-incremental (i.e., batch) learning from examples. Each example consists of a set of symbolic attributes and a classification. The

tree is developed from the top down, and the process is guided by the frequency of the examples' classification.

The idea of building a decision tree based on examples was first introduced by the work of Hoveland and Hunt in the late 1950's. Friedman [12] layed the foundations for the CART system [13] for building decision trees for continuous valued attributes. Quinlan, who was also the one to popularize decision trees, had developed ID3 [14] for building trees for symbolic attributes. A later version of ID3 was implemented on a computer software named C4.5 [15]. The method for building a symbolic decision tree according to C4.5 is described in the following subsections.

Notations

. The notations that will be used for describing decision trees are:

- $A = \{a_1, a_2, \dots, a_n\}$ denote the set of attributes.
- $D_i = \{a_{i1}, \dots, a_{il}\}$ is the symbolic domain of attribute i , a_{ij} is the symbolic value j of attribute a_i .
- C is the set of output classifications.
- For each node N , $E(N) = \{e^1, \dots, e^m\}$ is the set of examples corresponding to the node, where e^j equals $\{e_1^j, \dots, e_n^j, y^j\}$, $e_i^j \in D_i$ and $y^j \in C$.
- For a node N that is tested by attribute a_i , $N|a_{ij}$ is the child node of N , fulfilling the test $a_i = a_{ij}$.

Symbolic Tree building.

A subset of examples is randomly chosen from the training set. On this set of examples, the decision tree will be built iteratively (the remaining examples will be used for pruning).

1. **Calculate the information of the examples in the node.** Starting from the root node, calculate the node's entropy:

$$I(N) \triangleq - \sum_{i=1..|C|} p_i \cdot \log_2(p_i)$$

where

$$p_i = \frac{|\{e^j : e^j \in E(N) \text{ and } y^j = i\}|}{|E(N)|}$$

2. **Choose the best attribute via gain calculations.** Now we need to decide which attribute is the best for splitting the examples in the root node, in a manner that will minimize the amount of entropy that will remain after the split is done. The amount of information that will remain after splitting the node according to attribute a_i is:

$$\sum_{j=1}^{|D_i|} \frac{|E(N|a_{ij})|}{|E(N)|} I(N|a_{ij})$$

The gain from splitting the node by that attribute is

$$Gain = I(N) - \sum_{j=1}^{|D_i|} \frac{|E(N|a_{ij})|}{|E(N)|} I(N|a_{ij})$$

3. **Split the node.** The attribute chosen to be tested at the node is the one that maximizes the gain. After the attribute had been chosen, $|D_i|$ branches and child nodes are formed.
4. **Recursive node splitting and termination criteria.** The stages of calculating the gain of the examples in a node, and of splitting the node are performed recursively until all the examples in that specific node are exclusively classified, or until no more meaningful node splitting can be done (gain for all attributes is zero), this node is then referred to as a leaf.

Handling missing attributes.

There are cases in which certain attributes cannot be measured for some of the examples. The offered solution to this problem (in C4.5) is to alter the gain criteria by multiplying it by the fraction of the examples with a known value for this attribute. At the classification stage, whenever an example with an unknown value for the tested attribute is encountered, it is evenly split between the branches of the node. The outcome in this case will be a distribution of the output classes (rather than a single classification).

Pruning.

When building the tree from the examples, there is a possibility that the tree generated will be very complex and will “over-fit the data”. There are two approaches for simplifying a tree:

- *pre-pruning* - deciding not to divide a set of training cases any further. This can be done by putting a threshold on some measure such as the error rate or minimal gain for splitting.

- *post-pruning* - removing some of the structure built by recursive partitioning.

This is usually performed by choosing a set of examples, different from the ones used for training.

For each node, we replace the sub tree originated by it with a leaf (pruning). The accuracy of the original tree and the pruned tree are compared, if the pruned tree has better accuracy, than the it is kept, else the original tree is maintained.

2.2.2 Fuzzy Logic

As stated by Zadeh 1965:

Fuzzy logic is determined as a set of mathematical principles for knowledge representation based on degrees of membership rather than on crisp membership of classical binary logic.

Fuzzy logic is an extension of boolean logic. It defines the ideas and mathematics of dealing with sets that enable their variables to belong to them to a certain degree. This characteristic is very useful in classification problems where we are dealing with a numeric attribute that has a symbolic representation which does not have distinct boundaries, or whose boundaries might be subjective. In such a case, using the crisp representation can result in an error in classification.

Figure 2.3 shows a crisp and a fuzzy representation of the attribute Tall. In the crisp representation, a man of 179cm height is considered as being of average height, whereas a man of 181cm height is considered tall. In the fuzzy representation, both men are considered as being of average height to some extent, and tall to some extent. On the contrary, a person whose height is 200cm is classified as “definitely tall”.

For an illustration of the fuzzy logic concepts, we will use an example and some figures taken from [16]. The example discusses the everyday problem of how much to tip in a restaurant. The answer depends upon the level of service and the level of the food. It also depends on some prior knowledge about the output: how much is considered a cheap, average or generous tip, other considerations can also be taken into account.

This problem can be solved without the use of Fuzzy logic. It will require us to determine the mathematical relations between the input variables and the output. These relations can be rather complicated, therefore the task of understanding and modifying the code can be a difficult one.

We will try to show that a fuzzy logic representation improves the comprehensiveness of the solutions to problems, where intuitive understanding is required and especially where prior knowledge of the problem should be used.

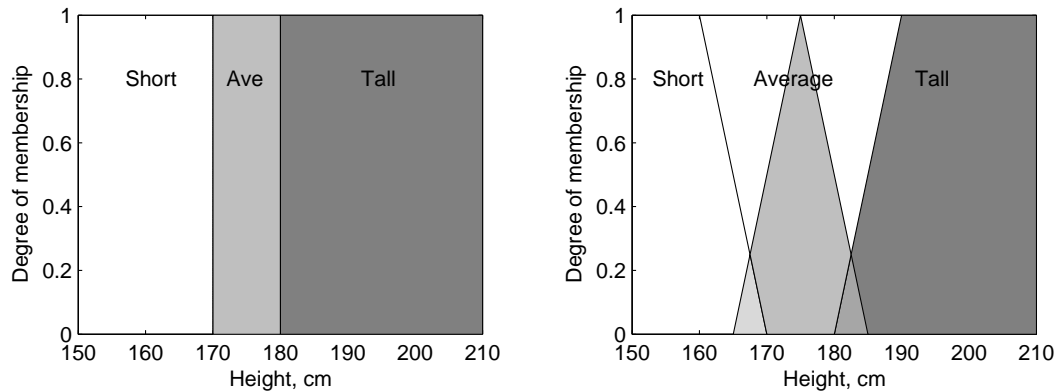


Figure 2.3: Crisp (left) and fuzzy (right) sets of short, average and tall man.

It also allows the output to be of a smoother nature with respect to changes in the input variables.

Fuzzy sets

A fuzzy set is a set with fuzzy boundaries. It is represented by a membership function, defined as follows:

$$\mu_A(x) = 1 \text{ if } x \text{ is totally in } A$$

$$\mu_A(x) = 0 \text{ if } x \text{ is not in } A$$

$$0 < \mu_A(x) < 1 \text{ if } x \text{ is partially in } A$$

In principle, every continuous function that holds the formally stated conditions, can be used as a fuzzy membership function. The most commonly used ones are the Triangular, Trapezoid, Sigmoid and Gaussian ones. Examples of these membership functions are shown in figure 2.4.

Linguistic variables

Fuzzy logic is usually used for building fuzzy rules that can be easily understood by humans. Therefore, it is common to describe fuzzy variables as linguistic variables.

In the tipping example, the linguistic variables are the quality of the service and the quality of the food. The linguistic variable “quality of service” can take the linguistic terms: poor, good or excellent. The linguistic variable “quality of food” can take the linguistic terms rancid, good or delicious.

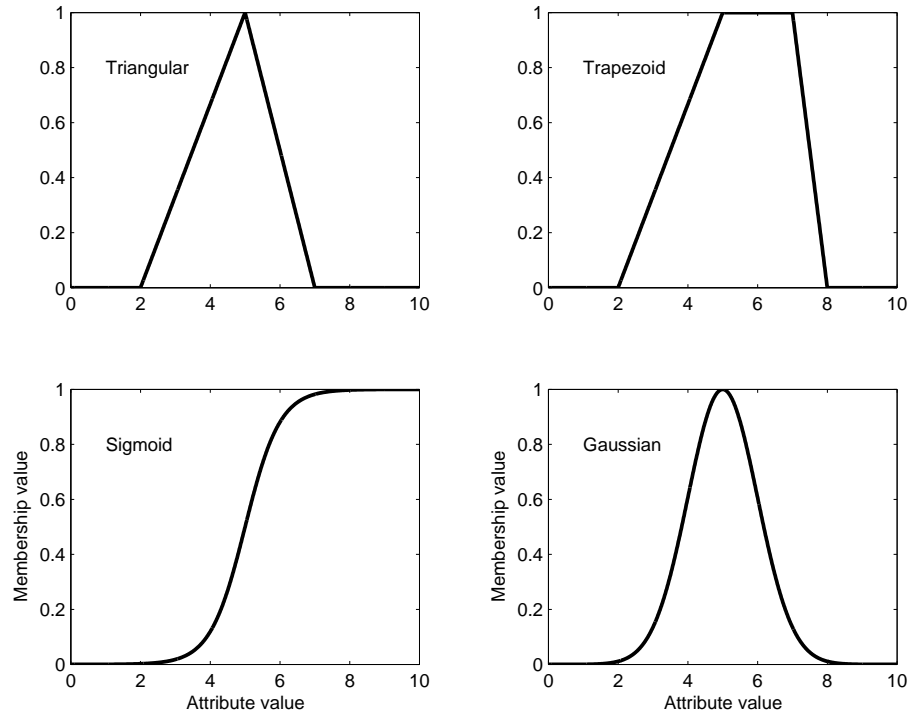


Figure 2.4: Examples of Triangular, Trapezoid, Sigmoid and Gaussian membership functions.

Operations on fuzzy sets

The following operations on fuzzy sets are defined:

- **Complement:** $\mu_{-A}(x) = 1 - \mu_A(x)$
- **Intersection:** The fuzzy generalization of this operator is termed *T*-Norm. there are several ways to define this operator on fuzzy sets, two of the more common ones are min and prod:

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

$$\mu_{A \cap B}(x) = \text{prod}[\mu_A(x), \mu_B(x)] = \mu_A(x) \cdot \mu_B(x)$$

- **Union:** The generalization of this operator is termed *S*-Norm. We will state here the more popular max and probor implementations:

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

$$\mu_{A \cup B}(x) = \text{probor}[\mu_A(x), \mu_B(x)] = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)$$

Fuzzy rules

A fuzzy rule is a rule of a type:

If x is A

then y is B

where x and y are linguistic variables and A and B are linguistic values, determined by their fuzzy sets. The first part of the rule is called the antecedent, and can consist of multiple parts with the operators AND or OR between them. The latter part is called the consequent, and can also include several outputs.

Three fuzzy rules were chosen for the tipping problem:

1. *If service is poor or food is rancid then tip is cheap.*
2. *If service is good then tip is average.*
3. *If service is excellent or food is delicious then tip is generous.*

Figure 2.5 illustrates the antecedent and the consequent of the third rule and its Mamdani style inference as explained in the next subsection.

Fuzzy inference.

Fuzzy inference is the process of mapping from a given input to an output, using the theory of fuzzy sets. There are two types of inference systems:

1. **Mamdani-style inference.** This is the most commonly used inference technique. It was suggested by Mamdani at 1975. The inference process is performed in four steps: fuzzification of the input variables, rule evaluation, aggregation of the rule outputs and defuzzification.

Fuzzification. Taking the crisp inputs, and determining the degree to which these outputs belong to each of the appropriate fuzzy sets.

Rule evaluation. Taking the fuzzified inputs, and applying them to the antecedents of the fuzzy rules.

If a given fuzzy rule has multiple antecedents, a fuzzy operator (AND or OR, translated to the generalized union and intersection as stated above) is used to obtain a single number that represents the result of the antecedent evaluation. This number is then applied to the consequent membership function. This stage is illustrated in figure 2.5.

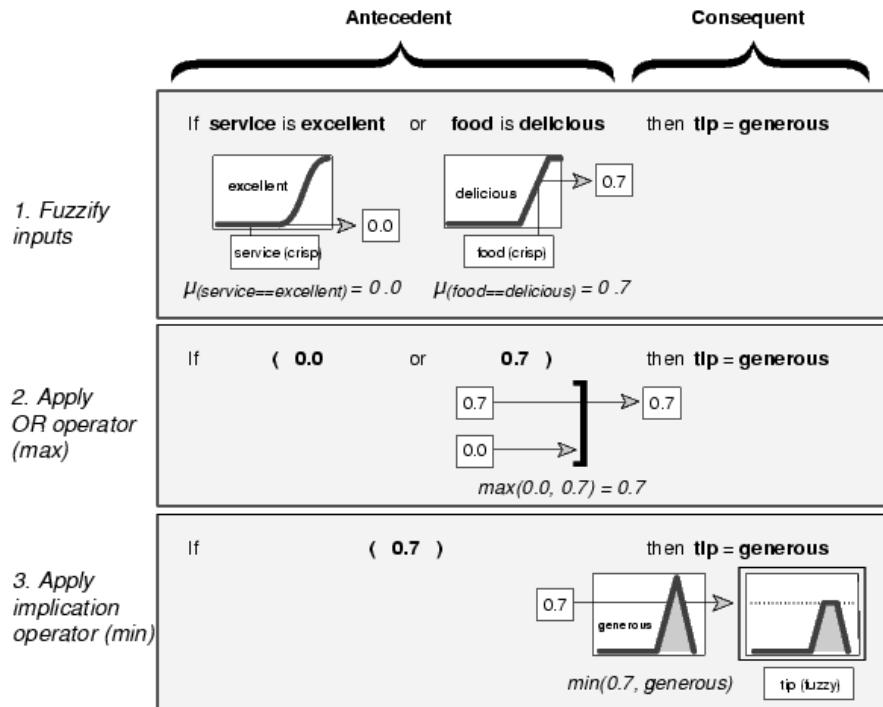


Figure 2.5: A graphical illustration for applying a fuzzy rule. This example shows the inference of the third rule in the tipping example. The input is fuzzified using the membership functions, then the OR (Union) operator is applied (using max), eventually the result is applied on the appropriate output membership function (tip is generous) by clipping (min).

Aggregation of the rule outputs. This is the process of unification of the outputs of all the rules. This is done by taking the membership functions of all of the rules' consequents, after the rule evaluation process, and combining them (usually by summation or via the max operator) into a single fuzzy set. (this is done separately for each output variable).

Defuzzification. This is the last step in the Mamdani inference system. Here the output fuzzy set is turned into a crisp number.

There are several defuzzification methods, the most popular one is the **centroid** technique. This is actually equivalent to finding the center of gravity (COG) of the fuzzy set A in the output range $[a,b]$.

$$COG = \frac{\int_a^b \mu_A(x) x dx}{\int_a^b \mu_A(x) dx}$$

Figure 2.7 shows the Mamdani inference system for the tipping problem (the flow of the inference is explained in Figure 2.6). The

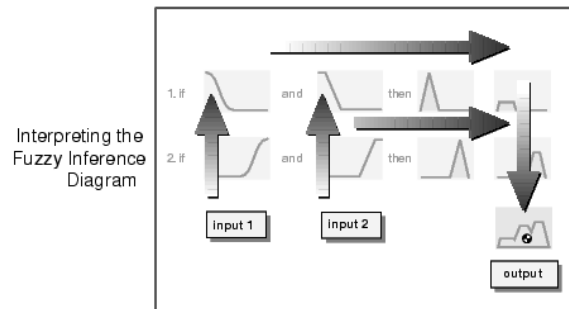


Figure 2.6: Fuzzy diagram inference flow. The flow proceeds up from the inputs in the lower left, then across each row, or rule, and then down the rule outputs (rule aggregation) to finish in the lower right. The black and white circle denotes the result of the defuzzification process

system uses min as the T-Norm operator and max as the S-Norm operator.

2. Sugeno-style inference.

This inference system is less popular, but more computationally effective. The idea is to use a single spike, a *singleton* as the membership function of the rule consequent. The method was first introduced by M. Sugeno in 1985.

A fuzzy singleton is a membership function that is equals 1 at a particular point and zero everywhere else. The location of the singleton is determined by a mathematical function of the (crisp) values of the input variables. The inference system is similar to that of Mamdani.

A rule in a Sugeno type inference system is of the form:

If x is A
 and y is B
 then z is $f(x, y)$

In the most commonly used *Zero-Order Sugeno Fuzzy Model* $f(x, y)$ is constant (i.e., the singletons' location is independent of the input parameters).

Figure 2.8 shows the fuzzy inference process for the tipping problem, using a zero-order Sugeno fuzzy model.

The step equivalent to defuzzification in this model is done by computing the *Weighted Average* (WA) of the output singletons:

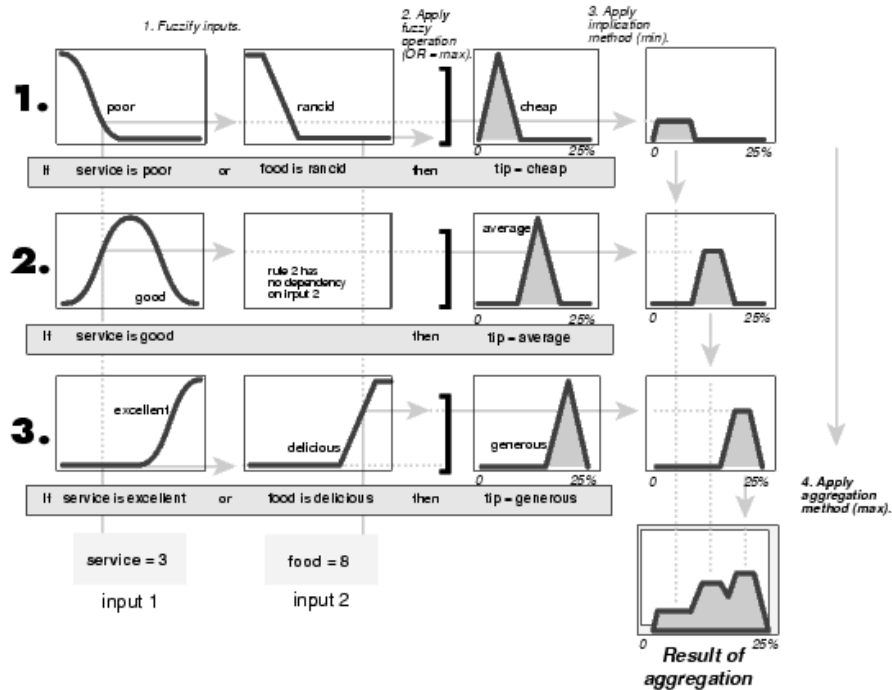


Figure 2.7: Solving the tipping problem via a Mamdani style fuzzy inference system. Each fuzzy rule inference is shown in a separate line. The aggregation of the rules' outputs is shown vertically and is done via a max operator.

In our tipping example illustrated on Figure 2.8 the output is calculated as:

$$WA = \frac{w(z_1) \cdot z_1 + w(z_2) \cdot z_2 + w(z_3) \cdot z_3}{w(z_1) + w(z_2) + w(z_3)}$$

$z_{1..3}$ are the output singletons' locations on the output variable range, $w(z_i)$ denotes the output level of the i 'th singleton.

2.2.3 Fuzzy Decision Trees

One predominant weakness of using traditional decision tree building process is the generation of sharp decision boundaries at every node within the tree. This can result in a classification error, and also in lower comprehensibility of the induced algorithm.

The fuzzy decision tree differs from traditional decision trees in two respects: It uses splitting criteria based on fuzzy restrictions, and its inference

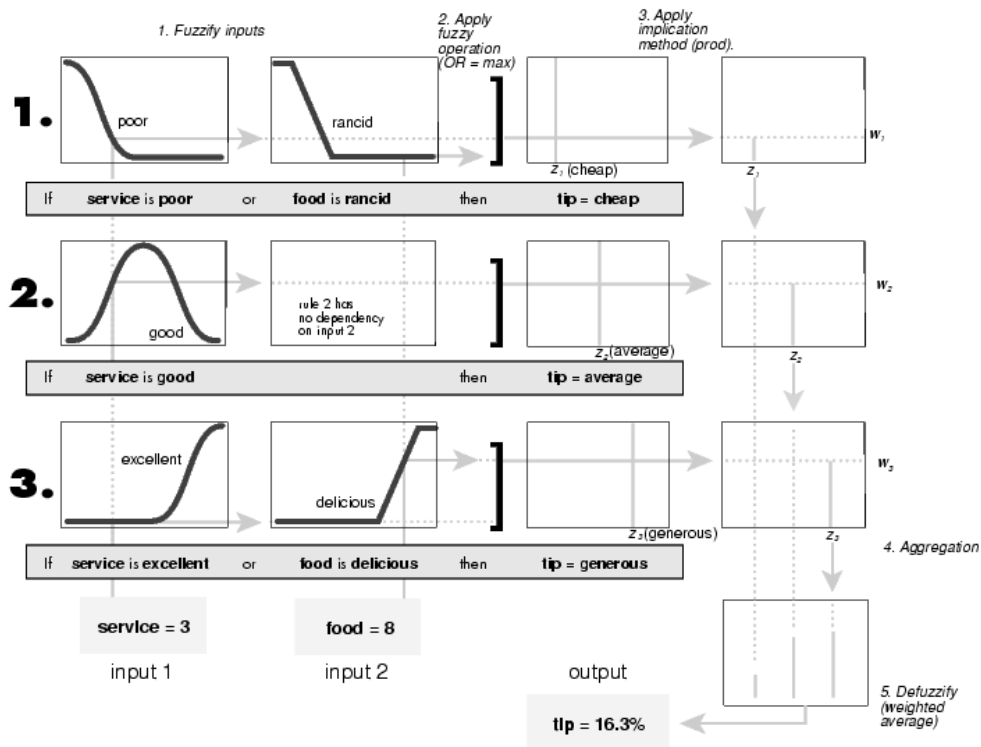


Figure 2.8: Zero Order Sugeno style fuzzy inference system for the tipping problem. The output membership functions are represented by singeltons, and the defuzzification stage is replaced by a weighted average of the outputs of the single rules.

procedures are different. Another basic difference that is induced by the former ones is that each example passes through more than one path.

Several works have been done in order to fuzzify the decision tree boundaries. Janikow [17] suggested a method for building the fuzzy tree, using fuzzy partitioning based on prior field knowledge. This method is commonly used in practice, and is explained hereafter.

Mathematical Notations

The mathematical notations used to describe fuzzy decision trees are given hereafter. For ease of understanding, examples from the tipping problem are given in parenthesis for some of the notations. Figure 2.9 is also used for illustration. Note that the fuzzy rules and membership functions in the figure, are different from those described in the former subsection.

1. The set of fuzzy variables is denoted by $V = \{V_1, V_2, ..V_n\}$ (e.g., $V_1 = Service, V_2 = Food$).

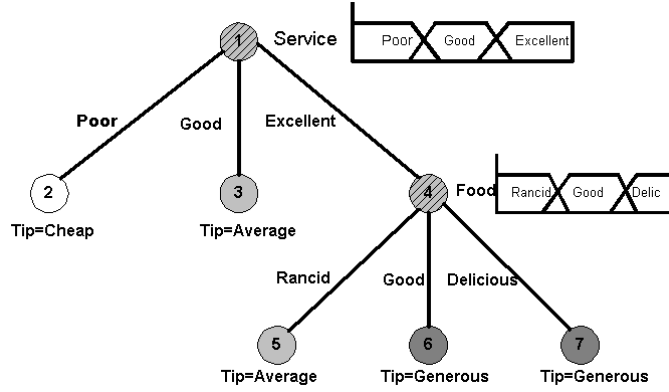


Figure 2.9: An example of a fuzzy decision tree for the tipping problem. The tree resembles a symbolic decision tree. The fuzzy sets for each tested attribute are illustrated to the right of the node

2. for each $V_i \in V$

- D_i denotes the set of fuzzy terms for variable V_i (e.g., $D_{service} = \{Poor, Good, Excellent\}$).
- $v_{i,p}$ denotes the fuzzy term p for the variable V_i (e.g., $v_{food,delicious}$).

3. The set of output fuzzy terms for the decision tree is denoted by D_c ($D_c = \{Cheap, Average, Generous\}$).

4. The set of training examples is $E = \{e^j : e^j = (e_1^j, \dots, e_n^j, y_j)\}$, where $e_i^j \in U_i$ is a numerical value describing the value of V_i for example j (e.g., $e_1^{service} = 9$) and y^j is the crisp value of the examples' classification.

5. for each node N of the fuzzy decision tree

- $F(N)$ denotes the set of fuzzy restrictions on the path leading to N (e.g., for node 7 in Figure. 2.9 $F(N) = \{Service = Excellent; Food = Delicious\}$).
- $V(N)$ is the set of attributes appearing on the path leading to N :

$$V(N) = \{V_i \mid \exists p [V_i \text{ is } v_{i,p}] \in F(N)\}$$

(e.g., for node 7, $V(N) = \{Service, Food\}$).

- $\chi(N) = \{\chi_j(N)\}$ is the set of memberships in N for all the training examples.
- $N|v_{i,p}$ denotes the particular child of node N created by using V_i to split N and following the edge $v_{i,p} \in D_i$ (e.g., $D|v_{service,good}$ will demonstrate node 3).

- $S_{V_i}(N)$ denotes the set of N 's children when $V_i \in (V - V(N))$ is used for the split ($S_{service}(1)$ = nodes 2, 3, 4).
- $P_k(N)$ denotes the example count for decision $v_{c,k} \in D_c$ in node N .
- $P(N)$ and $I(N)$ denote the total example count and information measure for node N , respectively.
- $G_i(N) = I(N) - I(S_{V_i}(N))$ denotes the information gain when using V_i to split N . $I(S_{V_i}(N))$ is the information content after splitting node N by attribute V_i of the fuzzy set.

6. The following fuzzy functions are defined:

- f_0 . This function is used to determine how a data input for a fuzzy variable V_i , satisfies the fuzzy restriction [V_i is $v_{i,j}$]. (i.e., applying the fuzzy membership function on the crisp input - see fig 2.5 step 1).
- f_1 . This function is used to combine levels of satisfaction of fuzzy restrictions of the conjunctive antecedent (the resulting value is often called *degree of fulfillment* or *satisfaction* of a rule. This is done by an intersection operator (usually either max or multiplication) of the membership values (see fig 2.5 step 2).
- f_2 . The function is used in order to propagate satisfaction of the antecedent to the consequent. This is often done by clipping or scaling the output membership function according to the value gained by f_1 (see fig 2.5 step 3).
- f_3 . This function is used for conflict resolution from multiple consequents. This is the rule aggregation stage, usually performed by summation or by finding the maximum of the formally scaled or clipped output membership functions.

Dealing with missing attribute values

It is common to use the same method, suggested by Quinlan [18] for symbolic decision trees, for treating missing attribute values in fuzzy decision trees. The idea is to evenly split an example into all children, if the feature to be tested is unknown, and then to normalize the information for that attribute by the percentage of examples with unknown value:

- denote $P(N)|(u_i \text{ unknown})$ as the total count of examples in the node N with unknown values for V_i
- define $f_0(e_j, [V_i \text{ is } v_{i,p}]) = (\frac{1}{|D_i|} \text{ if } e_j^i \text{ unknown}, [\mu_{v_{i,p}}(e_j^i) \text{ otherwise}])$

Tree Building

In ID3, a training example's membership in a partitioned set (and thus in a node) is binary. In Fuzzy logic, an example can be a member of a node to a certain membership value. An Example e_j has its membership $\chi_j(N)$ in a node calculated on the basis of the restrictions $F(N)$ along the path from the root to N . Matches to individual restrictions in $F(N)$ are evaluated with f_0 , and then combined via f_1 . For efficiency considerations, this evaluation can be implemented incrementally.

Algorithm 1. Fuzzy tree building algorithm

1. Start with the set of training examples E . If all the examples are regarded as being of equal importance, then $\chi(\text{root}) = 1$ (It is possible to use different weights for different examples here, in this case $\chi(\text{root}) = W$ where W is a vector of the examples weights (for more detail see [17]).
2. At any node N still to be expanded, compute the example counts $P(N)$:

$$P(N) = \sum_{k=1}^{|D_c|} P_k(N)$$

where

$$P_k(N) = \sum_{j=1}^{|E|} f_2(\chi_j(N), \mu_{v_c, k}(y_j))$$

Where $\chi_j(N)$ is the membership of example e_j in N . This membership is calculated incrementally with f_0 and f_1 from the restrictions $F(N)$.

3. Compute the standard information content:

$$I(N) = -\sum_{k=1}^{|D_c|} \left(\frac{P_k(N)}{P(N)}\right) \cdot \log \frac{P_k(N)}{P(N)}.$$
4. At each node, search the set of remaining attributes from $V - V(N)$ to split the node:
 - (a) Calculate $I(S_{V_i(N)})$, the weighted information content for V_i (adjusted for missing attributes):

$$I(S_{V_i(N)}) = \frac{P(N) - P(N|(e_i \text{ unknown}))}{P(N)} \cdot \frac{\sum_{v_{i,p} \in D_i} (P(N|v_{i,p}) \cdot I(N|v_{i,p}))}{\sum_{v_{i,p} \in D_i} P(N|v_{i,p})}$$

- (b) Select attribute V_i such that the information gain $G_i(N) = I(N) - I(S_{v_i}(N))$ is maximal (Can be adapted to the domain size). The expansion stops if the remaining examples at node N with $\chi_j(N) > 0$ have a unique classification, or when $V(N) = V$; (other criteria may be added).

5. split N into $|D_i|$ sub-nodes. Child $N|v_{i,p}$ gets examples with memberships denoted by $\chi(N|v_{i,p})$ calculated in the following way:

$$\chi_j(N|v_{i,p}) = f_1(f_0(e_j, v_{i,p}), \chi_j(N)).$$

If a child node contains no training examples, it is removed.

The tree building mechanism is similar to that of ID3. The only difference is due to the fact that a training example can be found in a node to any degree. Given that node memberships can be computed incrementally, computational complexities of the two algorithms are the same up to constants.

Inference of the assignment

After building the tree, a mechanism for aggregating the results from all the leaves that a testing (or pruning) example reaches is required. In [17] several methods that rely on the defuzzification stage are presented. In [19], the same author presents an alternative approach using ideas from Exemplar Learning for the inference.

The idea of exemplar learning is that selected training examples are retained, and a decision tree procedure returns the class assigned to the “closest” exemplar. These exemplars can also be generalized, allowing multiple exemplars to be used for making the decision.

Two types of parameters R_ℓ and r_ℓ are used to define the inference. y_j will denote the class assignment to a training example e_j . Taking all individual examples that fall into a leaf, we can compute these parameters as:

$$R_\ell = \sum_{e_j \in E} (\chi_e^\ell \cdot y^j), \quad r_\ell = \sum_{e_j \in E} (\chi_e^\ell)$$

Now the result for the classification of a new example with membership χ_e^ℓ at leaf ℓ , can be calculated via:

$$\delta = \frac{\sum_{\ell \in \text{Leaves}} f_2(\chi_e^\ell, R_\ell)}{\sum_{\ell \in \text{Leaves}} f_2(\chi_e^\ell, r_\ell)}$$

other variations that don't weigh individual exemplars by the number of examples they contain, or prefer using the output membership function of the class with highest accumulative memberships in the training set, are also suggested in [19].

Finding the fuzzy sets automatically

In a later article [20], the same author with M. Fajfer, suggest performing fuzzy partitioning to some of the attributes on the tree, by first considering

those attributes to consist of one fuzzy set that includes the whole attribute domain. Then an attempt is made to split this set to any two membership functions that will minimize the gain criteria. Along the path this attribute will be tested again, trying to split the fuzzy set that was chosen on the path, into two fuzzy sets in the same manner. After the tree is built, it is abundant and re-built using the sets that were found in the former stage.

Some researchers (e.g., [21, 22]) suggest fuzzifying the decision tree after it was constructed, by applying genetic algorithms or neural networks for adapting the membership functions.

Another approach was to use a CART-like algorithm for finding the crisp classifications of the continuous attribute, and then to fuzzify it using the attribute values of examples with adjacent values for that attribute [23].

X. Boyen and L. Wehenkel [24] state that attempts to find the fuzzy partitioning automatically were found to prefer the crisp partitions rather than the fuzzy ones, and use a criteria different from the gain criterion for building a binary fuzzy tree.

Chapter 3

Methods

3.1 General

The algorithmic section of this thesis consists of two basic parts:

1. *Feature extraction and examples generation.* This part deals with analyzing the recorded signals, determining and extracting the features that are candidates to be used by the learning algorithms, dividing the data into examples and matching the features to these examples.

This part is described in detail in Section 3.3 and is implemented on Matlab, over a GUI platform, initially developed by S.Papayan (Itamar-Medical).

2. *Learning from examples and patients classification* This part contains the algorithm for creating and testing the fuzzy decision trees and includes description of special characteristics we have added above known fuzzy decision trees creation algorithms, due to specific requirements of our problem. It is described in Section 3.4 and is implemented in C++, over a platform of conventional crisp decision trees written by D. Margaritis (Carnegie Mellon University).

The data used for the research was collected by Itamar-medical. It consists of two groups of patients. One for training purposes, the other for testing the results. The experimental setup is described in Section 3.2.

A preliminary algorithm for apnea detection from PAT and oximetry was based on 4 basic heuristic rules that were visually found by researchers in the field. On the basis of these rules and some statistical work, a deterministic algorithm was written. This algorithm and its results are described shortly in Section 4.1.

Later on, while experimenting with some methods from machine learning for solving the problem (rule-based expert systems, decision trees and fuzzy logic), we have written an algorithm that incorporates ideas from fuzzy logic

for solving the problem. This algorithm and its performance are described in Section 4.2.

Results of the offered thesis solution, which is based on FDT will be compared with those of the two algorithms above.

3.2 Experimental setup

The development and testing of the algorithms were performed on the basis of two separate sets of night studies (training and testing). The population tested consisted of people that were suspected to have OSAS and of healthy volunteers (i.e., with no complaints about daytime sleepiness or snoring). All the subjects went through a night study at a sleep laboratory. They were attached to a standard polysomnography device (Section 3.2.1), and to the PAT-Watch 100 (Section 3.2.2). The two recording devices were injected with a dedicated signal for synchronization.

There were 69 subjects on the validation group. forty-nine subjects were men and twenty were women. The mean age of the group \pm STD was 52 ± 13 years. The Averaged BMI (Body Mass Index) \pm STD was 29 ± 6 . The exclusion criteria for screening out patients were as follows: permanent pacemaker, non-sinus cardiac arrhythmia, peripheral vasculopathy (a disease of the blood vessels) or neuropathy (a functional disturbance or pathological changes in the peripheral nervous system), severe lung disease, finger deformity that precludes adequate sensor application, use of α -adrenergic receptor blockers, and alcohol or drug abuse during the last 3 years. The study protocol was approved by the ethics committee of Rambam Medical Center, and the subjects gave their written informed consent, prior to participation [5].

3.2.1 Polysomnography

The polysomnography recordings were performed using a computerized polysomnography system (Embla; Flaga medical) with the following channels: EEG (C3-A2 and O2-A1), EOG (for measuring right and left eye movements), chin electromyogram (measures the tension of the muscles of the chin), arterial oxygen saturation, nasal-oral airflow (thermistor), ECG, chest and abdominal wall motion, bilateral tibialis electromyogram (measure leg movements), body position and auxiliary synchronization channel.

The sleep stages were extracted automatically using the Somnologica software (Flaga medical). Two technicians (manual-scorers) manually scored arousals, apnea and hypopnea according to the polysomnography channels. Respiratory-related arousals were also scored in recordings that included the NAF channel (this channel measures the changes in air pressure close to the nose). The Wrist-PAT channels were hidden from the manual scorer.

3.2.2 PAT-Watch

The Watch PAT100 (WP100) (Itamar-medical; Caesaria, Israel) is an unattended ambulatory device. This device was designed to be easily mounted on a patients' arm, and is powered by an interior battery. Two probes mounted on the index and the ring fingers measure the PAT signal (see Section 2.1.3) and the Oxygen level respectively. An interior actigraph measures the movement of the device (hence of the arm). All signals are recorded at 100Hz.

The data is recorded on a Flash card. The card can be ejected from the device and read using the ZZZPAT software product (Itamar-medical).

3.2.3 ZZZPAT pre-processing

The data on the flash card consists of general information about the patient and about the study and of the recorded channels. This data is copied to a PC's Hard drive, together with the data supplied from the polysomnography recordings. The recorded channels from the Watch-PAT are read, the PAT signal is reconstructed and filtered. The signals are truncated to the actual test time.

The data from the polysomnography device is converted to the same format, the data from both devices is synchronized using the synchronization channel. This information, along with the additional information from both the manual and the automatic analysis of the polysomnography recordings is saved in a binary format to allow for further processing.

3.2.4 PAT Amplitude and Heart Rate Extraction

As stated before, the PAT signal comprises of changes in the arterial volume. Hence, the cardiac activity is prominent in the PAT signal.

The PAT signal is first filtered by a high pass filter, in order to avoid low frequency components resulting from slow changes in blood pressure or in body temperature. Then a dedicated algorithm written by I. Dvir, S. Papyan and N. Yenokian (Itamar-medical) is incorporated for detecting the heart beat equivalents in the signal. Each such heart beat pattern is termed *a complex*. Irregular complexes are declared as artifacts. The PAT Amplitude of a complex is calculated as the difference between a local maximum and a local minimum in that complex. The heart rate is calculated as the distance from the maximum point of the complex, to the maximum point on the consequent one.

The two upper rows in Figure 3.1 show the filtered PAT channel, and the complex detection in a time segment that includes an artifact.

Oximeter

Due to the fact that the oximeter has a different intrinsic clock than the Watch-PAT, it is re-sampled to fit a 1Hz frequency according to the Watch-PAT clock.

Summary

The outputs of this stage are a PAT amplitude array, a heart rate array and an array containing the time location of the samples of the two former ones.

The input to our feature extraction algorithms will be those three arrays, an array containing the valid segments of the PAT and of the oximetry channels, the actigraphy channel, the resampled oximetry channel and the sleep stages from Embla. In the examples classification stage the scoring of apnea, hypopnea and RERAs is also used.

3.3 Feature Extraction

3.3.1 Signals Pre-Processing

Amplitude and HR processing

At this stage re-sampling, noise reduction and smoothing of the PAT Amplitude lead and of the heart rate lead are performed. In our suggested algorithm, this is done by first applying a median filter of window size of 3 samples (each sample represents one complex) in order to reduce spikes that result from either a misidentification of a complex, an identification of a single complex as two separate ones, or an arrhythmia. (See Figure 3.1). Afterwards the leads are re-sampled to 1-Hz frequency and then smoothed by a low-pass filter, to remove rhythmic components resulting from the effect of respiration on the arterial volume and heart rate, and from local artifacts.

Actigraph energy extraction

The energy from the actigraph channel is generated in order to be used for extraction of actigraph movements (Section 3.3.2).

The Actigraph signal is filtered with an anti-aliasing filter and decimated by factor 10 (from 100Hz to 10Hz). The mean value of the decimated signal is calculated and removed. From the resulting signal, the “momentary energy” is calculated for each sample by squaring the result. The momentary energy is averaged every 5 samples to achieve the actigraph energy lead at 2Hz. Figure 3.5 shows an example of the actigraph channel energy extracted from the signal.

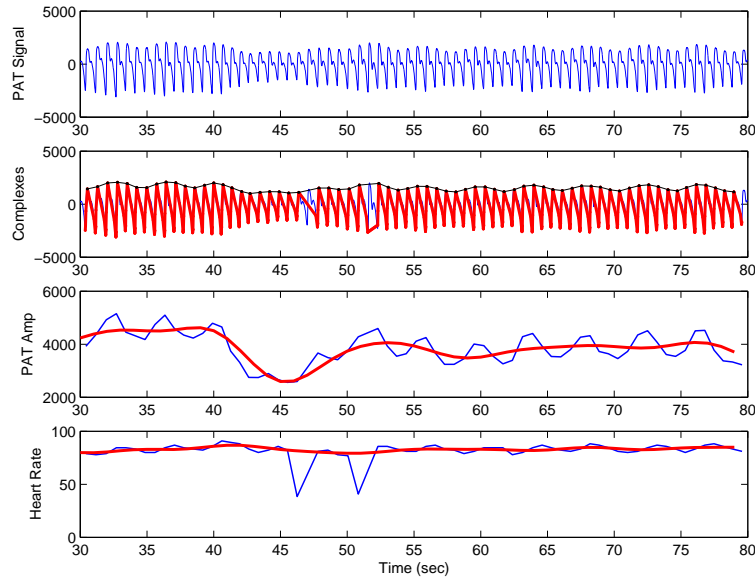


Figure 3.1: PAT amplitude and heart rate noise reduction and filtering to avoid miss-identified complexes and artifacts. The first two plots show the PAT signal and the identified complexes (each black dot on the upper line of the second plot illustrates the maximal signal point of each complex). The blue (thin)\red (thick) lines on the third and fourth plots show the original\smoothed amplitude and heart rate respectively.

3.3.2 Extracting Physiological Attributes

PAT amplitude and Heart rate change events

The PAT Amplitude and the heart rate change events are crucial for detecting arousals, which are a marker of a possible respiratory event. The increased sympathetic activation is linked with a *decrease* in the PAT amplitude and an *increase* in the heart rate.

Input Parameters The algorithm for detecting the change events requires as input the amplitude (or heart rate) channel (smoothed and sampled at 1Hz), and the following parameters:

- Minimal and maximal allowed durations.
 - Minimal amplitude event duration = 4sec.
 - Maximal amplitude event duration = 40sec.
 - Minimal heart rate event duration = 4sec.
 - Maximal heart rate event duration = 40sec.
- Minimal and maximal change percent limits.

- Amplitude minimal change percent = 15%.
- Amplitude maximal change percent = 100%.
- Heart rate minimal change percent = 5%.
- Heart rate maximal change percent = 100%.
- Minimal and maximal change slopes (Change percent divided by duration).
 - Minimal Amplitude change slope 0.001%/sec.
 - Maximal Amplitude change slope 20%/sec.
 - Minimal heart rate change slope 0.001%/sec.
 - Maximal heart rate change slope 15%/sec.
- Change event direction (1 for increase -1 for decrease)
 - Amplitude change event direction = -1.
 - Heart rate change event direction = 1.
- For overlapping events: Percentage of change for choosing to maintain the shorter event = 85% for both amplitude and heart rate (3).
- For consequent events (4): Maximal time gap between events for unification = 10 sec.
Maximal percentage of change for unification = 25%.

Output Parameters The output is a structure of the change events, containing the following fields:

- Start time.
- End time.
- Duration.
- Change Percent.
- Slope.
- The baseline level of the signal (i.e., value of the signal at start time).

Algorithm 2. Change events detection algorithm

1. *Find candidates for start and end points of events:*
 - *If seeking for decrease event:*

- *Start candidate: Maximal slope change and signal is decreasing or local maxima.*
 - *End Candidate: local minima.*
 - *If seeking for an increase event: find candidates by opposite logic.*
2. *For each start candidate:*
 - *Find the next end candidate.*
 - *Make sure the end candidate is within the min/max duration limits.*
 - *Check if it meets the change percent and slope limits.*
 - *If all conditions are met, write the event defined by the start and end time into a temporary event database.*
 3. *For each pair of events, if one is contained within another, and if the change percent of one is at least a constant ratio of the other's, remove the longer one, else, remove the shorter one.*
 4. *Unify close events if the distance between the end time of the first and the start time of the second is less than a constant and the difference in the signal's value between the start time of the second event and the end time of the first event, is less than a constant ratio of the signal's difference in the unified event.*
 5. *Copy the remaining events into a new event database.*

Figure 3.2 illustrates PAT amplitude and heart rate events detection.

Events of change in the oximetry level

An obstructive apnea usually results in a decrease in the oximetry level (of at least 3-4%) called a desaturation. The apnea, which usually ends with an arousal or a micro-arousal causes the renewal of respiration, that results in a re-saturation (oxygen saturation increase). Sometimes the decrease in saturation level is very slow and hardly notable, and yet a re-saturation will appear at the end of the process (this is assumed to be a common case for RERA's). Therefore it is important to find both the desaturations and the re-saturations.

There is a certain time delay between the physiological phenomena causing the change in oximetry and its appearance in the finger measurements. This is due to several reasons: The time it takes the air to diffuse into the blood system, the time it takes the blood to flow from the heart to the finger, and some smoothing filters present in the oximeter. It was empirically possible to note a time delay of about 15sec from the start/end of respiration to the desaturation/re-saturation. However, this time delay is not consistent.

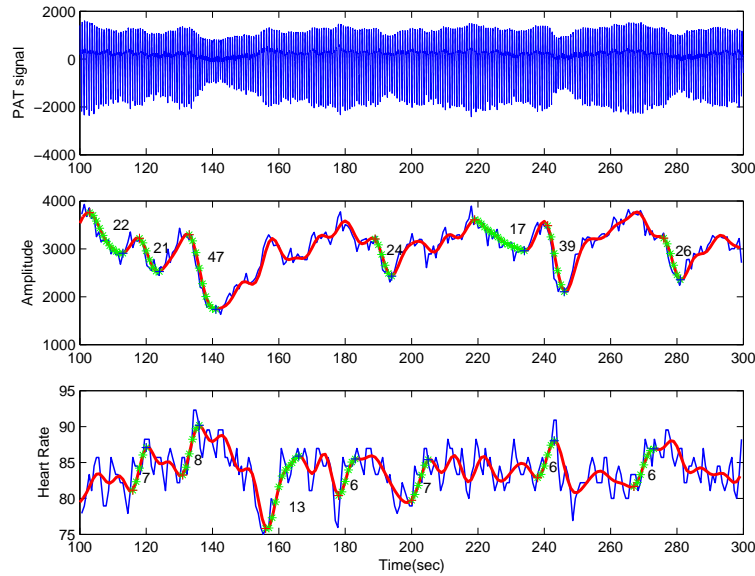


Figure 3.2: A demonstration of detecting change events in the amplitude and the HR extracted from the PAT signal. The first plot shows the PAT signal. The second plot shows the PAT amplitude (raw and smoothed) and the detected events (green), the numbers next to the event indicate the decrease in %. The third plot shows the HR events in a similar way (the numbers indicate the increase in %).

The input variable to the algorithm of detecting desaturations is the saturation channel (also samples at 1Hz). Resaturations are detected in the same manner, but the input channel is inverted (i.e., the input channel is 100%-Saturation). The following set of parameters is used:

- Window size in seconds is 30 seconds and 15 seconds for desaturations and for resaturations respectively.
- Minimal and Maximal limits of the change percentage are 2%-18% for both desaturations and resaturations. (Here the percentage is the percentage of bound red cells, and not the percentage with respect to the baseline level).

The algorithm for detecting desaturation events is as follows:

Algorithm 3. Desaturation detection algorithm

1. Traverse the saturation signal with a moving window.
2. At each window, find the maximum and minimum points.
3. If the maximal value minus the minimal value is between reasonable limits, write the time indices between the time locations of the maximum and minimum points into an array.

4. Traverse the array of indices, each group of consecutive indices forms an event and is written into the event database of the desaturations.

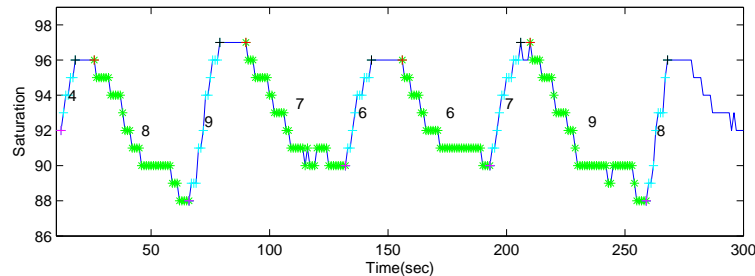


Figure 3.3: A demonstration of detecting change events in the Oximetry channel. Desaturation events are marked in green asterisks, re-saturation are marked with cyan plus. The decrease/increase in oximetry level in % is written next to each event

Figure 3.3 demonstrates oximetry channel change events.

Movements in the actigraph channel

Body movements are also a marker of a possible arousal. The movements are extracted from the energy signal, generated as in Section 3.3.1.

The parameters required for the algorithm are:

- Parameters for calculating the basic BG level:
 - The actigraph energy sampling frequency = 2Hz.
 - Size of window for background level calculations = $60 \cdot$ sampling frequency (1 minute).
 - Maximal energy value to be taken into account when calculating the background level = 50000 counts.
- Parameters for adjusting the background level according to respiratory modulations:
 - RespModThresh4LocalMaxima = 700 counts;
 - RespModFactor4LocalMaxima = 2;
 - Number of histogram bins = 13.
 - Maximal Duration for time time difference calculations = 6.5sec.
 - Minimal time period for respiratory modulations = 3sec.
 - Maximal time period for respiratory modulations = 5.5sec.

- RatioThresh = 2;
- Factor4BGWResp = 2;
- Parameters for declaring a movement:
 - Maximum duration = 30sec.
 - Entrance threshold = 2200 counts.
 - Exit threshold = 1800 counts.
 - Minimal time difference between two movements = 4sec.

The output is an event database containing the following events' information:

- Duration
- Energy Area between energy plot and background level in the time boundaries of the event (converted into $\log(\text{energy})$ units).
- Maximal Energy minus background level (in log unites).

Algorithm 4. Movements detection algorithm

1. Calculate the background level each 1 minute as follows:

- (a) Take all samples below an energy threshold.
- (b) Perform a median filter.

(the incentive for using the threshold is to prevent treating samples that result from long significant movements as part of the background level).

2. A histogram of the distances between peaks in the energy is used in order to overcome problems of respiratory modulations in the actigraph energy (if such modulation is detected, the background level is increased).

In order to detect these modulations:

- (a) Detect local maxima on the energy channel.
- (b) Remove local maxima that are too low compared with the neighboring maxima.
- (c) For each local maximum calculate the distance from it to the consequent maxima points (keep doing it up to a maximal time difference).
- (d) Build a histogram from all these distances. while creating the histogram, also calculate the mean value at each histogram bin.
- (e) Find the bin with maximum frequency on the histogram.

- (f) Check if it is between limits that can fit the respiration time period.
- (g) if it is, check if it's frequency is at least twice of the average frequency of the other bins. If so, than periodicity is assumed.
- (h) If periodicity is assumed, the background level is updated to be twice that of the mean value of the maxima points in the maximal bin. If periodicity is not found, the old value of the background level is maintained.

3. Use two thresholds (an entry threshold and a lower exit threshold) to declare the actigraph movements.

4. unify close events.

Figure 3.4 shows a segment that contains respiratory modulations in the actigraph channel. The background level before and after the histogram calculations are shown.

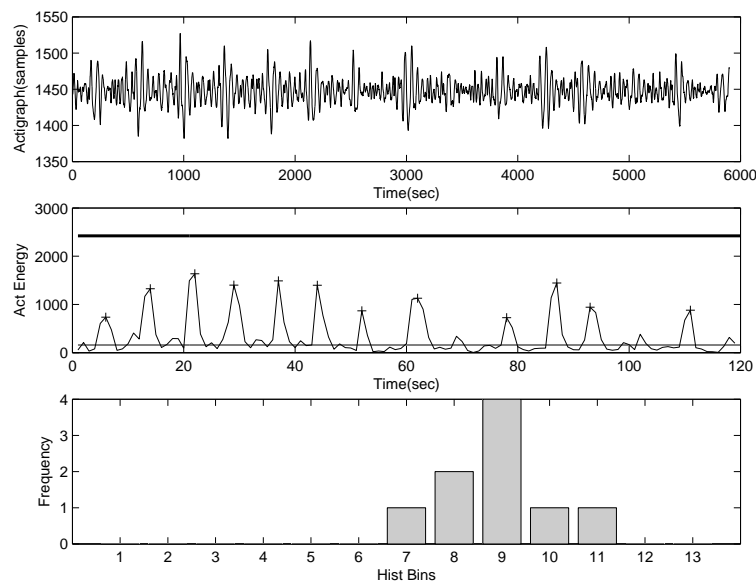


Figure 3.4: The figure demonstrates detection of the background level on the actigraph channel in the presence of respiratory modulations. The first plot shows a segment of the actigraph signal. The second plot shows the energy of the signal as calculated in 3.3.1. The thin straight line shows the background level calculated from the median of the samples. The thick line shows the calculated background level, after taking the respiratory modulations into account. The third plot shows the distances histogram.

An Example of an actigraph movements detection is shown in figure 3.5.

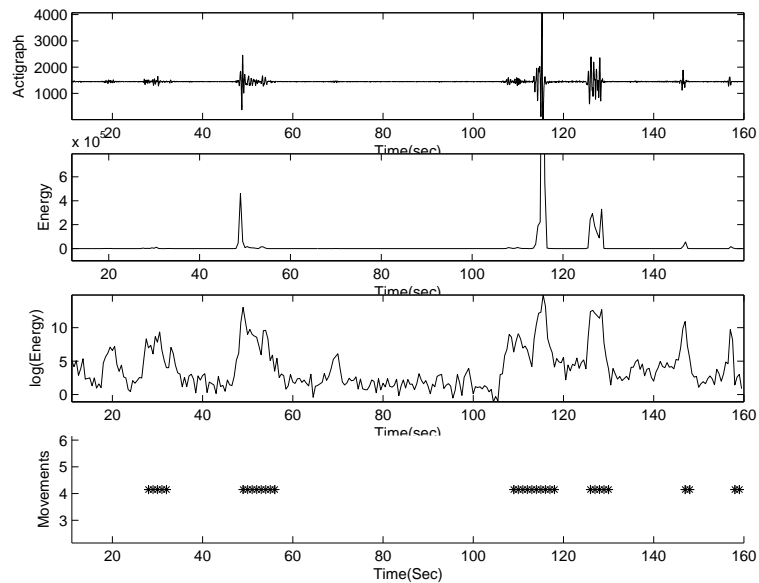


Figure 3.5: Actigraph movement detection example. The first plot shows the actigraph channel. The second plot shows the energy in that section. The third plot shows the $\log(\text{energy})$ in order to emphasize differences in the energy at lower values. The fourth plot shows the location of the detected movement events.

Local re-saturation index

The changes in oximetry generally give a good estimation of the OSA severity of a patient: a severe patient will also have a high desaturations index and vice versa. A popular measure of these changes is called the ODI (Oxygen Desaturation Index). The ODI equals the number of desaturations with at least 4% decrease in oxygen level per hour.

We empirically found that there is a better correlation between the number of 3% and above re-saturations and the RDI index than the ODI index. It is also suspected that the PAT and heart rate response of a severe patient during an apnea is different from that of a mild patient. Furthermore, the probability of an apnea to appear in a time segment that has high local re-saturation index is different from the probability for it to occur in a time area that lacks saturation changes, therefore it was desired to use this criteria as an attribute. Moreover, it was notable that the re-saturation index can vary significantly in different time segments of the same subject. Hence, a local re-saturation index was calculated according to the following algorithm:

The input variables are the resaturation time locations and the valid sleep time locations.

The parameters for the algorithm are:

- The desired window size = 20 minutes

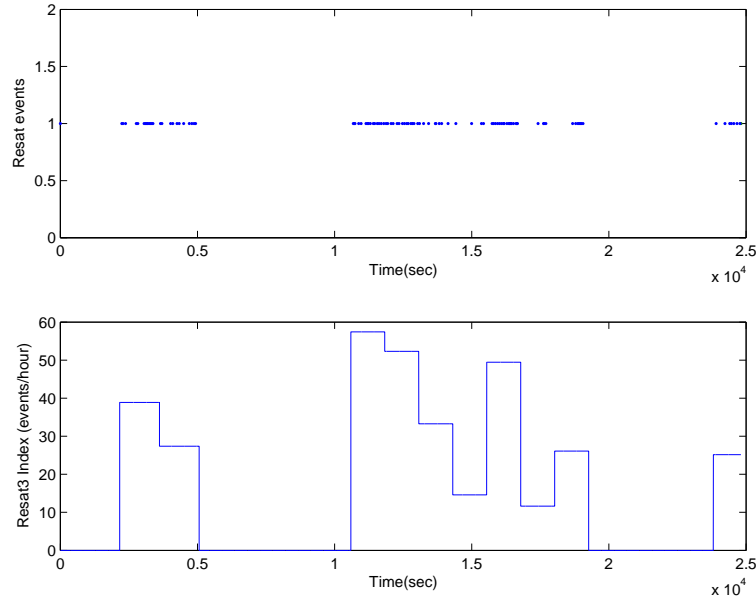


Figure 3.6: Local re-saturation index calculations. The first plot marks the re-saturation events location (that are of at least 3%), the second shows the calculated local re-saturation index.

- Maximal time gap for segments unification = 5 min.
- Maximal duration for one segment = 25 minutes.
- Minimal duration limits = 15 minutes.

Algorithm 5. Local resaturation index algorithm

1. *Divide the test into several non-overlapping segments. Since it is desired to calculate the index only for valid sleep periods, the division into segments (of about 20 minutes) is performed in the following manner:*
 - *From the array of valid sleep time indices, find start and end time of each valid-sleep time segment (using the locations in which the difference between two consequent indices is greater than 1).*
 - *Unify segments that the time gap between them is smaller than a constant.*
 - *If the segment's length is more than the maximal duration for one segment, divide it into n equally sized segments where*

$$n = \left\lfloor \frac{(\text{segment length})}{\text{Desired window size}} \right\rfloor$$

- *Else, if the segment length is between time limits (between 15 minutes and 25 minutes) treat it as one segment.*
 - *Else if the segment is too short, unify it with the next one and repeat the considerations for the unified segment.*
2. *Calculate the local index as the number of resaturations divided by the length of the segment in hours.*

Figure 3.6 shows an example of the calculated index for one patient.

Automatic events existence.

The physiological changes that can imply on the possibility of having a respiratory event are not always existent in its neighborhood, hence, some of the attributes will have missing values. However, the fact that an event doesn't exist or could not be measured is meaningful to the classification process and should not be bypassed using conventional solutions for missing attributes. Hence, attributes indicating whether an automatic event was detected or not are added to the attributes list of the algorithm. The following events' existence attributes are used:

- Amplitude change exists.
- Heart rate change exists.
- Desaturation exists.
- Re-saturation exists.
- Actigraph movement exists.

Distance between automatic events.

The time delays between different physiological events can also be a useful marker for the existence of a respiratory event or an arousal. Empirical research implies that synchronization between the nadir point (point of minimal value) of the amplitude change and the maximal point of the heart rate change increases the possibility for an arousal. It was also suggested that a desaturation that follows an actigraph movement is probably caused due to the movement itself and not due to a genuine change in the blood saturation level. Therefore, several distances between automatic events are suggested as possible attributes for the algorithm:

- Distance between end of amplitude change event and end of heart rate change event.

- End of amplitude change and end of desaturation.
- End of amplitude change and beginning of re-saturation.
- End of re-saturation and beginning of desaturation.
- End of amplitude change and beginning of movement.
- Beginning of movement and beginning of desaturation.

PAT Amplitude periodicity.

The attenuation of the PAT amplitude may be periodic in recording segments of patients that suffer from OSAS or from PLM. The typical time period of these two syndromes is different with the PLM ranging around $17 \pm 5sec$ and the OSAS ranging about $40 \pm 20sec$. Two attributes are used to address this issue:

1. Periodicity existence.
2. The time period (if the former attribute is true).

Input Parameters The parameters used in the algorithm of periodicity detection are:

- *Window size* = 75 seconds (on each side)
- Minimal ratio for declaring periodicity = 40%.
- Minimal and maximal time periods for respiratory modulations = 3.5 sec and 5.5 sec.

For each example location (to understand the time location of examples see Section 3.3.3). The periodicity detection is performed in the following manner:

Algorithm 6. Algorithm for periodicity detection

1. *Take a time segment of $\pm Window$ size around the location of the example and remove its DC level (mean).*
2. *Zero-pad the edges of the result in order to gain more samples for the PSD calculations (to allow the frequency response to be of sufficient resolution).*
3. *Multiply the result by a Hanning window (to avoid distortions caused by the finite length of the segment).*
4. *Calculate the power spectral density (PSD) of the result.*

5. Find the maximal point of the PSD distribution.
6. If the density in the area of the maximal point, divided by the sum of all densities is higher than a threshold, consider the segment periodic.
7. If the segment is periodic, check if the time period (extracted from the location of the maximal point) not too small or too large (with respect to the size of the segment). If so, declare that the segment is periodic, save the time period.

REM

Some OSAS patients have a tendency to have more apnea during REM sleep (REM-related OSAS patients). Moreover, since REM sleep resembles wakefulness in the EEG and EOG activity, there may be a difference in the expected PAT response for an arousal. Due to these reasons, we use a binary attribute REM/NREM to the attributes list.

3.3.3 Defining negative and positive examples.

In this section we shall discuss the issue of generating examples for the learning process. In many cases, once the features are extracted, this problem becomes trivial (e.g., in a problem of deciding whether a patient suffers from a certain illness or not, each patient exclusively defines one example and the task of correlating attributes such as body temperature, blood pressure etc). In our case we will define a positive example as an example that relates to an apnea event marked by the scorer, and a negative example as an example that does not relate to a true apnea event.

Our aim is to estimate the number of apnea events per hour of sleep per patient. For the scorer, it is possible to mark the whole duration of an apnea by using channels from the polysomnography. This information is not present in the recordings that we analyze. We can only look for markers indicating that an apnea event have occurred near a certain point in time. Therefore, we should try to find a marker for an apnea event in the vicinity of its true occurrence.

Furthermore, there are inconsistent time delays between the physiological phenomena that we try to estimate (respiratory disturbances) and the ones that actually measure (sympathetic activation, oximetry and actigraphy). These inconsistent time delays also indicate that a special solution is required in this case.

First we need to define the “drive” for generating an example. One possible way for doing this, could be to define the drive as the existence of a manual scorer event and choose the relevant attributes as the ones closest to its location. Apart from leaving us with the question of how to define negative

examples, it would also be impossible to use the same procedure for defining the examples for testing new subjects, as we will not have the manually scored data for them.

Another way could be to divide each sleep study into constant window segment with/without overlapping. Then to classify the examples according to the existence of the manually scored event in their vicinity. In this case, the matching between the automatic events and the examples, can vary significantly when choosing a different window size, or if starting the division process at a slightly later time. Errors in the matching process are likely to appear, especially when the manual or automatic events' locations are close to the boundaries of the window.

Determining the examples' time locations.

When a manual scorer marks an apnea event, he uses signals as the abdominal and respiratory belts to determine the exact duration and time location of the apnea. However, when we estimate the existence of an apnea by the PAT, oximetry and actigraph, this information is lacking. From this data, it is only possible to estimate the timing of the arousal or micro-arousal that appears as the end of the apnea event.

The chosen solution is based on the idea of using the events that were automatically found in the feature extraction stage (the "Signal events") in order to approximate possible time locations for the (markers of) the manually scored events (the target events). These possible time locations are used for determining the examples time locations.

Since the physiological markers that can be seen in the PAT amplitude, heart rate, movement and re-saturation are caused by the arousal, marking the end of the respiratory event, we will use the *end time* of the manually scored events as the point representing their existence.

After the signal events (i.e., amplitude and heart rate changes, re-saturation, desaturation, and actigraph movement) were found, we try to define the best point that will be used as a marker for these events. Usually start time or end time of the event is selected, but other locations such as time location of maximal point in the actigraph energy can be used as well.

Using field knowledge, visual inspection of the recordings, and distance histograms, we came to the conclusion that the following markers and delays are best used for approximating the possible time locations of the arousals:

- Amplitude change event: End time used as a marker. No delay is required.
- Heart rate change event: End time used as a marker. No delay is required.

- Desaturation: End time used as a marker. Delay time -10sec.
- Re-saturation: Start time used as a marker. Delay time -10sec.
- Actigraph movement: Point of maximal energy used as a marker. No delay is required.

Each marker is given a weight that is affected by how accurately we expect it to predict the location of the manually scored events.

The parameter used for matching the signal events to the target events is the maximal allowed time gap between the two = 25sec.

Algorithm 7. Algorithm for matching signal events with target events

1. Initialize an array of the night study length (in seconds).
2. For each automatic event, the expected weight is added to the cell in the array corresponding to the (marker + delay time) of the event.
3. Traverse the array. Starting from the first non-zero location, the center of gravity (COG) of the array in a time window is found. This COG will be an example's location.
4. Repeat until end of the sleep study is reached.

Finding the examples' duration.

Since the target of the learning process in our case is the respiratory disturbance index (number of events divided by time), we need to assign each example with its duration. The examples' duration must cover the whole valid sleep time of the test, with no overlapping time segments.

The process of finding the examples duration is as follows:

Algorithm 8. Algorithm for finding the examples' duration

1. Divide the sleep study into segments of consecutive valid sleep times.
2. For each segment:
 - The start time of the first example in the segment is simply the start time of the segment.
 - If the end time of the segment precedes the location of the next example, then the end time of the example is the end time of the segment. Else, the end time is the average time between the location of the current example and the location of the next example. The next example's start time is $StartTime(n+1) = EndTime(n) + 1$.

- If a segment contains no examples, create one in the middle of the segment. Its start and end times will overlap the start and end time of the segment.

3. Repeat the process until the end of the test is reached.

4. Compute the duration of each example ($= \text{EndTime} - \text{StartTime} + 1$).

An example of finding examples' locations and start/end times can be found in Figure. 3.7.

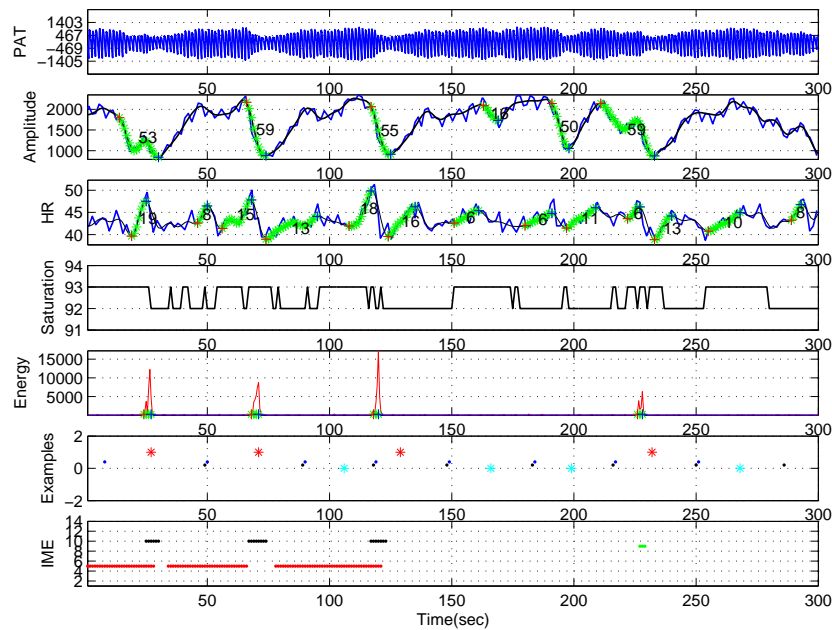


Figure 3.7: A demonstration of finding examples locations. The first plot shows the PAT signal. The second and third plots show the amplitude and heart rate leads respectively. The events extracted from these leads are shown in green. The fourth plot shows the saturation signal. No desaturations or re-saturations were detected in this segment. The fifth plot shows the actigraph energy and movement events (green). The sixth plot shows the extracted examples locations. Positive examples locations are marked with a red asterisk. Negative examples are marked with a cyan asterisk. a black dot marks the start time of an example, the blue dot marks the end time of an example. The last plot shows the manually scored events. Respiratory events are marked with red dots, RERAs are marked in green and black dots mark arousal events.

Matching the attributes to the examples.

The events closest to the examples' locations are chosen to describe the example. This is done using a nearest neighbor matching algorithm. The parameter

for the algorithm is the maximal window size = 25sec.

For each automatic event:

Algorithm 9. Matching attributes to examples

1. Write an array of the signal events' markers+delays.
2. Write an array of the target events' markers.
3. Write a matrix of the distances between the two arrays (since the arrays are sorted, there is no need to calculate all the elements in the matrix).
4. Find the minimal distance in the matrix.
5. If it is small enough, match the corresponding automatic event to that example, remove the pair (i.e., whole row and whole column elements) from the matrix.
6. Repeat until the remaining minimal distance is too large.

Finding the example's classification.

This task is performed in a similar manner as the previous one. The two arrays to be matched are the examples locations and the manually scored events markers.

3.4 Learning From Examples.

The learning algorithm builds the fuzzy decision tree from the examples listed in the text files that were generated by the Matlab application described in the former section.

A C++ program that builds crisp decision trees according to C4.5 was used as a basis for our program. We have changed the program to allow reading the additional information required for the fuzzy features and to allow fuzzy gain calculations and fuzzy partitioning of the tree, added special features such as the use of attribute hierarchy, changed the pruning criteria, and the accuracy calculations to fit both the fuzziness and the fact that we care more for the RDI per patient than for correct classification per example. We have also changed the testing stage to support the fuzziness and have also added the possibility to save the generated tree and test it on a separate run of the program.

3.4.1 Algorithm flow

Algorithm 10. Learning algorithm flow

1. *Read the program parameters.*
2. *Read the Training data file.*
3. *Partition the training file data into 3 sets of training, pruning and testing examples, according to the input parameters.*
4. *Create the preliminary decision tree (see Section 3.4.3).*
5. *If pruning percentage is greater than zero than post-prune the decision tree (see Section 3.4.5).*
6. *If the test percentage is greater than zero, test the tree's performance on the testing examples chosen from the training file. Accuracy testing is described in Section 3.4.6.*
7. *Save the tree in a binary file format (see Section 3.4.7).*

3.4.2 Attribute hierarchy

Another unique feature in our solution is the use of attributes hierarchy. We define preliminary attributes as attributes that must precede other attributes, before these are tested.

This mechanism is used for addressing two problems:

1. *Dealing with missing attributes.*
2. *Forcing separate rule generation for certain cases.*

Let us first describe the **mechanism of preliminary attributes**:

Each attribute V_i is attached with a field defining the attributes that must precede it (on the path leading to a node that tests V_i). This information is used in the process of finding the best attribute to split a node:

Algorithm 11. finding the best attribute to split the node

1. *Find the attributes ($V_i \notin F(N)$) that will best split the node (according to the maximal gain criteria, regardless of preliminary attributes). Maintain the information gain calculated for all of the attributes.*
2. *If the selected attribute has preliminary attributes assigned to it, check if these attributes were already used (on the path leading to the node).*

3. If so, choose this attribute for splitting the node. If not, find the preliminary attribute with maximal gain amongst the preliminary attributes that were not previously used yet.

$$V_{max\ gain} = \{V_i | (Gain(V_i) \text{ is maximal} \ \& \ V_i \notin F(N))\}$$

Note that if a preliminary attribute itself contains a list of preliminary attributes that were not chosen yet, the process will be repeated until no such attribute is found (beware of loop-linking the preliminary attributes).

Dealing with missing attributes.

Examples with missing features, resulting from missing automatic events, are likely to be present. As mentioned in Section 3.3, the fact that an example could not be matched with an automatic event is significant to the decision process.

Our solution to the problem of dealing with missing attributes of that nature, relies on using attributes that indicate whether an automatic event exists or not in the example, prior to using attributes that are induced by that event (e.g., the attribute “amplitude actigraph distance” will have two preliminary attributes: amplitude event existence and actigraph movement existence).

This process verifies that the existence of a measured value will always be tested prior to testing the attribute’s value.

Forcing separate rule generation in certain cases.

As discussed in Section 3.3.2, it is reasonable to think that the set of rules, membership functions and classification probabilities will differ in the case of a severe patient, from that of a mild patient.

In the problem of detecting respiratory events, the fact that a patient has low or high re-saturation index (i.e., that the patient has a mild or severe respiratory disturbance), will not necessarily appear on the root node of the tree, since its significance depends on other tests that will follow. To force the use of local resaturation index at the root node, we define it as a preliminary attribute to all of the attributes, hence we gain three sub-trees for each of the linguistic terms of that variable.

3.4.3 Preliminary tree building

For building the preliminary fuzzy tree (the tree before pruning) we follow the mechanism described in the background chapter 2.2.3:

Algorithm 12. Algorithm for building the preliminary fuzzy decision tree

1. Create the root node.
2. Set weights of all training examples to one ($\chi_j(\text{root}) = 1$).
3. Select the attribute V_i that will best split the node (see Section 3.4.2).
4. Create the children nodes according to the number of linguistic values of the chosen attribute.
5. Initialize each child node:
 - (a) Let each child contain the examples membership of the parent.
 - (b) Keep information about the linguistic value that leads to the node.
 - (c) Keep link to the parent node.
6. For each child:
 - (a) If the chosen attribute is fuzzy then, for each example that is a member of the child node:
 - i. Calculate the membership value ($\mu_{v_i,p}(u_j^i)$) of the attributes' value for the example to the fuzzy term (using f_0).
 - ii. Calculate the membership of the example to the node:

$$f_1(\mu_{v_i,p}(u_j^i), \chi_j(\text{parent node}))$$
 - iii. Update the number of examples in the node.
 - iv. If this number is significant create a sub-tree for that node (by performing steps 3 - 6).
 - (b) If the chosen attribute is binary a similar process as in 6a is performed. The membership values are either copied to the child node or reduced to zero, according to the attribute's value and the child node.

3.4.4 Considerations of computational complexity and memory allocations.

Due to the distributivity property of the T-norm (intersection) operator, calculating the membership of a training example to a node ($\chi_j(N)$), can be done in one of two ways:

1. Starting from the root node, compute f_0 of the example for all the nodes on the path. intersect the results using f_1

2. While building the tree, maintain the membership value of each training example in the node. When calculating the membership to a new node, it is only necessary to compute the membership of the example to the currently tested attribute, then use f_1 with the membership of the parent node.

The second solution is more computationally effective but requires more memory allocations. The memory allocated for the membership information will be $NumNodes \cdot NumExamples \cdot sizeof(double)$. For a tree of hundreds of nodes with thousands of examples the demand is for dozens of megabytes.

In this thesis, we define a process that will be both computationally efficient and will also have reasonable memory requirements.

Since we are using a DFS (Depth first search) recursion for creating the tree, it is sufficient to maintain the array containing the membership of examples to a node, only until all its children are built and are given these values. The parent node will no longer need to keep this information.

Our solution for handling membership values is therefore:

1. Build the root node, allocate an array for the membership values.
2. Create the node's children.
3. Allocate an array of membership values for all children except the last one.
4. Copy the membership values from the parent to the children.
5. Link the array of the parent node to the last child.
6. Calculate the new membership values according to the old ones and the values of the the attribute tested by that node.
7. Repeat the process (2-6).
8. When a node becomes a leaf, the array of membership values is released.

3.4.5 Pruning.

Pruning is also performed in a recursive DFS algorithm.

1. For each node (that is not a leaf) post prune its children:
 - (a) Test the accuracy of the tree on the pruning examples (see Section 3.4.6).

- (b) Replace the sub-tree of the node by a leaf node. The new leaf node parameters will be:

$$R_\ell(\text{new leaf node}) = \sum_{N_i \in \text{subtree}} R_\ell(N_i)$$

$$r_\ell(\text{new leaf node}) = \sum_{N_i \in \text{subtree}} r_\ell(N_i)$$

Test the accuracy of the tree after the node was replaced by the leaf.

If *Accuracy after pruning* \geq *Accuracy before pruning* – *Constant* then keep the pruned tree, else maintain the original tree.

3.4.6 Accuracy testing.

The accuracy of a decision tree is usually defined as follows:

$$\text{Accuracy} = \frac{\text{number of correctly classified examples}}{\text{number of tested examples}}$$

Where an example is considered correctly classified if the algorithm classifies a negative example as negative, and a positive one as positive. In our case, although it is desired to approximate the respiratory events (target events) locations, the more important issue is to know *how many* events have occurred per hour.

Let us suppose that we can use the percentage of correctly classified examples as the accuracy measure. Let us also take into account that, most of the examples are negative examples (only about 3% of the examples in the training set were found to be positive) and that, in many cases, similar features can lead to contradicting classifications.

In such a case, the generated algorithm will tend to prefer negative classifications. Thus, we will suffer from under-scoring of the algorithm, with respect to the manually scored events (i.e., the indices the algorithm will produce will be lower than the target ones).

We could try to solve this problem by equalizing the number of positive and negative examples in the training and pruning sets of examples. But, due to the inconsistency of the data, this will cause over-scoring, since the algorithm will avoid many cases of negative classifications in attribute values that have also caused some positive classifications.

Our conclusions from the above are that:

1. It is best to use the crisp result of an examples' classification (δ_j) as the probability that it will be a true event, and use this probability as is, rather than transform it to a binary value.
2. It is required to verify that the algorithm will not generate biased results (in terms of RDI's).

One more feature of the problem that needs to be addressed is that for the physician that accepts the results, an error of x units in the lower range of RDI's is more significant than an error of the same magnitude in the upper range of RDI's.

So, we will use a criteria that takes the manually scored RDI into account when calculating the accuracy of the training set. For the testing set of patients, the automatic and manual RDI's will be computed by the algorithm (accuracy measured for the testing set will be explained in Section 4).

We sum up all the examples that belong to a specific patient and compute his expected RDI as:

$$RDI_{exp}(P_i) = \frac{\sum_{e_j \in E(P_i)} \delta_j}{\sum_{e_j \in E(P_i)} Duration(e_j)}$$

where P_i stands for patient number i . $E(P_i)$ are the examples that originate from patient i .

The accuracy measure will consist of three terms, one for the error rate per example, one for the magnitude of the error in RDI and the last for the bias error in RDI:

$$\frac{\sum_{e_j \in E} (\delta_j - y_j)}{|E|} + \left(1 - \frac{\sum_{P_i \in P} |\Delta RDI(P_i)|}{ManRDI(P_i)}\right) + \left(1 - \sum_{P_i \in P} \frac{\Delta RDI(P_i)}{ManRDI(P_i)}\right)$$

where $\Delta RDI \triangleq ManRDI - AutoRDI$

3.4.7 Saving and loading the tree's structure using a binary file

The ability to learn the algorithm beforehand, and to incorporate the learned tree in a product that will be able to estimate a patient's RDI in a short period of time is essential in our case.

For this purpose and for the purpose of being able to deal with a very large amount of testing data, a mechanism that allows saving the learned tree in binary file format and is able to test it on a separate run was added.

The file used for maintaining the decision tree structure is saved and loaded using a recursive DFS algorithm. For each node on the tree, the following information is kept:

- number of children
- test attribute number
- linguistic value of the attribute tested on the parent node

- Total training examples times output values on the node (R_ℓ).
- Total training examples on the node (r_ℓ).

When the tree is loaded, this data is copied back to the original tree structure, to allow for testing or for classification of unseen examples.

Chapter 4

Experimental Results.

Four algorithms and their comparative results are described in this chapter. The four algorithms were tested on the same set of night studies, to allow for exact comparison of their performance. The first one described hereafter, is a basic algorithm developed by Itamar-medical LTD (an evaluation of this algorithm was quoted in [5]). An evaluation of this algorithm and its performance on our set of night studies is given in Section 4.1.

at the earlier stage of our research work, we wrote the second algorithm which was used as a test case for making a first step towards fuzzy reasoning. The algorithm is described in Section 4.2.

Last but not least, the results of the algorithm suggested by this thesis are illustrated in Section 4.3.

The statistical measures and the comparative analysis are described in Section 4.4.

4.1 Basic algorithm

The basic algorithm is based on heuristic rules that were visually found by researchers in the field, and on some statistical work, performed mainly for parameter optimization.

The basic amplitude and heart rate extraction methods are the same as the ones we use in our algorithm (up to constants). The smoothing of these channels is done through a moving average (no other noise reduction is performed). The PAT amplitude decrease and heart rate increase are found through an algorithm similar to the one we use for detection of desaturation and re-saturation events (see Section 3.3.2). The algorithm declares that an apnea event exists if one of the following occurs:

Algorithm 13. Basic heuristic algorithm for apnea detection

1. *An amplitude decrease of at least 30% and a heart rate increase of at least 10% were present.*

2. A desaturation of at least 4% was present.
3. An amplitude decrease of at least 30% and a desaturation of at least 3% were present.

where the *and* operator is implemented as a time-based intersection between the events (the events duration is determined by the duration of the increasing/decreasing part of the signal).

A scatter plot of the algorithm's result is given in 4.1.

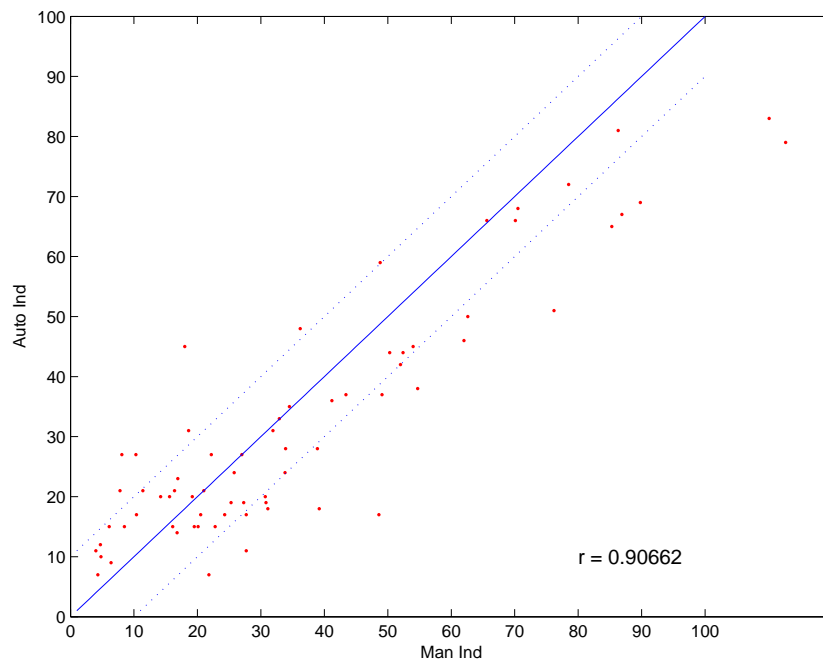


Figure 4.1: The figure shows a scatter plot of the RDI indices of the basic algorithm VS. the RDI indices of the manual scorer (gold standard) in red dots. The blue line indicate the values in which the automatic RDI id equal to the manual RDI. The two dashed lines indicate an error of 10 RDI's from the equality line.

4.2 Rule base algorithm

This algorithm was written while searching for the appropriate AI method for addressing the problem. We have used a couple of methods for setting the thresholds for the events changes:

- An Expert system [25] was used for choosing the right amplitude and heart rate threshold for declaring an event. The error measure used was the number of errors in classification.
- Using the first rule from Section 4.1, exhaustive search was used for determining the thresholds of the amplitude and heart rate changes, that would give the best performance in RDI per night study. The thresholds were chosen for the first half of the night study and then tested on the second half.

Both attempts did not give good results. The expert system had preference for thresholds that were too high (since these were the cases in which the probability for the presence of a true event is very high). Using these thresholds would inevitably result in under-scoring in terms of RDI. The exhaustive search worked well for some of the night studies, but miserably for others. Moreover, the thresholds chosen were significantly different from one night study to another.

These empirical results have implied, that using a crisp set of thresholds and relying merely on the PAT amplitude and heart rate changes are both undesired. It appeared that an approach that would fuzzify the thresholds and “defuze” as much information as possible from all the channels can be the one to give the best results. These conclusions led to the writing of the second algorithm that is described hereafter:

Algorithm 14. Rule base algorithm for apnea detection

1. *Detect amplitude and heart rate changes.*
2. *Detect desaturation and resaturation events.*
3. *Calculate (an equivalent to) the actigraph energy.*
4. *Use a sigmoid function to fuzzify the amplitude and heart rate change percentage.*
5. *For each lead event, estimate the expected target event’s location and margins to that location.*
6. *In an array of the night study length (in seconds), sum up the contribution from all of the automatic events.*
7. *Traverse the array, if the value in the array exceeds a predefined threshold, declare it as a location of the probable location for an event.*
8. *Unify close events’ locations.*

9. If a probable event's location is found a time segment that does not contain any changes in oximetry, filter that event out, (as it is probably a marker for an arousal that is non-respiratory).

Figure 4.2 illustrates the scatter plot generated by this algorithm.

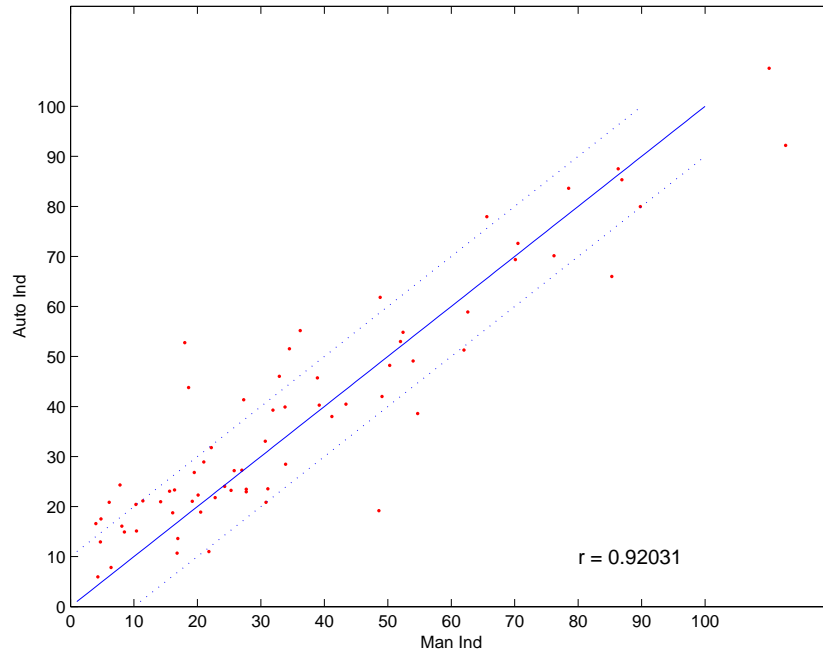


Figure 4.2: The figure shows a scatter plot of the RDI indices of the rule base algorithm VS. the RDI indices of the manual scorer (gold standard) in red dots. The blue line indicate the values in which the automatic RDI is equal to the manual RDI. The two dashed lines indicate an error of 10 RDI's from the equality line.

4.3 Results of the suggested algorithm

We run the learning algorithm on the set of training data, using the local ODI (local rate of desaturations) as a preliminary attribute to all other attributes, and using the binary attributes as preliminary attributes to the features that are dependant upon the existence of that type of event.

We achieved a tree of 89 nodes (consisting of 3 subtrees according to the local ODI as expected), it was possible to gain shorter trees also, by putting tighter constrains in the pruning stage, but the accuracy of the tree was poorer then.

When we reviewed the rules generated by the tree we found that most of the rules made sense, in terms that we would expect the same tendency, from our visual inspection of the signals.

The generated tree suggested Higher probability for an event when:

- Resaturation and / or desaturation occurs.
- As the amplitude change percent grows higher.
- Existence / higher change percent of HR events.
- Existence of actigraph movement event.
- Good synchronization between amplitude and HR events.

Using the learning algorithm allowed us to re-insure and quantize the influence of each of these features on the probability of having an apnea event.

When we run the algorithm on the validation group we gained the scatter plot shown on Figure 4.3.

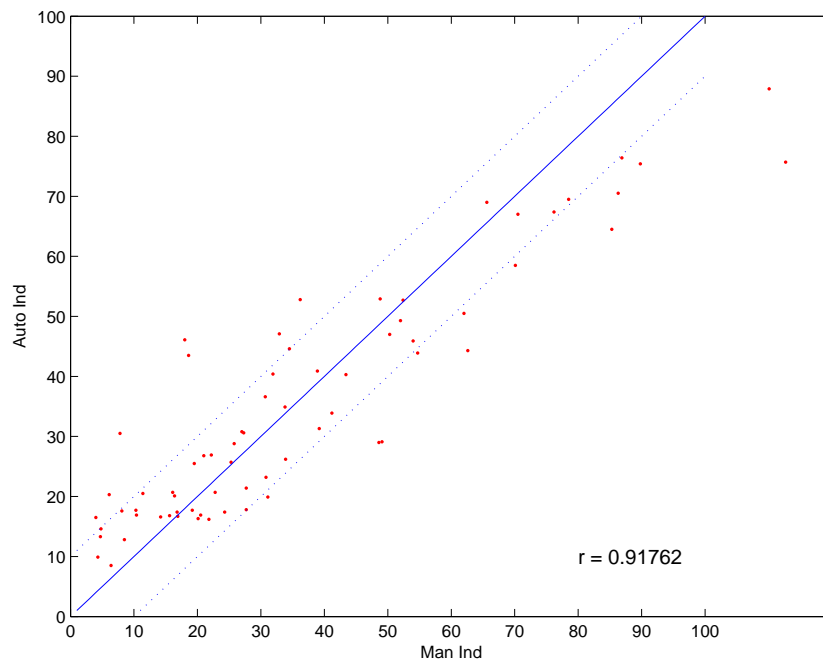


Figure 4.3: The figure shows a scatter plot of the RDI indices of the thesis algorithm VS. the RDI indices of the manual scorer (gold standard) in red dots. The blue line indicate the values in which the automatic RDI id equal to the manual RDI. The two dashed lines indicate an error of 10 RDI's from the equality line.

4.4 Comparative analysis

4.4.1 measures for comparison

The measures we will use for comparison are described hereafter:

- **Correlation Coefficient** - A statistical measure of the interdependence of two or more random variables. Fundamentally, the value indicates how much of a change in one variable is explained by a change in another. The random variables in our case are the manual and automatic RDI's.
- **Sensitivity**- The proportion of true positives it detects out of all the positives. In our case the positives are defined as patients (night studies) that are diagnosed as having an RDI larger than the CPAP treatment threshold. (the CPAP treatment threshold is set to 20 events per hour).
- **Specificity** - The proportion of true negatives it detects out of all the negatives. It is thus a measure of how accurately it identifies negatives. Again, negativity corresponds to the manually scored RDI's of a threshold that is lower than the therapeutic threshold.
- **Area Under Curve (AUC)** - A plot of true positive (sensitivity) versus false positive results (1-specificity). The curve shows the diagnostic capability of the automatic RDI: The Man RDI is set to the therapeutic threshold of 20 events per hour and the Automatic RDI varies from 1-100. A curve of the sensitivity and (1-specificity) for the range of thresholds is drawn. The AUC measure is the area under the ROC curve.
- **Accuracy**- Accuracy is the percentage of correctly classified incidents (both positives and negatives). Hence, the accuracy in our case is the number of correctly classified patients in terms of therapeutic needs.
- **Absolute difference** - The average of the absolute errors in RDI.
- **Mean difference** - The sum (magnitude and size) of the errors in RDI. (i.e., $\sum AutoRDI - ManRDI$).
- **White's agreement** - This measure is specific to the field of apnea detection. It relays on the fact that, for a physician it is more important to have good accuracy in RDI for the lower range of values (patients that do not have a severe disturbance) than the higher one. Therefore the measure is defined as the number of correctly classified instances where an instant is considered to be correctly classified if: The manual RDI is below a threshold (40 events per hour) and the error magnitude is less than a threshold (10 events per hour) OR The manual RDI is above the threshold (of 40) and the Automatic RDI is also above that threshold.

4.4.2 Statistical results

The statistical results of the three versions described above are summarized in Table 4.1.

Table 4.1: Comparison of performance of the different algorithms

Criteria - Algo	Basic	Rule Base	Thesis
Correlation	91	92	92 ($p < 0.0001$)
Sensitivity	73	93	89
Specificity	67	63	83
Area Under Curve	82	90	91
Accuracy	71	87	87
Abs Diff (mean \pm STD)	10 \pm 8	8 \pm 7	9 \pm 8
Mean Diff (mean \pm STD)	-4 \pm 12	2 \pm 10	-1 \pm 11
White's Agreement	75	80	84

Two measures that are most commonly used for determining the diagnostic capabilities of an algorithm are the sensitivity and specificity. An algorithm is considered as being a good diagnostic tool if its sensitivity and specificity are both balanced and high. Note that it is easy to achieve 100% sensitivity or 100% specificity with a trivial algorithm that always says “ill” or always says “healthy”. From the table above, we can see that there is an improvement in the sensitivity and specificity measures. This improvement is also reflected in the area under ROC curve and the accuracy.

Furthermore there is also an improvement in the White's agreement score which allows for better determination of the type of treatment desired for the patients and for its' necessity (whether the treatment is recommended or compulsory).

There is no real improvement, and even a degradation in some of the measures that describe the mean and STD of the errors of the indices. This is mainly caused by relatively large errors in the higher indices (people with severe OSA). This is a bit surprising, since it is considered much easier to accurately detect respiratory events in the higher indices than in the lower ones. Two reasons can explain the degradation of the accuracy of the thesis algorithm in the higher indices range:

1. The training set chosen contained more patients with mild OSA than patients with severe OSA.
2. The size of window chosen for generating the training and the testing examples (25 seconds) is rather big for severe patients.

Two suggestions for solving this problem could be:

1. Add patients with higher RDI values to the training set.
2. Change the minimal time gap between the examples according to the desaturation index of the patient's night study.

Better results in the higher RDI indexes range could improve both the criteria involving the differences in indexes and the correlation criteria, but would not improve patients screening and treatment plan. Therefore, we preferred not to focus on solving these problems.

Chapter 5

Summary

In our work, we have introduced the problem of estimating the RDI via indirect measurements of the blood flow changes in the finger, finger oximetry and arm movements. We had improved some of the signal processing methods used for extracting information from the raw data, and generated more features, in order to allow the learning algorithm to search for relevant information in these features too.

Due to the time -related character of our problem, we had to deal with the question of how to define the examples for the learning algorithm and how to relate its features and its label to it. A system of finding clues for the possibility that an event will occur, define the place with largest possibility as the "center" of the example, and choose its label and features using a nearest neighbors matching criteria was the solution chosen for this problem.

We also had to deal with performance measures. We found a solution for the dilemma of using more memory versus using more computation complexity by use of properties of depth first search characteristics. We added the dimension of using artificial intelligence. We have found an alternative way for estimating the rate of apnea events per hour during night sleep.

We have shown that, in our case, artificial intelligence can be used beneficially for generating rules to describe a phenomena, that traditionally required a lot of human research and trial and error processes to conclude empirically. By using a decision tree we had managed to generate an algorithm that "agrees" with the rules that were empirically found by human researchers, and added more rules to them.

We had utilized the properties of fuzzy logic, that proved to be highly appropriate for this type of problems, especially due to the fact that we are dealing with an indirect measurement of the phenomena we want to quantify. The fact that we try to define an apnea existence from the sympathetic activation that it may (or may not) generate, and not from actually viewing signals that measure abdominal or thorax respiratory effort or thermal changes in the nose-mouth area, makes it impossible to define clear-cut rules, and suggests

that a method that will integrate more information, in order to obtain the possibility of an event to be present, is more appropriate.

We have added the concept of attributes priorities to the fuzzy logic decision tree, and have used its functionality for using the information that an event did not occur via this mechanism of attributes priorities in the following manner: we add a boolean attribute for noting that an event is not present in this neighborhood and set dependencies of all the attributes that are connected with the boolean attribute to this attribute, such that these attributes could only be tested if the "parent" attribute was true. We had created a combined fuzzy-boolean tree that has inter-attributes connections.

While the algorithm shows an improvement in the diagnostic measures on the validation set, perhaps its real strength lies in the fact that it is generated automatically. While the two algorithms it is compared with required major efforts in order to understand and prioritize the features and select the relevant parameters, our algorithm was able to perform this task automatically.

As further research we suggest working on choosing the fuzzy membership functions automatically. We have started building a mechanism that does so. While experimenting, we concluded, that it is important to define the right optimization criteria in order to perform this task. We also suggest using our algorithm for estimating similar quantities, such as the rate of arousals per hour of sleep (ARI) or apnea classification.

References

- [1] R. P. Schnall and A. Shlitner et al, “Periodic, profound peripheral vasoconstriction a new marker of obstructive sleep apnea,” *Sleep*, vol. 22, no. 7, pp. 939–946, 1999.
- [2] P. Lavie and A. Shlitner et al, “Peripheral arterial tonometry: A novel and sensitive non-invasive monitor of brief arousals during sleep,” *IMAJ*, vol. 2, no. 3, pp. 246, 2000.
- [3] G. Pillar and A. Bar et al, “Autonomic arousal index(aai): An automated detection based on peripheral arterial tonometry,” *Sleep*, vol. 25, no. 5, pp. 541, 2002.
- [4] L. Grote and D. Zou et al., “Finger plethysmography - a method for monitoring finger blood flow during sleep disordered breathing,” *Respiratory Physiology and Neurobiology*, vol. 136, pp. 141–152, 2003.
- [5] A. Bar and G. Pillar et al, “Evaluation of a portable device based on peripheral arterial tone for unattended home sleep studies,” *Chest*, vol. 123, no. 3, pp. 695–703, 2003.
- [6] B. Raymond and R.M.Cayton et al, “Combined index of heart rate variability and oximetry in screening for sleep apnoea/hypopnoea syndrom.,” *Sleep Res*, vol. 12, pp. 53–61, 2003.
- [7] T. Pensel and U. Brandenburg et al, “New methods for the non-invasive assesment of symphathetic activity during sleep,” *Somnologie*, vol. 6, pp. 69–73, 2002.
- [8] B.Mazzanti and C.Lamberti et al, “Validation of an ecg-derived respiration monitoring method,” Available from: www.morata.it/paper.pdf.
- [9] E. Huupponen and S.L. Himanen et al, “Fuzzy detection of eeg alpha without amplitude thresholding.,” *Artificial intelligence in Medicine*, vol. 24, pp. 133–147, 2002.
- [10] A.M. Bensaid and N.Bouhouch et al, “Classification of ecg patterns using fuzzy rules derived from id3-induced decision trees,” in *Fuzzy Information*

- Processing Society - NAFIPS, 1998 Conference of the North American*”, pp. 25–28, 1998.
- [11] C. P. O’donnell and L. Allan et al, “The effect of upper airway obstruction and arousal on peripheral arterial tonometry in obstructive sleep apnea,” *Am J Respir Crit Care Med*, vol. 166, pp. 965–971, 2002.
- [12] J. H. Friedman, “A recursive partitioning decision rule for nonparametric classifier,” *IEEE Transactions on Computers*, vol. 26, pp. 404–408, 1977.
- [13] L. Breiman and J. H. Friedman et al, *Classification and regression trees*, Wadsworth, 1977.
- [14] J. R. Quinlan, “Induction of decision trees,” *Machine Learning*, vol. 1, pp. 81–106, 1986.
- [15] J.R. Quinlan, *C4.5-Programs for machine learning*, Morgan Kaufmann Publishers, 1988.
- [16] Mathworks, “Fuzzy logic toolbox for use with matlab - users guide,” Available from: www.mathworks.fr/access/helpdesk/help/toolbox/fuzzy/fuzzy.shtml.
- [17] C. Z. Janikow, “Fuzzy decision trees: Issues and methods,” *IEEE Transactions on systems, Man and Cybernetics*, vol. 28, no. 1, pp. 1–14, 1998.
- [18] J. R. Quinlan, “Unknown attribute values in induction,” in *Proc. of the Sixth International Workshop on Machine Learning*, pp. 164–168, 1989.
- [19] C. Z. Janikow, “Exemplar learning in fuzzy decision trees,” in *Proceedings of FUZZ-IEEE*, pp. 1500–1505, 1996.
- [20] C. Z. Janikow and M. Fajfer, “Fuzzy partitioning with fid3.1,” in *Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society*, pp. 467–471, 1999.
- [21] K. Crockett and Z. Bandar et al, “Fuzzy inference framework for induced decision trees,” in *Proceedings of the European Conference on Artificial Intelligence*, pp. 425–429, 1998.
- [22] S. Mitra and M. Kishori et al, “Fuzzy decision tree, linguistic rules and fuzzy knowledge-based network:generation and evaluation,” *IEEE transactions on systems, man, and cybernetics Part C: Applications and reviews*, vol. 32, no. 4, pp. 1–14, 2002.

- [23] J. Zeidler and M. Schlosser, “Fuzzy handling of continuous-valued attributes in decision trees,” in *Proc. ECML-95 Mlnet Familiarization Workshop ”Statistics, Machine Learning and Knowledge Discovery in Databases”*, pp. 41–46, 1995.
- [24] X. Boyen and L. Wehenkel, “Fuzzy decision tree induction for power system security assessment,” in *Proc. of SIPOWER’95, IFAC Symp. on Control of Power Plants and Power Systems*, pp. 151–156, 1995.
- [25] J. Zeidler and M. Schlosser, “Averaging expert predictions,” in *Proceedings of the 4th European Conference on Computational Learning Theory*, pp. 153–167, 1999.

Appendix A

Data file format

This appendix describes the file format of the examples required for the learning algorithm. It includes a list of parameters and their membership functions, and then a list of examples.

The file is generated automatically using an application based on Matlab platform. The first line in the file contains the number of the attributes and number of examples. Then a list of the attributes characteristics (starting with the classification which is also regarded as an attribute in this respect). Each attribute is described in one line that describes the following characteristics:

1. Attribute number.
2. Attribute name.
3. Attribute type ('f' for fuzzy 'b' for binary).
4. Number of membership functions (equals zero if not fuzzy).
5. For each membership function the following fields are listed:
 - (a) The linguistic term.
 - (b) The number of parameters to describe the membership function
 - (c) The membership function parameters:
 - i. If number of the membership functions' parameters equals 2, then the membership function parameters describe the parameters a and c in the function describing a sigmoid:

$$\mu(x) = \textit{sigmoid}(x; a, c) = \frac{1}{1 + \exp(-a \cdot (x - c))}$$

- ii. If previous field equals 4, then the membership function parameters define the parameters a_1, c_1, a_2, c_2 in the following equation:

$$\mu(x) = \textit{sigmoid}(x; a_1, c_1) - \textit{sigmoid}(x; a_2, c_2)$$

6. Number of preliminary attributes (for more details see Section 3.4.2).
7. If the number of preliminary attributes is not zero, a list of the previous attributes numbers is given.

After all the information about the attributes was listed, a list of all the examples is detailed. Attribute values are listed according to their appearance in the attributes' information list.

Appendix B

List Of attributes and their use in the algorithm.

We have used thirty-three attributes for generating the examples to be used by the learning algorithm. These attributes and the algorithms used for generating them are explained in detail in section 3.3.2.

A summary of the list of attributes is given hereafter

- Amplitude change existence.
- Amplitude change duration.
- Amplitude change baseline.
- Amplitude change percent.
- Amplitude change slope.
- Heart rate change exists.
- Heart rate change duration.
- Amplitude to heart rate change time difference.
- Desaturation event exists.
- Desaturation duration.
- Desaturation change percent.
- Desaturation slope.
- Amplitude change to desaturation time difference.
- Resaturation event existence.

APPENDIX B. LIST OF ATTRIBUTES AND THEIR USE IN THE ALGORITHM.70

- Resaturation event duration.
- Resaturation recovery (checks if returns to saturation level before desaturation event started).
- Resaturation event change percent.
- Resaturation event slope.
- Amplitude to resaturation time difference.
- Resaturation to desaturation time difference.
- Movement event existence.
- Movement event duration.
- Movement total energy.
- Movement maximum energy.
- Amplitude change event to Movement event time difference.
- Movement event to resaturation event time difference.
- REM flag.
- Periodicity in amplitude channel flag.
- Periodicity in amplitude time period.
- Local ODI.

The learning algorithm chose to use 14 of these attributes. From visually analyzing the generated tree we see the following tendencies:

- The most important attribute was, of course the local ODI, as this was manually set by using the attributes hierarchy mechanism.
- Next the amplitude change event existence and the resaturation event existence were chosen (when they exist, the probability of a true event is increased).
- Next significant attribute was the amplitude change percent (higher change percent increases the probability of a true event).
- Next was the desaturation event existence (existence increases the probability for an event).

- Next attributes used were the desaturation change percent (higher change percent increased the probability for an event), the resaturation slope, Heart rate existence (existence increases the probability for an event), actigraph movement existence (existence increases probability, amplitude event duration (longer increases probability for an event). REM (lower probability for an event during REM periods).
- Attributes chosen for which is was more difficult to understand the physiological explanation where: resaturation slope, heart rate baseline (slower heart rate increases probability for an event) and amplitude baseline (higher increases probability).
- For examples with higher local ODI, the periodicity flag was tested (periodicity increased probability of a true event). This is consistent with our physiological knowledge.
- For Amplitude to heart rate change event distance, the probability was higher if close to zero (lower for both positive and negative distances). This is also consistent with our physiological knowledge.