Abteilung Neuroinformatik
Prof. Dr. Günther Palm

Temporal Sensorfusion for the Classification of Bioacoustic Time Series

Dissertation zur Erlangung des Doktorgrades Dr. rer. nat.
der Fakultät für Informatik der Universität Ulm

von

Christian R. Dietrich

aus Hirschegg

2003

Amtierender Dekan:	Prof. Dr. Friedrich W. von Henke
Erster Gutachter	Prof. Dr. Günther Palm
Zweiter Gutachter	PD Dr. Alfred Strey
Tag der Promotion	25.06.04

Abstract

Classifying species by their sounds is a fundamental challenge in the study of animal vocalisations. Most of existing studies are based on manual inspection and labelling of acoustic features, e.g. amplitude signals and sound spectra, which relies on the agreement between human experts. But during the last ten years, systems for the automated classification of animal vocalisations have been developed.

In this thesis a system for the classification of *Orthoptera* species by their sounds is described in great detail and the behaviour of this approach is demonstrated on a large data set containing sounds of 53 different species. The system consists of multiple classifiers, since in previous studies it has been shown, that for many applications the classification performance of a single classifier system can be improved by combining the decisions of multiple classifiers.

To determine features for the individual classifiers these features have been selected manually and automatically. For the manual feature selection, pattern recognition and bioacoustics are considered as two coherent interdisciplinary research fields. Hereby the sound production mechanisms of the *Orthoptera* reveals significant features for the classification to family and to species level. Nevertheless, we applied a wrapper feature selection method, the sequential forward selection, in order to determine further discriminative feature sets for the individual classifiers.

In particular, this thesis deals with classifier ensemble methods for time series classification applied to bioacoustic data. Hereby a set of local features is extracted inside a sliding time window which moves over the whole sound signal. The temporal combination of local features and the combination over the feature space is studied.

Static combining paradigms where the classifier outputs are simply combined through a fixed fusion mapping, and *adaptive combining paradigms* where an additional fusion layer is trained through a second supervised learning procedure are proposed and discussed. The decision template (DT) fusion scheme is an intuitive approach for such a trainable fusion

scheme, which is typically applied to recognise static objects. During the second supervised learning step the DT algorithm uses confusion matrix data to adapt the fusion layer. In several empirical studies it has been shown that the classification performance of adaptive fusion schemes, particularly for the so called decision template, is superior.

Many linear trainable decision fusion mappings, e.g. the combination with the linear associative memory, the pseudoinverse matrix and the naive Bayes fusion scheme are based on the same idea. Links between these methods are given. However, regarding the classification of temporal sequences these methods do not consider the temporal variation of the classifier outputs.

In order to deal with variations of classifier decisions within time series we propose to calculate multiple decision templates (MDTs) per class. Two new methods called *temporal decision templates* (TDTs) and *clustered decision templates* (CDTs) are introduced and the behaviour of these new methods is discussed on real data from the field of bioacoustics and artificially generated data. In contrast to the combination with decision templates the multiple decision template approaches lead to an increased expression power of the fusion layer. This enhances the classification performance for classification problems where the outputs of the individual classifiers show different characteristic patterns, which is a typical behaviour in time series classification applications. Hereby several characteristic patterns may exist for the whole time series belonging to the same class or for the individual local decisions of a single time series. Whereas the TDT approach offers the advantage to learn several characteristic patterns for whole time series, the CDT approach is also able to learn different characteristic patterns over time. Such patterns have been found in the temporal domain of the *Orthoptera*.

Acknowledgements

This thesis was supported by the help and the assistance of many people whose efforts I hereby would like to gratefully acknowledge.

First of all, I would like to thank my adviser Prof. Dr. Günther Palm for his steady support, for giving me indispensable motivation and for giving me the right ideas at the right time. His valuable advice and guidance will always be appreciated. I am greateful to PD Dr. Alfred Strey for serving as a reading committee member, who supported this work with valuable directions and a meticulous review of this manuscript.

Especially, I would like to thank to my thesis adviser Dr. Friedhelm Schwenker who contributed to this thesis with his guidance, generous assistance and many fruitful discussions. Despite of his own tight schedule, he always found a large amount of time for writing many papers in the field of classifier fusion together with me, and for carefully reading this manuscript.

Many thanks to the members of the DORSA project who improved this work in the field of bioacoustics, significantly. In particular, I would like to thank PD Dr. Klaus Riede, Dr. Sigfrid Ingrisch, Dr. Klaus G. Heller and Dr. Frank Nischk for providing their sound recordings, suggestions, fruitful discussions and the successful cooperation.

Special thanks to all my present and former colleagues at the department of neural information processing for their friendliness and helpfulness that makes working each single day a real pleasure for me. Valuable discussions with my colleagues improved this work considerably.

I also want to thank Barbara Previer for carefully reading this manuscript and to Dr. Axel Baune for his support regarding LaTeX.

Last, but most important I am grateful to my family for all their patience, support and loving. My son Daniel who makes the nights shorter but my days much brighter and my marvellous wife Dr. Katrin Dietrich for her encouragement and never-ending love.

Contents

List of Figures

List of Tables

Abbreviation

Classification and Classifier Combination

$s(t)_{t=1}^{T}$	A discrete waveform consisting of T samples, recorded at a environment temperature of ϑ degree
$\lambda = (\lambda_1, ..., \lambda_J)$	Pulse onsets of $s(t)_{t=1}^{T}$
$\mu = (\mu_1, ..., \mu_J)$	Pulse offsets of $s(t)_{t=1}^{T}$
$\mathbf{x}_i = (\mathbf{x}_{i,1}, ..., \mathbf{x}_{i,D_i}) \in \mathbb{R}^{D_i}$	A feature vector from the i-th feature space
$X = (\mathbf{x}_1, ..., \mathbf{x}_I) \in \mathbb{R}^{\sum D_i}$	A sequence of I feature vectors $\mathbf{x}_i \in \mathbb{R}^{D_i}$, $i = 1, ..., I$
$X(\jmath) = (\mathbf{x}_1(\jmath), ..., \mathbf{x}_I(\jmath))$	A sequence of I feature vectors extracted inside the $\jmath$-th time window $W^{\jmath}$, $\jmath = 1, ..., \mathcal{J}$
$(\mathbf{x}_i(\jmath))_{\jmath=1}^{\mathcal{J}}$	A feature stream consisting of temporal features from the i-th feature space, $\mathbf{x}_i(\jmath) \in \mathbb{R}^{D_i}$, $\jmath = 1, ..., \mathcal{J}$
$(X(\jmath))_{\jmath=1}^{\mathcal{J}}$	A set of I feature streams $X(\jmath) \in \mathbb{R}^{\sum D_i}$, $\jmath = 1, ..., \mathcal{J}$
$\mathcal{C} = \{\mathcal{C}^1, ..., \mathcal{C}^N\}$	A set of N probabilistic classifiers
$\mathcal{C}^n(\mathbf{x}_n) \in \Delta$	The output of the n-th classifier given $\mathbf{x}_n \in \mathbb{R}^{D_n}$, $\Delta := \{(\mathbf{y}_1, ..., \mathbf{y}_L) \in [0,1]^L \mid \sum_{l=1}^{L} \mathbf{y}_l = 1\}$
$\mathbf{y} \in \mathbb{R}^L$	Real classifier output (activity of L output neurons)
$\mathcal{F} : \Delta^N \rightarrow \Delta$	Fusion mapping for the combination of N classifiers
$\mathbf{z} \in \Delta$	The probabilistic classifier ensemble output
$\Omega = \{1, ..., L\}$	The set of L class labels

$P(\mathbf{x})$	Probability density function (p.d.f) of $\mathbf{x}$
$P(\mathbf{x}\|\omega = l)$	Class-conditional p.d.f of $\mathbf{x}$ given $\omega = l$
$P(\omega = l)$	Prior probability of class $l \in \Omega$
$P(\omega = l\|\mathbf{x})$	Posterior probability of class $l \in \Omega$ given $\mathbf{x} \in \mathbb{R}^D$
$\mathcal{D}$	Labelled training data in order to train the base classifiers
$\mathcal{D}^{\mathrm{V}}$	Labelled validation data to train the fusion layer
$\mathcal{D}^{\mathrm{T}}$	A separate test set
$\rho \in [0, 1]$	Fraction of data points of $\mathcal{D}^{\mathrm{V}}$ which are also in $\mathcal{D}$
$\mathcal{P}$	Decision profile ($I \times L$ matrix)
$\mathcal{T}^{\omega}$	Decision template (or set of decision templates) of class ω ($I \times L$ matrix)
$\mathcal{S}(\mathcal{P}, \mathcal{T}^{\omega}) \in [0, 1]$	Similarity between the decision profile and the decision template of class ω
$K \in \mathbb{N}$	Number of decision templates for each class
$S \in \{1, ..., K\}$	Number of decision templates to select

Bioacoustic Features

B	Filter-bank energies
D, $\bar{\mathbf{D}}$	Pulse distances
L	Pulse length
C	Pulse frequency contour
F	Pulse frequency
E, $\hat{\mathbf{E}}$	Energy contour of pulses
T	Time encoded signals
A	Maximal amplitude of pulses
P	Probability density of pulse distances

Some Common Acronyms

K-NN	K-nearest neighbour (classifier)
AM	Associative memory
ANN	Artificial neural network
BKS	Behaviour-knowledge space
BP	Backpropagation (optimisation algorithm)
CDT	Clustered decision template
CFT	Classification, fusion, temporal integration
CTF	Classification, temporal integration, fusion
DORSA	In German: Deutsche Orthopteren Sammlungen
DT	Decision template
DTW	Dynamic time warping
EM	Expectation-maximisation
FCT	Fusion, classification, temporal integration
FFT	Fast Fourier transform
FSRC	Fractional sampling rate conversion
GA	Genetic algorithm
GBC	Generalised borda count

GIS	Geographic information system
HMM	Hidden Markov model
LPC	Linear predictive coding
LVQ	Learning vector quantisation
MCS	Multiple classifier system
MDT	Multiple decision template
ML	Machine learning
MLP	Multilayer perceptron
NB	Naive Bayes (fusion scheme)
NN	Neural network
OSF	Orthoptera species file
OWA	Ordered weighted averaging
p.d.f	probability density function
PI	Pseudoinverse, e.g. pseudoinverse matrix
PR	Pattern recognition
RBF	Radial basis function
SFS	Sequential forward selection
SFSM	Sequential forward selection for multiple classifiers
SVM	Support vector machines
TDNN	Time delay neural network
TDT	Temporal decision template
TES	Time encoded signals
TS	Trace segmentation

Chapter 1

Introduction

The general issue of fusion of multiple experts has been of interest to the pattern recognition, machine learning and neural network communities who have been studying multiple classifier systems (MCS) for more than a decade [XKS92]. The combination of multiple classifiers has been proposed as an approach to the development of high performance classification systems [GR00] and to solve difficult pattern recognition problems involving large class sets and noisy input [HHS92].

A special case is the combination of multiple classifiers for the classification of time series. Typical applications can be found in many domains: the *audio domain* (speech recognition, sound identification, bioacoustics, etc.), the *vision domain* (motion detection, target tracking, object recognition, gesture recognition, face recognition in video sequences etc.), the *locomotion domain* (synchronised movement, robotics, mapping, navigation, etc.) and the *bioinformatic domain* (DNA sequences for example to locate or classify gens and biomedical signals). In these domains the classification is only possible by taking the processing of the temporal patterns into consideration because the variation of the features over time is usually significant for the classification of the temporal sequence. Looking at the amplitude or frequency at one point is of course not sufficient to recognise words or species. The consideration of these time dependent temporal variations makes time series classification distinct and more complex than the classification of static objects.

Many well-established techniques which are typically based on a priori information about the classification problem [Geu02] are known and have been evaluated [Kad02, MSC91, RJ86, RJ93]. Among the huge variety of algorithms *partially recurrent neural networks* [MSC91], *time delay neural networks* [WHH+89], *hidden Markov models* [BPSW70] and *dynamic time warping* [SC78] are most popular. Recently however, many neural network architectures were adapted to integrate temporal information for the processing of temporal sequences [EP99]. Most of these architectures are composed of a short-term memory that holds on to relevant past events and an associator that uses the short-term memory to classify or predict [Moz93].

Due to the success of multiple classifier systems for the classification

of static objects, in the last few years they also have been applied for the classification of temporal sequences. Promising classification results can be found for many applications [CXC96, GD00, DG01].

The goal of this thesis is to study the classification of temporal sequences consisting of features from multiple feature spaces utilising classifier ensemble methods with static and adaptive fusion mappings. Typically, for adaptive fusion mappings the use of the prior behaviour of the classifier yield useful information for the combination of the individual classifiers. Such prior behaviour is for example measured a confusion matrix that is calculated in a second training phase. Many combining methods like decision templates (DTs) [KBS98], pseudoinverse matrices (PIs), naive Bayes fusion (NB fusion), behaviour-knowledge space (BKS) and generalised borda count (GBC) [Par01a] are based on this idea. But, these methods are optimised to classify static objects.

In order to integrate temporal information for the combination of a set of classifiers into the fusion layer we propose to calculate multiple decision templates (MDTs) per class. Two new methods called *temporal decision templates* (TDTs) and *clustered decision templates* (CDTs) are introduced [DPS03]. Both methods show many similarities to the DT fusion scheme, but in contrast to the combination with decision templates the expression power of the fusion layer is increased. This enhances the classification performance for classification problems where the outputs of the classifiers show different characteristic patterns, which is a usual behaviour in time series classification applications.

The behaviour of these methods is demonstrated on a real and on a synthetic data set with temporal features. Whereas the real data set is from the field of bioacoustics and contains *Orthoptera* songs of two families: the *Gryllidae* (crickets) and the *Tettigoniidae* (katydids), the synthetic data set contains time series consisting of highly overlapping Gaussian distributions. Both data sets are utilised to evaluate the discussed combining schemes.

1.1 Motivation

The work described in this thesis is part of the DORSA project (in German: Deutsche Orthopteren Sammlungen) which deals with the collection of important *Orthoptera* specimens in German research collections to be appropriated via the internet (see `www.dorsa.de`) [RID01]. It is linked to the global species database *Orthoptera Species File* (OSF) and also includes multimedia information such as pictures, video-files and especially

sound-files. In addition, locality information will be transformed into distribution maps with a Geographic Information System (GIS) [IRD01].

The purpose of this thesis is to develop a prototype of a "rapid assessment tool" for the automatic recognition of *Orthoptera* songs to species level and to study classifier ensemble methods for the classification of bioacoustic data. The tool will help to improve classical taxonomic work such as the description of new taxa by allowing a more comparative analysis of cricket and katydid songs. In addition, the tool automatically extracts discriminative features to facilitate taxonomic work.

A further application is to detect, classify and monitor *tropical biodiversity* of a particular region by analysing the spectrum of insect sounds in this region. Currently these biodiversity assessments are very time consuming and require extensive and thus expensive input from taxonomical specialists. With the "rapid assessment tool" the biodiversity assessment in forests will be as simple as walking into them with a microphone and recording the sounds of resident insects. This is possible because the diversity of sounds made by forest insects is related to its species diversity. Furthermore, the "rapid assessment tool" may also be used to collect locality information such as used in the GIS system. The overall system will be especially useful for biologists, ecologists, conservationists and applied entomologists.

From the viewpoint of computer science the development of the "rapid assessment tool" mainly contains an interesting classification problem due to the structure of the animal its vocalisations. Generally the classification of bioacoustic vocalisations is a typical time series classification problem similar to speaker recognition and speaker identification. Although the structure of animal vocalisations is relatively simple, in contrast to human speech, the classification of species is often difficult due to adverse conditions (e.g. field recordings with nonhomogeneous noise background and other environmental influences) [KM98].

As in speech recognition where onset- and offset positions of words are determined, in the classification of animals the positions of the animal its signal has to be determined. For example, for *Orthoptera* species the positions of the individual pulses allows to extract local features within these pulses. These features are then locally classified and the classification results are combined through temporal fusion [DSP01a]. But fusion of multiple classifiers is also used to combine local features from different feature spaces and to combine the classifier decisions derivated in several time scales. These time scales are particularly important for the katydid species whose pulses consist of pulse groups called syllables. Thus, for katydid species the classifier decisions of descriptive features extracted

within pulses and the classifier decisions of features extracted from syllables are combined to get the final decision. Due to the fact that the local decisions are rather uncertain one important requirement to the classifier and the combining scheme is, that both are able to handle this uncertainty. In addition the system has to be robust to distortions, natural variability, noise and missing information [MSC91].

1.2 Related Work

Related work can be found in the field of classifier fusion and automated bioacoustics. Whereas in the field of classifier fusion methods of a narrow range are discussed, in the field of automated bioacoustics typical classification systems developed for the automated classification and characterisation of animal vocalisations are referred. However, for both research areas, throughout this thesis, relevant previous and related work is surveyed and many pointers to the literature are provided.

Classifier Fusion

Fusion of multiple classifiers has been studied intensively for decades. This fruitful and ongoing research is documented in a vast amount of scientific articles and numerous books. It is therefore beyond the scope of this thesis to give a complete account of work that has been done in this field up to now. Rather, we mainly refer previous work regarding classifier ensemble methods which use confusion matrix data to obtain some prior knowledge of the behaviour of the classifiers[1] and related work concerning ensemble methods for the classification of temporal sequences. In the first case the prior knowledge is derivated from errors that the individual classifiers have made during the overall training phase.

Early examinations of such methods can be found in [XKS92] which have been cited by many authors in the field of MCS. XU introduces static combining schemes, e.g. averaging, voting and the probabilistic product and adaptive combining schemes, e.g. the *naive Bayes* (NB) *combination* (see Section 3.6) which assumes that the classifiers are mutually independent. The ideas are based on the Bayesian formalism for evidence gathering and uncertainty reasoning which have been extensively by studied in [Pea88].

Very similar to the naive Bayes combination is the PALISK-algorithm (in German: Paralleler Linearer Statistischer Klassifikator) proposed by

[1] In many publications in the field of MCS these classifiers are denoted as *base classifiers* or *first level classifiers*.

HAUSMANN [Hau94]. During the training phase and the classification phase, each component of the individual feature vectors is classified individually. The number of classifiers is then given by the dimensionality of the feature vectors. HAUSMANN compares the PALISK fusion scheme with the Bayes classifier and artificial neural networks and applied his method in a speech recognition system for the classification of phonemes.

Another adaptive combining method is *behaviour-knowledge space* (BKS). The method was proposed by HUANG and SUEN [HS95] and considers the support from all classifiers in the team to all possible class labels combinations jointly. Hereby the individual classifier decisions of a team of N classifiers are used to build a look-up table. Let L be the number of class labels and $(\omega_1, ..., \omega_N) \in \Omega^N$ with $\Omega = \{1, ..., L\}$ be the crisp classifier outputs of the classifiers in the team. Then every possible combination of class labels is an index regarded as a cell in the look-up table (BKS table) whose dimensionality is given by the number of classifiers N. In the training phase the elements of the cells are determined by assigning the most representative class labels in the training data to the corresponding cell. On that score, each cell can have either: no class label, one class label or a set of class labels.

In the test phase the BKS table is used to combine the individual classifiers. Let $(\hat{\omega}_1, ..., \hat{\omega}_N) \in \Omega^N$ be the crisp classifier outputs calculated by assigning test data to the N classifiers. Again the sequence of class labels determines a cell in the look-up table. The ensemble output is then given by a single element of the determined cell, which means, that for an empty cell one class label is randomly selected from Ω, as for a cell containing a set of class labels a single class label is randomly selected from this set. For a cell containing a single class label the ensemble output is then given by exactly this class label. The method was improved by WERNECKE [Wer92] who considers the 95 % confidence intervals of the frequencies in the BKS-cells. In several publications this method is denoted as BKS+.

The discussed methods (e.g. NB, PALISK, BKS and BKS+) have in common, that they are based on crisp classifier decisions. This implies, that for the training of the fusion layer a large number of feature vectors must be classified. Otherwise, a large number of elements in the confusion matrix are zero which is usually worsening the classification performance of the classifier ensemble.

We now want to consider weather the combination of soft classifier outputs may be preferable in contrast to crisp classifier decisions. Such issues have been the key subject of interest in a series of workshops on audio and video based personal identity authentication [BS01] and many other applications.

KUNCHEVA was one of the first authors who proposed an adaptive classifier fusion scheme by taking the soft decisions of multiple classifiers into account. She has developed the *decision template* (DT) combining scheme [KBS98, Kun01] which was originally denoted as *fuzzy template*. The classification performance of the DT combining scheme in comparison to other well known methods for the combination of multiple classifiers was studied in several publications. For example, in [KBD01] a numerical study for 14 different fusion schemes[2] and the decision template fusion scheme using 11 different similarity measures on the `Satimage` and `Phoneme` data set from the database `ELENA` is given. The authors found, that the classification performance of the DT algorithm is superior in comparison to other methods.

Further preliminary work concerning the combination of multiple classifiers with decision templates may be found in [KJ00]. In these studies genetic algorithms (GAs) have been applied to select feature sets for the individual base classifiers. A theoretical essay regarding 4 different similarity measures in the context of decision templates may be found in [Kun01, KW01a]. Here it has been proven that all these measures are equivalent in the sense that they produce the same ordering of class labels if the individual classifier decisions sum up to the same value.

Some methods for the combination of multiple classifiers have also been used for the classification of temporal sequences. Based on the ideas of XU the combination of multiple experts in a text-dependent speaker identification system is proposed in [CXC96]. The system consists of a set of classifiers that are trained by using features extracted in sliding input-windows catching the temporal features. Different classifiers are generated by using different input-window sizes. Therefore, each classifier represents a specified time scale. The individual classifiers are combined with the naive Bayes combination.

A survey about neural fusion architectures for the classification of temporal sequences by using a set of multilayer perceptrons (MLPs) have been proposed in [TKP97, TW99]. In these studies it is systematically investigated at which stage of processing acoustic and optic information in speech recognition should be combined. Hereby for two layered MLPs the combination of acoustic and optic information can be accomplished at three levels: (1) combination of the input layers (early fusion), (2) com-

[2]The evaluated fusion schemes are: averaging, behaviour-knowledge space, Dempster-Shafer combination, fuzzy integral, majority voting, maximum, minimum, naive Bayes, probabilistic product, product and learning the classifier outputs with 4 different types of classifiers.

bination of the hidden layers (intermediate fusion), (3) combination of the output layers (late fusion). The temporal integration and the final decision is the task of an associative memory (AM).

Automated Bioacoustics

Most of the systems for the automated classification of animal vocalisations have been developed during the last ten years and are based on pattern recognition (PR) approaches including machine learning techniques, statistical hidden Markov models (HMM) and artificial neural networks (ANNs) [GTCW96, KM98, MMR98b]. All these techniques handle feature vectors extracted from the waveform. Unfortunately the time variability in the acoustic signals creates formidable difficulties for the convenient and effective application of these algorithms. Consequently, there exists a large variety of methods to lessen the natural time variability, e.g. linear time scale normalisation and dynamic time warping (see Section 2.2).

In the following some of the recent automated species recognition systems are mentioned, to give an overview of the current state of the art in the field of automated bioacoustics:

A. TAYLOR developed a software system consisting of a feature extractor and a decision tree classifier [Qui93] in order to classify the flight calls of 9 species of birds which are general indicators of diversity or ecological change [Tay95a, Tay95b]. The classification system makes no attempt to segment or isolate the individual calls. Hence, it works entirely by extracting a set of features from local peaks detected in short time spectra of the incoming signal. Due to the narrow bandwidth of the calls, frequency tracks are derivated by searching for sequences of local peeks in the spectrograms. The system has been tested on a set of 138 flight calls. 78 % of calls were identified correctly, 4 % incorrectly and 18 % were left unclassified.

A similar system is used by GRIGG et al. to classify the vocalisations of 22 species of frogs occurring in northern Australia [GTCW96]. The classifier system is used in unattended operation to monitor the sounds of frog populations. To obtain significant features sequential forward selection (SFS) was applied.

A software package for the automated identification of *Orthoptera* songs has been developed by CHESMORE et al. [CFS98]. The system automatically extracts various signal parameters, e.g. resonant frequencies, syllable length, tooth impact rates, etc. by using a digital signal processing card. Based on these signal parameters a blackboard system consisting

of a number of independent knowledge sources called experts, 17 species of British *Orthoptera* have been classified. The pre-recorded songs are from the cassette tape which accompanies a comprehensive guide to British *Orthoptera* [MH88].

Dynamic time warping (DTW) and hidden Markov Models[RJ93] have been used by KOGAN and MARGOLIASH [KM98] for the classification of bird songs (four zebra finches (*Taeniopygia guttata*) and four indigo buntings (*Passerina cyanea*)). In DTW they created templates for different sound intervals, e.g. song units, silent units and background noise. For the classification with HMMs they applied the hidden Markov toolkit which is primary designed for speech recognition. However, its 18 major tools and programs can be adapted and used for analysis and recognition of any acoustic time series, including the vocalisations of animals.

In MURRAY et al. [MMR98b] KOHONEN's self organising neural networks [Koh95] have been applied to analyse vocalisations of false killer whales (*Psendorca crassidens*). They used short time measurements of duty cycles and peak frequencies as input of the classifier. The weight vectors of the competitive neural networks were used to interpret different types of whistles, e.g. ascending whistles, high frequency pulse trains and low frequency pulse trains.

The self organising feature map combined with a source-filter model of the sound production of humpback whales is utilised by MERCADO et al. to classify the vocalisations of these animals [MK98]. Each individual vocalisation was characterised in terms of their pitch, duration, noisiness, and formant structure using a combination of linear prediction, cepstral processing, and manual measurements. Further preliminary work studying the structure of killer whale pulse trains can be found in [MMR98a].

In [PJ00, Par01b] PARSONS et al. utilised discriminant function analysis and multilayer backpropagation perceptrons to classify the search-phase echolocation calls of 12 species of British bats to species level. They extracted one temporal and four spectral features from each call. To reduce the dimensionality of the feature vectors of the frequency-time course of the individual calls, curve-fitting was applied. Discriminant function analysis achieved an classification accuracy of 79 %, while the multilayer backpropagation perceptrons achieved 87 %.

1.3 Major Contribution of this Thesis

Many contributions of our work were accepted for presentation and publication at various conferences [DSP01a, DSP01b, IRD01, RID01, DSRP01,

DSP02, DSP03a, DSP03b, SDP04] or were published in international journals [SD00, SDKP03, DPS03, DPRSnt]. The main contributions of this thesis can be summarised as follows:

1. A software package for the classification of *Orthoptera* sounds from 53 different species has been developed. The software package performs feature extraction, feature selection, classification and classifier fusion.

2. A signal segmentation algorithm for the detection of pulses of *Orthoptera* songs has been developed. This algorithm allows to extract the pulse positions in several time scales.

3. Three fusion schemes for the classification of time series have been proposed and discussed.

4. Decision templates have been applied to the classification of temporal sequences and algorithms to calculate multiple decision templates per class have been proposed. Two new methods called temporal decision templates and clustered decision templates have been developed.

5. Decision templates have been discussed in the context of supervised neural network training schemes by assuming the normalised correlation as similarity measure. In particular, links to the well established methods: Pseudoinverse matrix, linear associative memory, and naive Bayes decision fusion are given.

All methods investigated in this thesis, beside the linear vector quantisation (LVQ), the radial basis function (RBF) networks and parts of the feature selection algorithms have been implemented by the author of this thesis in C/C++ utilising the Matlab C and C++ libraries. Linear vector quantisation was applied with the well known LVQ_PAK [KHK+96] developed by KOHONEN which was also used for the initialisation of the prototype vectors of the RBF networks provided by F. SCHWENKER [SKP01]. Some of the feature selection algorithms have been developed in a practical course conducted by the author of the thesis [RBF+03].

1.4 Outline of the Dissertation

Chapter 1: "Introduction" has provided a motivation, a description of the DORSA project and reviews related work in the field of classifier fusion and automated classification of animal vocalisations.

The utilised classification methods, e.g. RBF neural networks, nearest neighbour classifiers and LVQ classifiers are described in **Chapter 2:** "Classification Methods". This Chapter also deals with multi-phase learning schemes for RBF networks by initialising the prototype vectors with classification trees. In addition, a introduction into time alignment and pattern matching is given.

Chapter 3: "Multiple Classifier Systems" contains a brief introduction into MCS and categorises algorithms for the combination of multiple classifiers. A general framework for the extraction of local features in time series and three architectures for the classification of time series are proposed. The main contribution of this thesis is described in Section 3.5 which deals with time series classification with multiple decision template approaches. In Section 3.6 links to supervised neural network learning schemes are investigated by assuming decision templates with the normalised correlation as similarity measure.

Automated song analysis and the sound production of the *Orthoptera* is the introduction of **Chapter 4:** "*Orthoptera* Bioacoustics" which deals with the classification of *Orthoptera* songs. This Chapter deals with the segmentation of the acoustic signals and the extraction of local features in great detail. The evaluation of single features and the automated selection of feature components is applied to determine feature sets for the experiments. Classification results for the proposed fusion schemes are given and discussed.

A description of a synthetic data set consisting of Gaussian distributions is given in **Chapter 5:** "Synthetic Data". The classification results for the proposed fusion methods are discussed.

Finally, **Chapter 6:** "Summary, Conclusions and Future Research" contains a survey of major results and discusses suggestions and ideas for future work.

Chapter 2

Classification Methods

This Chapter deals with methods for the classification static objects and temporal sequences. A presentation of the applied classifiers is given in Section 2.1 which also includes a discussion. Furthermore, this Chapter deals with time alignment and pattern matching (TAPM) methods (see Section 2.2) that are used in many applications (see Chapter 1). For the overall Chapter some basic knowledge of neural networks and speech recognition is advantageous.

2.1 Neural Networks

Pattern recognition (PR) and machine learning (ML) techniques are often components of *intelligent systems* and are used for pre-processing of data and decision making. Historically, the two major approaches to pattern recognition are the *statistical* (or decision theoretic) and the *syntactic* (or structural). During the last thirty years, the emerging technology of *artificial neural networks* (ANNs) has provided a third approach, especially for *black box* implementations of PR algorithms [Sch92]. ANNs are massively parallel computational schemes and the form of learning is closely related to classical approximation techniques, such as generalised splines and regularisation theory [PG90]. Hereby minimising the expected classification error is obviously an important issue of all approaches.

All three approaches share common features and common goals and so the boundaries between them are fading. Among the huge number of artificial neural network structures [Bis95, Bra95, Hay94, MPS+01], and more continue to appear as research continues, many of these structures have common *topological properties*, *unit characteristics* and *training approaches*. Basically, an ANN is characterised by three entities:

1. **The network topology or interconnection of the neural units**.
   Recurrent neural networks and non-recurrent neural networks are distinguished. An important sub-class of non-recurrent networks are feedforward neural networks in which cells are organised into

layers, and only unidirectional connections are permitted between adjacent layers [MSC91].

2. **The characteristics of the artificial neurons**.
   Different linear and non-linear transfer functions are considered, e.g. the heaviside function, the sigmoid function, the Fermi function and the identity function [Bra95]. Among the earliest models of neurons were the model of MCCULLOCH and PITTS [MP43] and FUKUSHIMA's cognitron [Fuk75].

3. **The strategy for pattern learning or training**.
   Supervised learning procedures , unsupervised learning procedures and reinforcement learning procedures are distinguished [Her02]. All these strategies differ in context of the learning algorithm, the training sample set and the problem to be solved.

All these entities have influence to the training convergence properties and the classification performance of the ANN.

2.1.1 Radial Basis Function Networks

Radial basis function (RBF) networks were introduced into the neural network literature by BROOMHEAD and LOWE in 1988 [BL88]. The theoretical basis of the RBF approach lies in the field of interpolation of multivariate functions $f : \mathbb{R}^D \rightarrow \mathbb{R}^L$. The goal of interpolating a set of M tuples $(\mathbf{x}^\mu, \mathbf{y}^\mu)_{\mu=1}^M$ is to find a function F with $F(\mathbf{x}^\mu) = \mathbf{y}^\mu$ for all $\mu = 1, \ldots, M$, where F is a linear combination of radial basis functions. Usually the RBF is defined as a nonincreasing function which takes the maximal value for $||\mathbf{x} - \mathbf{c}||_2^2 = 0$ [Kun00b]. The most popular and widely used RBF is the *Gaussian function*

$$\varphi_j(\mathbf{x}) = \exp\Big(-\frac{||\mathbf{x} - \mathbf{c}_j||_2^2}{2\sigma_j^2}\Big) \tag{2.1}$$

utilising the L_2-norm which is well known as *Euclidean distance*. More general, the L_p-norm between $\mathbf{x}$ and $\mathbf{c}_j$ [Ska97] with $p \in [1, \infty)$ is given by

$$||\mathbf{x} - \mathbf{c}_j||_p = \Big(\sum_{d=1}^{D} |\mathbf{x}_d - \mathbf{c}_{jd}|^p\Big)^{\frac{1}{p}}, \tag{2.2}$$

where $||\cdot||_1$ is called the *Manhattan distance* [BJ92]. Hereby the vector $\mathbf{x}$ of $\mathbb{R}^D$ is called *feature vector* which is typically from a continuous data input

space, $\mathbf{c}_j \in \mathbb{R}^D$ a so-called *center* or *prototype vector* and $\sigma_j \in \mathbb{R}$ a *scaling parameter* which determines how steeply $\varphi_j(\mathbf{x})$ decreases with growing distance from the center $\mathbf{c}_j$. The Gaussian function $\varphi_j(\mathbf{x})$ may be considered as a nonlinear *transfer function* $\varphi : \mathbb{R} \rightarrow (0, 1]$ of the j-th neuron in the hidden layer that maps the neuron input activation $||\mathbf{x} - \mathbf{c}_j||_2^2$ into the output signal. In practice vector or matrix valued scaling parameters σ_j (also called widths) may be used. For example, an RBF neuron with Gaussian transfer function can be defined with scaling parameters $\sigma_j = (\sigma_{j1}, ..., \sigma_{jD}) \in \mathbb{R}^D$ in the following way

$$\varphi_j(\mathbf{x}) = \exp\Big(-\sum_{d=1}^{D} \frac{||\mathbf{x}_d - \mathbf{c}_{jd}||_2^2}{2\sigma_{jd}^2}\Big) \tag{2.3}$$

where each σ_{jd} describes the shape of the j-th Gaussian function in the d-th feature dimension.

A RBF network which classifies an input feature vector $\mathbf{x}^\mu$ into one of L different classes consists of D input neurons, a layer of J non-linear hidden RBF neurons and L output neurons (see Figure 2.1).

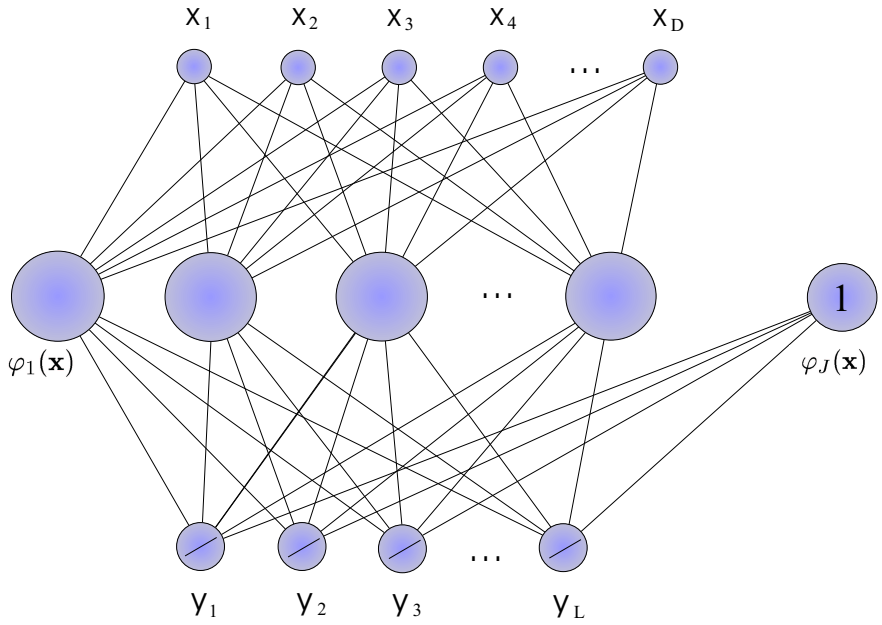


Figure 2.1: Feedforward structure of a radial basis function neural network with D input neurons, J RBF neurons in the hidden layer and L output neurons.

The output of the j-th RBF neuron is the value of the RBF $\varphi_j(\mathbf{x})$. Based on these outputs the activity of the l-th output neuron is then given by the

linear combination of RBFs

$$\mathbf{y}_l = \sum_{j=0}^{J} \mathbf{w}_{jl} \varphi_j(\mathbf{x}) \tag{2.4}$$

where $\mathbf{w}_{jl}$ is the synaptic weight from the j-th hidden neuron to the l-th output neuron, indicating the strength of its connections. Thus, a large positive value of $\mathbf{w}_{jl}$ indicates a strong *excitatory connection*, and a large negative value would be considered highly *inhibitory*. An additional neuron whose output is $\varphi_J(\mathbf{x}) = 1$ is utilised to gain the bias term of the individual output neurons (see Figure 2.1). Typically, in pattern recognition the individual network outputs (see Eq. 2.4) are interpreted as class membership estimates (see Eq. 3.3) and thus, the index of the output neuron with the maximal class membership estimate

$$\omega = \underset{l=1,\ldots,L}{\operatorname{argmax}}(\mathbf{y}_l) \tag{2.5}$$

specifies the class determined by the RBF network.

In the supervised learning procedure the RBF network is constructed by utilising a training data set of M examples $\mathcal{D} := \{(\mathbf{x}^\mu, \omega^\mu) | \mu = 1, ..., M\}$. Hereby the desired classifier output $\omega^\mu \in \Omega$ is one element from a finite set of L classes $\Omega = \{1, ..., L\}$ which is represented as binary coded unit vectors $\hat{\mathbf{y}}^\mu \in \{0, 1\}^L$ of length L and exactly a single one. The index l with $\hat{\mathbf{y}}_l^\mu = 1$ indicates that the input feature vector $\mathbf{x}^\mu$ should be categorised to class $\omega^\mu = l$.

Many strategies have been developed to adapt the RBF parameters and the synaptic weights in the second layer. The most popular training strategies which tune all parameters of the RBF network together in a single training phase are *backpropagation* (BP) *training* [Bra95], *expectation-maximisation* (EM) [Orr98], *orthogonal least squares* (OLS) [CCG91], *genetic algorithms* (GAs) [WC96] and *support vector machines* (SVM) [SSB+97]. But recently, *multi-phase training strategies* have been applied for the adaption of the network parameters. Most of them differ in the order of parameter adaption and the way in which the prototypes are initialised [MD89]. For the training of RBF networks we consider a very common training scheme called *three-phase learning* in which all parameters of the RBF network are adapted in three separate learning phases. The algorithm allows to use labelled and unlabelled training data for the initialisation of the prototype vectors. It performs the following three steps [SKP01]:

1. Training of the RBF kernel parameters (centers and widths) through a supervised or an unsupervised clustering procedure, e.g. vector quantisation and classification tree algorithms.

2. Supervised learning of the output layer of the network by gradient descent (see Eq. 2.7) or pseudoinverse solution [Hay94].

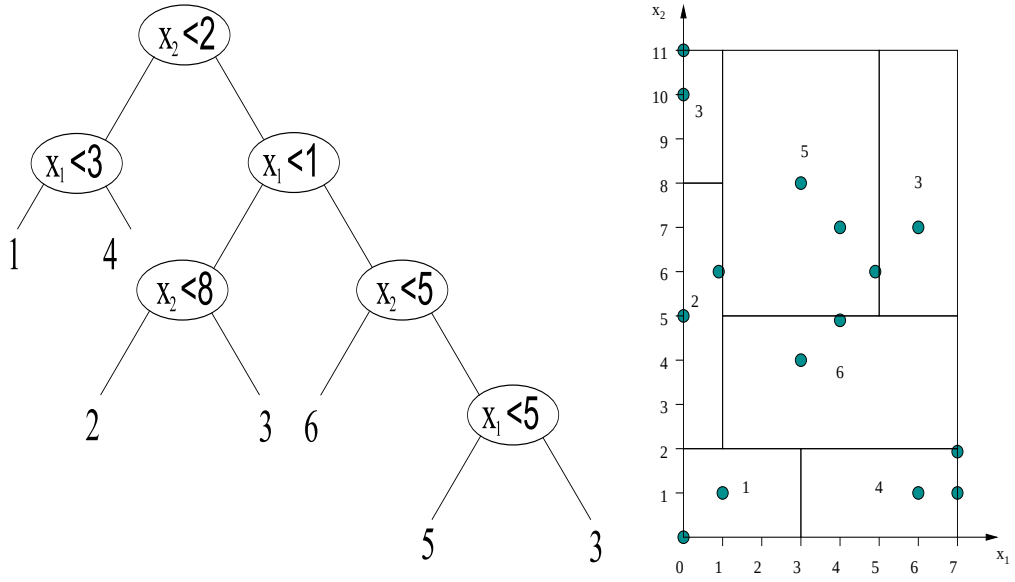
3. Training of all parameters of the RBF network simultaneously through a non-linear backpropagation-like optimisation procedure (see Eq. 2.7 and Eq. 2.8).

Hereby the radial basis function centers, the radial basis function widths and the synaptic weights of the output layer are adapted successively. RBF networks with free prototype vectors are shown to be universal approximators which are able to approximate each continuous function with arbitrary precision [PS93].

In the following we introduce the three-phase learning scheme in more detail by considering the supervised initialisation of the RBF kernel parameters with classification tree algorithms [HMS66] (see step 1) and the adaption of the RBF parameters (see step 2 and 3) by gradient descent. In contrast to many other methods to initialise the prototype vectors the classification tree algorithm also allows to adjust the RBF widths. A very popular initialisation scheme for the RBF centers (see step 1) with learning vector quantisation (LVQ) is discussed in Section 2.1.2 which deals with LVQ networks.

The initialisation of the radial basis function centers and scaling parameters by classification trees (see step 1) is introduced by KUBAT [Kub98]. Classification trees partition the feature space $\mathbb{R}^D$ into pairwise disjoint regions R^j, $j = 1, ..., J$. The binary classification tree is the most popular type in practice. Here each node has either two or zero children nodes. Each node in the classification tree is representing a region R of $\mathbb{R}^D$. If a node R has exactly two children then the regions represented by the children nodes R_{left} and R_{right} have the following properties: $R = R_{\text{left}} \bigcup R_{\text{right}}$ and $R_{\text{left}} \bigcap R_{\text{right}} = \emptyset$. If a node is a terminal node, called leaf, all data points within this region are classified to a certain class given at the terminal node. Usually, these regions are hyperrectangles parallel to the axes of the feature space. During the training phase the splitting points are determined. Features carrying the most information about the output tend to be splitted earliest and most often. This is some kind of automatic relevance determination which is a natural feature of classification trees [Orr99]. One of the most popular decision tree algorithm implementations is QUINLAN's C4.5 [Qui93].

Figure 2.2 shows this feature space partitioning procedure by example. Here a data set of $M = 15$ examples is shown (see Figure 2.2(b)). These data points are used to create the classification tree (see Figure 2.2(a)). The



(a) A binary decision tree of depth 4 with two dimensional features $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$. The classes are given at the leafs. Nodes within the tree are labelled with a feature dimension and a splitting point.

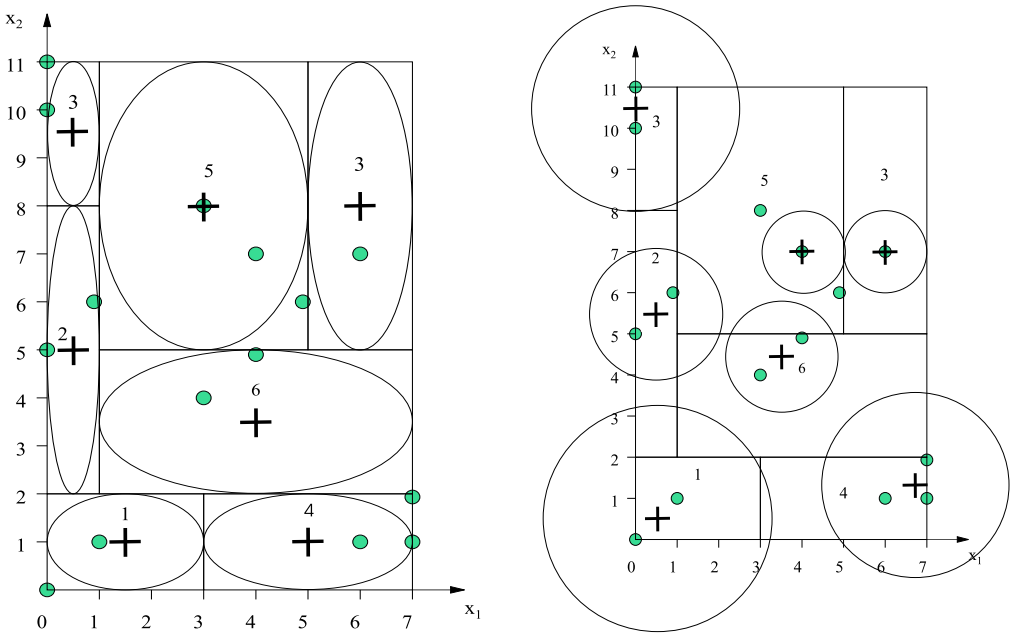
(b) Data points and regions (with class labels) of the $\mathbb{R}^2$.

Figure 2.2: Decision tree with classes at the leafs (Figure (a)) and the corresponding partition into hyperrectangles parallel to the axes of the feature space (Figure (b)). Boundary values, feature dimensions and class labels are given for the individual nodes. Each partition is determined through a single leaf of the decision tree.

corresponding regions are defined by the leafs of the classification tree (see Figure 2.2(b)). Each terminal node of the classification tree determines one rectangular region in the feature space $\mathbb{R}^2$. In the binary classification tree each branch is determined by a feature dimension $d \in \{1, ..., D\}$ and a boundary $B \in \mathbb{R}$. Typically, a class is represented in more than one leaf of the tree; for example class 3 in Figure 2.2 is represented in two different regions. A region R is defined by the path through the tree, starting at the root and terminating in a leaf.

In order to construct RBF networks with decision trees, for each region an RBF neuron is created. Hence, the number of leafs in the classification tree determines the number of hidden RBF neurons in the network. Due to the fact, that the size of the tree (model complexity) can be controlled

by the amount of pruning, the number of RBF neurons can be varied.



(a) The prototype vector of each RBF neuron is placed in the center of the region.

(b) The prototype vector of each RBF neuron is placed in the center of gravity of the feature vectors within each region.

Figure 2.3: Decision tree regions and data points of a small data set. The RBF centers (+) and the scaling parameters (shown through the shape of the RBFs) are exemplarily depicted for two different initialisation techniques.

In [Orr98] it is proposed to use the size and the position of each region (see Figure 2.2(b)) to calculate the prototype vectors and scaling parameters of the RBFs. Figure 2.3(a) for example shows prototype vectors placed in the centers of the individual regions and scaling parameters which have been defined in such a way that all basis functions φ_j, $j = 1, ..., J$ have the same value at the border of their region. But, also the data points within a particular region can be used for the calculation of the prototype vectors and the corresponding scaling parameters. For example in [SD00], we proposed to place the prototype of each region into the center of gravity of the data points within this region (see Figure 2.3(b)). The scaling parameter of each RBF is then set proportional to the Euclidean distance between the

center of gravity and its nearest neighbour prototype. Whereas these and further methods to initialise the RBF kernel parameters with decision trees are discussed in more detail in [SD00], [SKP01] contains a detailed numerical evaluation for multi-phase learning algorithms for RBF networks. In these studies the prototype initialisation has been performed by randomly chosen data points, $\hat{k}$-means, LVQ and classification trees.

For the second and the third learning step we suppose a training data set $\mathcal{D}$ of M examples where $\hat{\mathbf{y}}^\mu$ is the desired network output and $\mathbf{y}^\mu$ (see Eq. 2.4) is the network output for feature vector $\mathbf{x}^\mu$. Additionally, a differentiable error function as the *sum-of-squares error*

$$E := \sum_{\mu=1}^{M} ||\hat{\mathbf{y}}^\mu - \mathbf{y}^\mu||_2^2 \tag{2.6}$$

and a differentiable transfer function which is defined for D scaling parameters for each neuron, e.g. the Gaussian transfer function given in Eq. 2.3 is assumed.

Provided that the centers and the scaling parameters of the radial basis functions have been initialised, the weights of the output layer can be calculated (second learning step). An iterative procedure to minimise Eq. 2.6 is the gradient descent optimisation algorithm which adjusts the free parameters in such a way that the error function E is minimised. In the second learning phase the synaptic weights $\mathbf{w}_{jl}$ are the free parameters and the corresponding *batch learning rule* is given by

$$\mathbf{w}_{jl} = \mathbf{w}_{jl} - \eta_1 \frac{\partial E}{\partial \mathbf{w}_{jl}} = \mathbf{w}_{jl} + \eta_1 \sum_{\mu} (\hat{\mathbf{y}}_l^\mu - \mathbf{y}_l^\mu) \varphi_j(\mathbf{x}^\mu). \tag{2.7}$$

The third and final learning phase is applied to adapt the complete architecture by incrementally minimising the sum-of-squares error E by gradient descent. This leads to three types of batch learning rules for the synaptic weights $\mathbf{w}_{jl}$ (see Eq. 2.7), the prototype vectors $\mathbf{c}_{jd}$ and the scaling parameters σ_{jd} (see Eq. 2.8)

$$\begin{aligned} \mathbf{c}_{jd} &= \mathbf{c}_{jd} - \eta_2 \frac{\partial E}{\partial \mathbf{c}_{jd}} = \mathbf{c}_{jd} + \eta_2 \sum_{\mu} \frac{\mathbf{x}_i^\mu - \mathbf{c}_{jd}^\mu}{\sigma_{jd}^2} \varphi_j(\mathbf{x}^\mu) \sum_{l=1}^{L} (\hat{\mathbf{y}}_l^\mu - \mathbf{y}_l^\mu) \mathbf{w}_{jl} \\ \sigma_{jd} &= \sigma_{jd} - \eta_3 \frac{\partial E}{\partial \sigma_{jd}} = \sigma_{jd} + \eta_3 \sum_{\mu} \frac{(\mathbf{x}_i^\mu - \mathbf{c}_{jd}^\mu)^2}{\sigma_{jd}^3} \varphi_j(\mathbf{x}^\mu) \sum_{l=1}^{L} (\hat{\mathbf{y}}_l^\mu - \mathbf{y}_l^\mu) \mathbf{w}_{jl} \end{aligned} . \tag{2.8}$$

Hereby η_1, η_2 and η_3 are real-valued learning rates or step sizes for the adaption of the individual parameter types. More details about the derivation of these learning rules can be found in [Bis95, HKP91].

2.1.2 Learning Vector Quantisation

Clustering and vector quantisation techniques are usually unsupervised learning algorithms applied to divide data points into natural groups when no teacher signal is available [Pat96]. The result is a small number of representative centers (also denoted as prototypes) to minimise the quantisation error. A very established supervised clustering procedure which has been applied to initialise the RBF centers is learning vector quantisation (LVQ) [KHK+96] (see step 1 of the 3 phase learning algorithm). In LVQ the prototype vectors are properly placed within each zone such that the decision borders are approximated by the nearest neighbour rule in the Euclidean distance sense [HTS00].

In order to capture the entire LVQ network with the RBF network topology (see Figure 2.1), the prototype vectors are determined by LVQ and the synaptic weights in the hidden layer are connected one-to-one (see Figure 2.4). Again the network contains D input neurons, a layer of

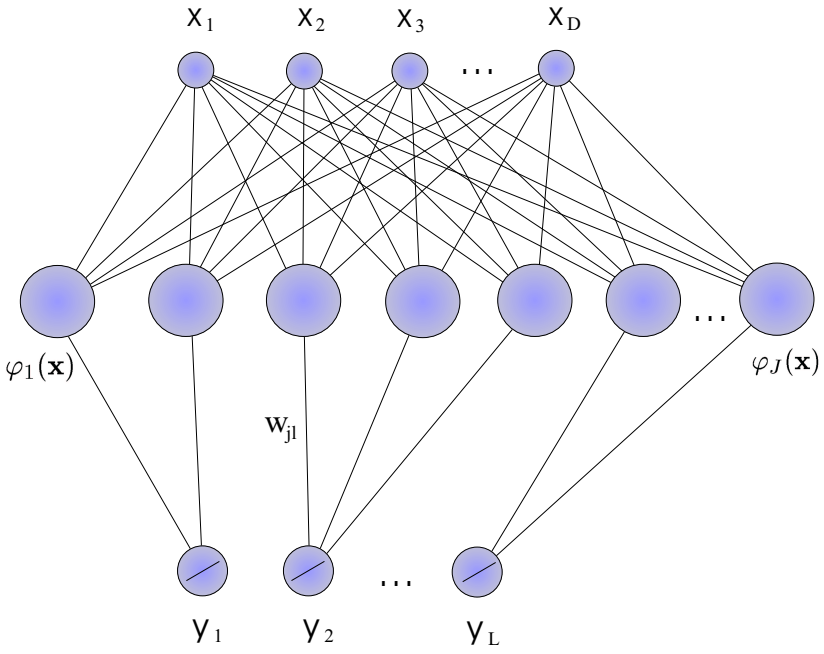


Figure 2.4: Feedforward structure of a LVQ neural network with D input neurons, J prototype neurons in the hidden layer and L output neurons. The class memberships of the prototype neurons are encoded in the weight matrix.

J prototype neurons and a layer of L output neurons. The prototype vectors $\mathbf{c}_1, ..., \mathbf{c}_J$ divide the input space into J disjoint regions called Voronoi

cells [Bra95]. In contrast to RBF networks the LVQ network is a *competitive neural network* trained by supervised learning in which each neuron has a binary output and one *active neuron* (or firing neuron) $\varphi_{j*}(\mathbf{x}) = 1$ is competitively selected. By assigning a data point $\mathbf{x} \in \mathbb{R}^D$ to the network $j*$ is determined by the *nearest neighbour rule* by calculating the Euclidean distance (see Eq. 2.2) between the data point $\mathbf{x}$ and the individual prototypes

$$\varphi_{j*}(\mathbf{x}) = 1 \Longleftrightarrow j* = \underset{j=1,...,J}{\operatorname{argmin}} ||\mathbf{x} - \mathbf{c}_j||_2^2. \tag{2.9}$$

All other neurons remain silent $\varphi_j(\mathbf{x}) = 0$, for $j \neq j*$.

To determine the class of the active neuron in supervised competitive learning networks, each prototype vector $\mathbf{c}_j$, $j = 1, ..., J$ is labelled with its class membership $\omega(\mathbf{c}_j) \in \Omega$. The classifier output is a binary unit vector which is given through

$$\mathbf{y}_l(\mathbf{x}) = \begin{cases} 1 & , l = \omega(\mathbf{c}_{j*}) \\ 0 & , \text{otherwise} \end{cases}, \quad l \in \Omega. \tag{2.10}$$

Equivalent, the class memberships of the individual prototype neurons may be encoded in the weight matrix by

$$\mathbf{w}_{jl} = \begin{cases} 1 & , \omega(\mathbf{c}_j) = l \\ 0 & , \text{otherwise} \end{cases} \quad j = 1, ..., J, \quad l = 1, ..., L. \tag{2.11}$$

After removing the connections with $\mathbf{w}_{jl} = 0$, each hidden neuron is connected with exactly one output neuron (see Figure 2.4). As in RBF networks the classifier output is then given by the linear combination (see Eq. 2.4).

For $\mathbf{c}^\mu = \mathbf{x}^\mu$, $\mu = 1, ..., M$ this is equivalent to the *nearest neighbour classifier* which is one of the most elegant and simplest classification techniques [Fuk90]. This classifier offers the advantage that it converges to the Bayes model under fairly nonrestrictive assumptions [DHS00] and the disadvantage that the computational cost for this classifier in the recall phase is rather high. Thus, in many prototype based classification paradigms the number of prototypes J is reduced. For example, the calculation of the prototypes for the *minimum distance classifier* is applied by averaging the feature vectors of the individual classes [KB98a]. This leads to $J = L$ prototypes and a decreased classification performance if one or more classes in the data set are represented by more than one cluster.

To calculate a problem specific number of prototypes in LVQ the number of prototypes J is reduced by clustering, thus $J \in [L, ..., M]$. For the adaption of these prototypes many LVQ variants, e.g. *Decision surface mapping* (DSM) [GS91] and *LVQ with training count* (LVQTC) [Odo97] have been published. Most of these algorithms are derived from the basic learning vector quantisation algorithm LVQ1 that also includes the classification by the nearest neighbour rule (see Eq. 2.9). But from the basic LVQ1 version, the OLVQ1, the LVQ2.1 and the LVQ3 clustering procedures have been suggested by KOHONEN. These algorithms have escaped much attention in the neural network and the pattern recognition society. The training is based on *Hebbian learning* which is a plausible biological learning model. According to the model, the connection between neurons that fire simultaneously is strengthened whilst the connection between neurons that act differently is loosened [Koh95]. In [KHK+96] the author describes a software package and recommends settings for learning rate parameters. It is also suggested, that learning be always started with the OLVQ1 algorithm, which has very fast convergence due to a optimised learning rate for each prototype (see Eq. 2.13). The recognition accuracy may be improved then by continuing the prototype adaption with LVQ2.1 or LVQ3.

For the supervised clustering with LVQ we again assume a finite training data set of M examples $\mathcal{D} := \{(\mathbf{x}^\mu, \omega^\mu) | \mu = 1, ..., M\}$. Hereby $\mathbf{x}^\mu \in \mathbb{R}^D$ is a feature vector and ω^μ the corresponding desired target class which is represented as a binary coded unit vector $\hat{\mathbf{y}}^\mu \in \{0, 1\}^L$ as used to train the RBF network. The algorithm starts with some initialisation of the prototypes $\mathbf{c}_j$, $j = 1, ..., J$, most often by assigning random values or by randomly taking J vectors from the training data set $\mathcal{D}$ [Kun00b]. Then in each point in time $t > 0$, randomly a feature vector $\mathbf{x}^{\mu_t}$ together with a teacher signal $\hat{\mathbf{y}}^{\mu_t}$ from the training data is chosen, $\mu_t \in \{1, ..., M\}$. By presenting the t-th feature vector $\mathbf{x}^{\mu_t}$ to the LVQ network the outputs of the individual neurons are calculated (see Eq. 2.9) and the network output is computed by the linear combination (see Eq. 2.4).

Let $j = j*$ be the index of the active neuron. Then by utilising the teacher signal $\hat{\mathbf{y}}^{\mu_t}$ in LVQ1 and OLVQ1 the prototype of the active neuron is adapted by

$$\mathbf{c}_j^{t+1} = \mathbf{c}_j^t + \eta_j^t \underbrace{(1 - ||\mathbf{y}^t - \hat{\mathbf{y}}^{\mu_t}||_2^2)}_{\gamma^t} (\mathbf{x}^{\mu_t} - \mathbf{c}_j^t), \tag{2.12}$$

where $\gamma^t = 1$ if the classification of $\mathbf{x}^{\mu_t}$ was correct, and $\gamma^t = -1$ if the classification was wrong. Consequently, the direction of the gradient update depends on the correctness of the classification. If the feature vector

$\mathbf{x}^{\mu_t}$ is correctly classified (network output equals teacher signal: $\mathbf{y}^t = \hat{\mathbf{y}}^{\mu_t}$) the prototype vector $\mathbf{c}_j$ is attracted towards to $\mathbf{x}^{\mu_t}$. Otherwise, the feature vector $\mathbf{x}^{\mu_t}$ has a repulsive effect to the prototype vector.

Whereas the learning rates η_j^t for the j-th neuron in LVQ1 are constant or monotonically decreasing, in OLVQ1 these learning rates are determined by the recursion

$$\eta_j^t = \min\left(\frac{\eta_j^{t-1}}{1 + \gamma^t \eta_j^{t-1}}, B\right) \tag{2.13}$$

where η_j^0 is considered as the initial learning rate [Koh90]. Due to the fact that η_j^t can also increase, for each η_j^t a upper bound B is determined. In the LVQ package B is set to $\eta_1^0 = \eta_2^0 =, ..., = \eta_J^0$ which is assumed in the range of $(0, 1)$.

In contrast to the LVQ1 and the OLVQ1 clustering procedure where only a single active neuron is determined to adapt the prototypes (see Eq. 2.9) in LVQ2.1 and the LVQ3 two prototypes with the minimal Euclidean distance to $\mathbf{x}^{\mu_t}$ have to be considered for the prototype adaption. These prototypes are determined by the 2-nearest neighbour rule, that takes the Euclidean distances between $\mathbf{x}^{\mu_t}$ and the 2 nearest prototypes into account. Additionally, the $\underline{K}$-nearest neighbour (K-NN) rule allows to calculate fuzzy classifier outputs and the classification with the K-NN rule is more robust against outliers in the prototypes. It works as follows:

For each feature vector $\mathbf{x}^{\mu_t}$ there is a sequence of decreasing distances, e.g. there is a sequence $(\tau_j)_{j=1}^J \subset \{1, ..., J\}$ with the property $||\mathbf{x}^{\mu_t} - \mathbf{c}_{\tau_1}^t||_2^2 \leq ... \leq ||\mathbf{x}^{\mu_t} - \mathbf{c}_{\tau_J}^t||_2^2$. The indices of the $K \leq J$ nearest neighbour prototype neurons to $\mathbf{x}^{\mu_t}$ are then given by

$$\mathcal{N}_K(\mathbf{x}^{\mu_t}) := \{\tau_1, ..., \tau_K\}. \tag{2.14}$$

Assuming the transfer function

$$\varphi_j(\mathbf{x}^{\mu_t}) = \begin{cases} 1 & , j \in \mathcal{N}_K(\mathbf{x}^{\mu_t}) \\ 0 & , \text{otherwise} \end{cases}, \quad j = 1, ..., J \tag{2.15}$$

leads to exactly K active neurons. If the weight matrix is defined as proposed in Eq. 2.11 the linear combination (see Eq. 2.4) counts for each class the number of active neurons.

In LVQ3 the prototype adaption is applied if at least one of the two nearest neighbours belongs to the correct class ω^{μ_t} and $\mathbf{x}^{\mu_t}$ falls into a zone

called window. Let $\mathcal{N}_2(\mathbf{x}^{\mu_t}) := \{\tau_1, \tau_2\}$ be the indices of the two nearest neighbours (see Eq. 2.14), $\mathbf{x}^{\mu_t}$ falls into the window if

$$\frac{||\mathbf{x}^{\mu_t} - \mathbf{c}^t_{\tau_1}||_2}{||\mathbf{x}^{\mu_t} - \mathbf{c}^t_{\tau_2}||_2} > \frac{1-W}{1+W}, \tag{2.16}$$

where W is a relative window width recommended in the range of 0.2 to 0.3. For class $\omega^{\mu_t} = l$ and network output $\mathbf{y}^t_l$, three cases have to be distinguished for the adaption of the prototype vectors $\mathbf{c}_{\tau_1}$ and $\mathbf{c}_{\tau_2}$

$$\mathbf{c}^{t+1}_{\tau_k} := \begin{cases} \mathbf{c}^t_{\tau_k} + \xi\eta^t_{\tau_k}(\mathbf{x}^{\mu_t} - \mathbf{c}^t_{\tau_k}), & \mathbf{y}^t_l = 2 \\ \mathbf{c}^t_{\tau_k} + \eta^t_{\tau_k}(1 - |\mathbf{y}^t_l - \hat{\mathbf{y}}^{\mu_t}_l|)(\mathbf{x}^{\mu_t} - \mathbf{c}^t_{\tau_k}), & \mathbf{y}^t_l = 1 \\ \mathbf{c}^t_{\tau_k}, & \mathbf{y}^t_l = 0 \end{cases}, \quad k \in \{1,2\}. \tag{2.17}$$

Hereby $\xi \in [0,1)$ lessons the prototype adaption if $\mathbf{x}^{\mu_t}$, $\mathbf{c}_{\tau_1}$ and $\mathbf{c}_{\tau_2}$ belong to the same class. For $\xi = 0$ the LVQ3 algorithm is equivalent to LVQ2.1 because in LVQ2.1 there is no prototype adaption if both prototypes belong to the same class. Further details about the individual LVQ clustering algorithms can be found in [Koh90, Koh95, KHK+96].

2.1.3 Fuzzy-K-Nearest Neighbour Classifiers

To determine the membership of an input vector $\mathbf{x}$ to each class in Ω, a *fuzzy-K-nearest-neighbour rule* may be applied[Sin98]. One simple way to calculate such membership values is by setting the remaining network connections in Eq. 2.11 to $\frac{1}{K}$ instead of 1 and using the K-NN rule to calculate the LVQ network outputs. This is equivalent to a combining rule for fuzzy K-nearest neighbour classifiers as suggested by JÓŹWIK in [J83]. Nevertheless, many authors also include the distance between $\mathbf{x}$ and the K nearest prototype vectors to calculate these membership estimates [VC98]. In the context of this thesis we consider two different fuzzy-K-nearest-neighbour rules which perform such distance weighting.

Type 1:

In order to calculate fuzzy classifier outputs for each of the K nearest neighbours $\mathcal{N}_K(\mathbf{x})$ to $\mathbf{x}$ (see Eq. 2.14), the distance between the feature vector $\mathbf{x}$ to the prototypes of the nearest neighbours is calculated by a monotonically decreasing transfer function, as for example given by [Sin00]

$$\varphi_j(\mathbf{x}) = \begin{cases} \left(\frac{||\mathbf{x}-\mathbf{c}_j||_2}{\sigma} + \alpha\right)^{-1} & , j \in \mathcal{N}_K(\mathbf{x}) \\ 0 & , \text{otherwise} \end{cases}. \tag{2.18}$$

The parameter $\alpha > 0$ is used to grade low distance values for $||\mathbf{x}-\mathbf{c}_j||_2$. Another widely used monotonically decreasing transfer function is the Gaussian function

$$\varphi_j(\mathbf{x}) = \begin{cases} \exp(-\frac{||\mathbf{x}-\mathbf{c}_j||_2^2}{\sigma}) & , j \in \mathcal{N}_K(\mathbf{x}) \\ 0 & , \text{otherwise} \end{cases}, \tag{2.19}$$

which is also widely used in RBF networks (see Eq. 2.1). For both transfer functions $\sigma > 0$ determines the slope of the transfer function and $\varphi_j(\mathbf{x})$ may be considered as membership value of $\mathbf{x}$ to the class of the j-th prototype. By applying Eq. 2.4 the individual membership values of the K active neurons are accumulated for each class individually. The support for the hypothesis that l is the true class label of $\mathbf{x}$ is then estimated by the normalised network output which is given through

$$\mathcal{C}_l(\mathbf{x}) := \frac{\mathbf{y}_l(\mathbf{x})}{\sum_{l=1}^{L} \mathbf{y}_l(\mathbf{x})}. \tag{2.20}$$

Now we have $\mathcal{C}_l(\mathbf{x}) \in [0,1]$ and $\sum_{l=1}^{L} \mathcal{C}_l(\mathbf{x}) = 1$ and call the classifier probabilistic (see Eq. 3.4) [Kun00a].

Type 2:

For each class $\omega \in \Omega$ let J^ω be the number of hidden neurons with class label ω, $J = \sum_\omega J^\omega$. Then for input $\mathbf{x}$ and class ω there is a sequence $(\tau_j^\omega)_{j=1}^{J^\omega}$ with $||\mathbf{c}_{\tau_1^\omega} - \mathbf{x}||_2 \leq ... \leq ||\mathbf{c}_{\tau_{J^\omega}^\omega} - \mathbf{x}||_2$. The indices of the K nearest prototype neurons ($K \leq J^\omega$) of $\mathbf{x}$ to class ω are then given by

$$\mathcal{N}_K^\omega(\mathbf{x}) := \{\tau_1^\omega, ..., \tau_K^\omega\}. \tag{2.21}$$

Then the whole set of active neurons is determined through

$$\mathcal{N}_K(\mathbf{x}) := \{\mathcal{N}_K^1(\mathbf{x}), ..., \mathcal{N}_K^L(\mathbf{x})\}. \tag{2.22}$$

After applying one of the transfer functions (see Eq. 2.18 and Eq. 2.19), and the linear combination (see Eq. 2.4) the class membership estimates $\mathbf{y}_1(\mathbf{x}), ..., \mathbf{y}_L(\mathbf{x})$ are calculated. Normalisation by Eq. 2.20 again leads to L probabilistic classifier outputs $\mathcal{C}(\mathbf{x}) = (\mathcal{C}_1(\mathbf{x}), ..., \mathcal{C}_L(\mathbf{x}))$ for the hypothesis that the true label of $\mathbf{x}$ is l. For both fuzzy-K-nearest neighbour rules the parameter K determines the degree of fuzziness of the classifier outputs [Sin98].

2.1.4 Summary and Discussion

The preceding Section gives an introduction into prototype based classifiers, e.g. RBF networks, K-NN classifiers and LVQ networks. We have discussed three-phase RBF network training in detail and introduced two algorithms (initialisation with decision trees and learning vector quantisation) for the initialisation of the first layer of the RBF network (see step 1). For the second and the third learning phase we introduced an iterative parameter adaption method called gradient descent. In order to calculate fuzzy membership values the fuzzy-K-NN rule is proposed.

Similarities between these network structures are shown by embedding the LVQ1, the OLVQ1 the LVQ2.1 and the LVQ3 network into the RBF network topology and discussing the training of these ANNs by using a training set $\mathcal{D}$ which contains binary coded class labels. Comparisons between these two ANN architectures can be made by considering the three entities which characterise an ANN (see Section 2.1). Although the network topology is almost similar, differences can be found for example in the weight matrix. Whereas in LVQ networks each prototype is firmly associated with a single class (see Figure 2.4) in RBF networks each prototype is associated to each class in Ω (see Figure 2.1). In addition, the scaling parameters of the individual RBF functions are adapted in the training phase.

The strategy for pattern learning for RBF networks may be applied by several methods as referred in Section 2.1.1. But for LVQ networks, the training algorithms are explicitly given. Both networks are not based on the approximation of the density functions of the class samples. In RBF networks a very common method for the adaption of the parameters is based on the minimisation of the squared error function by gradient descent (see Eq. 2.7 and Eq. 2.8). On the other hand in LVQ networks the adaption of the prototypes is applied to define the class borders according to the nearest-neighbour rule [KHK+96].

A further link between LVQ and RBF networks is given if the initialisation of the RBF parameters in the first phase of the multi-phase learning algorithm is applied with learning vector quantisation, e.g. LVQ1, ..., LVQ3 [SKP01]. Similarities concerning the learning rules of RBF networks and LVQ networks are discussed in [SKPH94].

2.2 Time Alignment and Pattern Matching

Time alignment and pattern matching (TAPM) is an important problem involved in many applications for the classification of temporal sequences. Many algorithms have been developed and have been evaluated in the most of the domains discussed in Chapter 1. However, for the *audio domain*, (e.g. speech recognition, sound identification, bioacoustics, etc.) the temporal alignment of feature vectors seems to be important and thus, a lot of methods have been developed. In speech recognition it comes from the fact that acoustic feature streams (e.g. a sequence of spectra) of a speech utterance (e.g. word, phrase or sentence) are seldom realised at the same speed across the entire utterance [MSC91]. Among the large variety of methods we group these into:

1. Methods which make use of knowledge or assumptions about the temporal sequence (time scale normalisation, trace segmentation, feature set transformations and dynamic time warping [RJ93]).

2. Methods which are able to learn temporal contexts (time delay neural networks, partially recurrent neural networks and hidden Markov models [ST95]).

For methods in group 2 the classifier is modified in such a way that it is able to learn temporal variations, most of the methods in group 1 apply a pre-processing of the feature vectors and usually work in conjunction with classifiers for static objects. Nevertheless, in each of the referred methods the sequential order of features is taken into account.

2.2.1 Pre-Processing of Features

In the pre-processing stage a sequence of feature vectors is transformed to serve as classifier input. Hereby the dimensionality of the feature vectors is fixed or the number of feature vector is decreased. Obvious methods are *time scale normalisation* and *feature set transformations*.

For the first class of algorithms linear and non-linear methods to transform variable-length feature vectors into a fixed number of classifier inputs are distinguished. Several linear methods are applied to expand or compress feature vectors of variable length into fixed length feature vectors. Most of these methods are selecting the desired number of features at equally spaced intervals. A simple and effective algorithm for linear time scale normalisation used in conjunction with neural networks is presented

in [Woo90]. However, time scale normalisation can be applied by extracting a fixed number of features by varying the time resolution during the feature extraction.[1] Hereby the basic assumption is, that the variation of the features is proportional to the number of features which depends on the duration of the utterance.

A non-linear sampling method used to reduce the length of a discrete feature stream is *trace segmentation* (TS) [CS96]. TS is based on the assumption, that despite timing differences for a feature stream of the same category, fluctuations in the features over time will occur in the same sequence, but over different length of time. To introduce this method we consider an example from the field of speech recognition. Let $(S(\jmath))_{\jmath=1}^{\mathcal{J}}$ with $S(\jmath) \in \mathbb{R}^D$ be a stream of power spectra. Where there is no change in frequency there will be a high density of points and where the frequency changes are rapid the points will be widely spaced. The number of feature vectors in such a sequence can be reduced by removing those which occur during the stationary portions of the patterns. This may be done by computing a trace of Euclidean distances (see Eq. 2.2) between two adjacent spectra $\hat{S}(\jmath) = ||S(\jmath) - S(\jmath+1)||_2, \quad \jmath = 1, ..., \mathcal{J}-1$. In order to calculate a feature stream of constant length $(\hat{S}(\jmath))_{\jmath=1}^{\mathcal{J}-1}$ is split into $G \leq \mathcal{J}-1$ intervals such that the intervals are representing approximately equivalent amounts of change. By maintaining the temporal order of the intervals the temporal order of the samples from the original feature stream $(S(\jmath)_{\jmath=1}^{\mathcal{J}})$ is obtained as well.

The second class of pre-processing algorithms are *feature-set transformations* [MSC91] which map variable-size feature vectors into a fixed number of parameters. The most popular transformation is the discrete Fourier transformation (DFT) which is often applied in conjunction with a filter-bank (see Section 4.2.2) [VHH98]. Hereby, care must be taken to ensure that the representations have a physically meaningful interpretation. For example, in speech recognition the filter-banks are typically nonuniform. The number of frequency channels defines the dimensionality of the feature vectors and the spacing between the filters including the bandwidth of the individual filters defines the critical band. All these parameters have been determined in perceptual studies. Accessorily, it is intended to determine the frequency channels in such a way that they give equal contribution to speech articulation. Most popular variants are the *mel scale* and the *bark scale* [RJ93].

Further well known feature-set transformations are the autocorrelation

[1] In Section 4.2.5 time scale normalisation is applied by calculating sonograms with a fixed number of spectra.

analysis [Wol99] which is used to measure the periodicity of a stationary signal and the Wavelet transformation [Dau01].

2.2.2 Dynamic Time Warping

A very general time alignment and normalisation scheme is *dynamic time warping* (DTW) which was introduced by SAKOE and CHIBA in 1978 [SC78]. DTW allows to be problem specific adjusted by modifying several constraints. This allows to use DTW in various fields, e.g. bioinformatics, robotics, chemical engineering and speech processing [CKHP02].

The classic DTW algorithm uses a local distance measure to measure the distance between a candidate sequence and a set of template sequences by calculating a *warping path*. Not surprisingly, multiple templates per class are commonly used in DTW systems [VHH98]. Suppose we have a candidate sequence $(\mathbf{x}(g))_{g=1}^{G}$ and a template sequence $(\mathbf{c}(h))_{h=1}^{H}$ with $\mathbf{x}(g) \in \mathbb{R}^D$ and $\mathbf{c}(h) \in \mathbb{R}^D$. To align these two sequences a local distance measure $\phi(g,h) = \phi(\mathbf{x}(g), \mathbf{c}(h))$ between two points of these sequences is applied to calculate a warping path on a $G \times H$ plane. Typically, the Manhattan distance or the Euclidean distance (see Eq. 2.2) is used to measure these local distances. The unknown warping path

$$W = \{w(g(q), h(q)) | q = 1, ..., Q\} \tag{2.23}$$

with $g(q) \in \{1, .., G\}$ and $h(q) \in \{1, .., H\}$ is then given by a sequence of corresponding monotonically non-decreasing pairs of indices. Figure 2.5 shows a warping path in the $G \times H$ plane and a linear path calculated by linear time scale normalisation. The overall distance Φ between the candidate sequence and the template sequence is then calculated by adding the local distances over the warping function path W. One natural and popular choice of finding the best alignment between the candidate sequence and the template sequence is to search the path with the minimal distance Φ as the minimum over all possible warping paths, such that

$$\Phi = \underset{W}{\operatorname{argmin}} \Big(\sum_{q=1}^{Q} \phi(g(q), h(q)) \Big). \tag{2.24}$$

This minimisation problem is solved to evaluate the dissimilarity between the candidate and each of the template sequences. Hereby the specific form of the accumulated distortion of Eq. 2.24 allows that *dynamic programming techniques* [Bel57] are readily applicable in the solution process. In order to obtain suggestive solutions, *warping constraints* are included

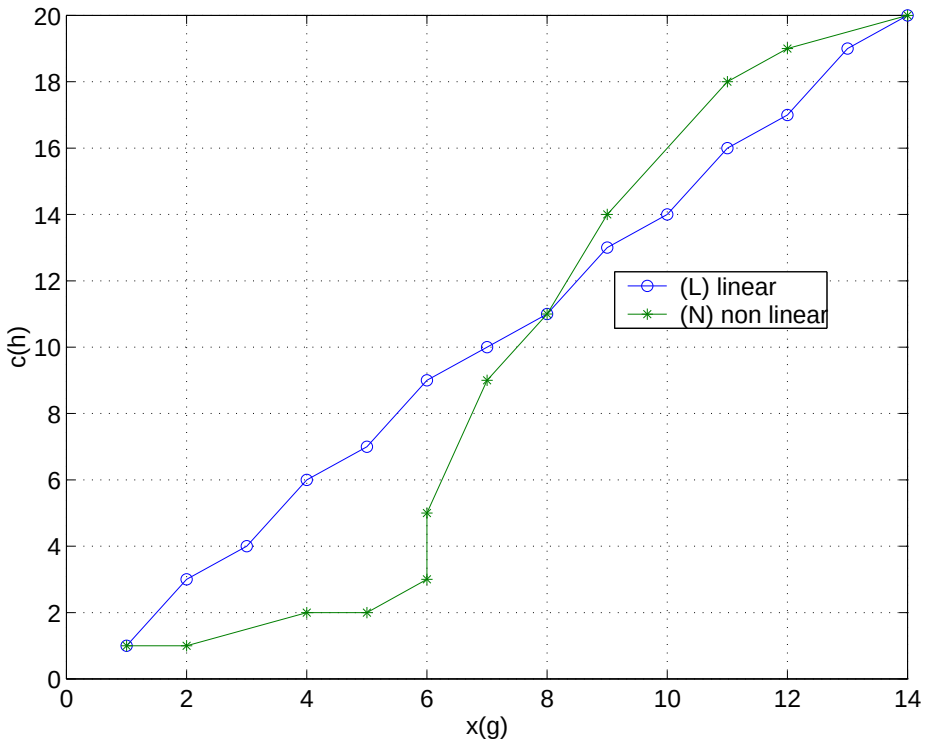


Figure 2.5: An example of dynamic time scale normalisation of two sequential patterns $((\mathbf{x}(g))_{g=1}^{G} \in \mathbb{R}^{14}$ and $((\mathbf{c}(h))_{h=1}^{H} \in \mathbb{R}^{20}$. The linear path (L) is determined by linear time scale normalisation and the non-linear path (N) is determined by dynamic time warping.

for the solution of the minimisation problem in Eq. 2.24. Among the large number of constraints the following are considered:

- **Endpoint constraints**.
  These constraints may be used if the endpoints of the candidate sequence and the template sequence are given, $g(1) = 1$, $h(1) = 1$, $g(Q) = G$ and $h(Q) = H$.

- **Monotonicity conditions**.
  These conditions are crucial in time alignment to maintain the temporal order of the sequences, $g(q+1) \geq g(q)$ and $h(q+1) \geq h(q)$.

- **Local continuity constraints**.
  They are used to constrict the number of allowable paths and to determine the degree of non-linearity, e.g. $g(q+1) - g(q) \leq \alpha$ and $h(q+1) - h(q) \leq \alpha$ with $\alpha \in \mathbb{N}^{+}$.

A detailed introduction into DTW including further warping constraints, e.g. global path constraints and slope weighting can be found in [RJ93].

2.2.3 Time Delay Neural Networks

Time delay neural networks (TDNN) have been successfully used for time series classification, e.g. phonem classification [WHH+89] and the recognition of hand gestures [YA99]. A TDNN is a multilayer feedforward network that uses time delays between all layers to represent temporal relationships between events in time. The network input is a temporal sequence of feature vectors, where only the features within a small window serve as network input at one point in time. During training and classification the time window is shifted and the next portion of the input sequence is given to the network until the whole sequence of features has been scanned through.

Because of the time delays the TDNN integrate activity from adjacent time-delayed vectors, which allows each vector to be weighted separately at several snapshots in time. This results in a temporal integration which is not linear; adjacent feature vectors are treated as additional inputs occurring at the same time [MSC91].

The TDNN is trained with a unique implementation of the backpropagation (BP) algorithm which uses copies of the network weights and the activities. This means that during the training phase, for each shift in time, copies of the TDNN parameters have to be accomplished. Network copies allow the entire history of the network activity to be present at once, thereby the BP algorithm computes the error for each network copy as they represent separate instances in time. Changes of the weights are calculated for each copy and the average of change in weights over all copies is used to update the weights.

Training is completed, when each TDNN copy will have almost identical weight matrices. This allows to use a single TDNN copy for the recognition. The trade-off is that multiple copies of the network make the training computationally more expensive. Furthermore, the number of iterations required to reach the minima increases [WHH+89].

2.2.4 Partially Recurrent Neural Networks

Partially recurrent neural networks are characterised by a feedforward network architecture serving as an associator, but also including a set of carefully chosen feedback connections to allow the representation of tem-

poral context. The recurrent connections are interconnected with so called *context units* which provide the network with a dynamic memory that holds on to relevant past events [WE95].

JORDAN [Jor86] described a two layer feedforward neural network architecture which involves recurrent links from the output neurons to the context cells in the hidden layer. Each context cell contains a self-recurrent connection to control how fast information about internal state representations decay [WAP99]. This architecture was modified by ELMAN [Elm90] by connecting the hidden units with the context cells and removing the self-recurrent connections of the context cells (see Figure 2.6). For this architecture the context units remember the previous internal state. The hidden units have the task of mapping both, the external input and also the previous internal state to some desired output.

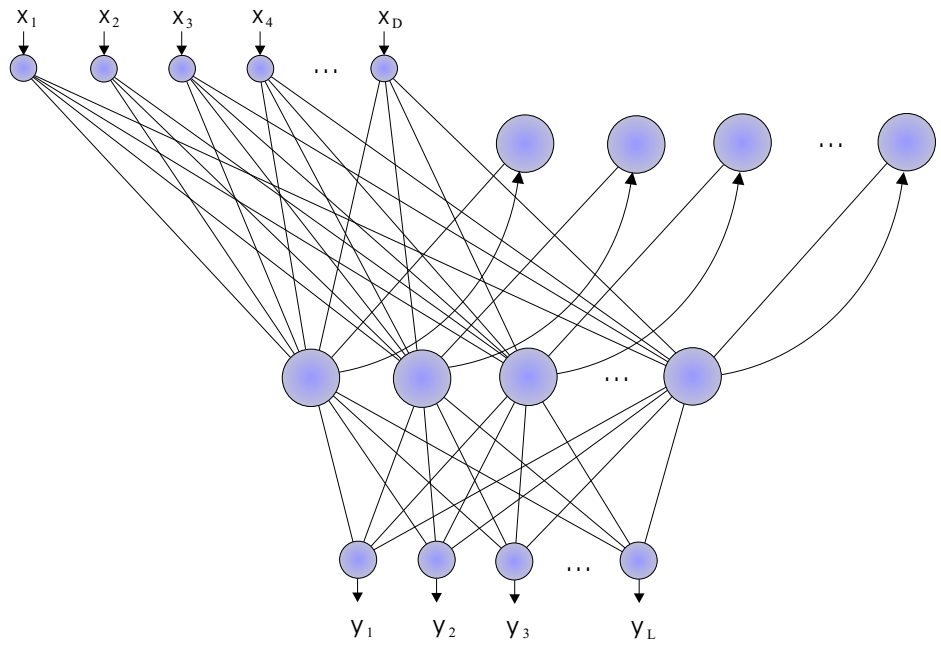


Figure 2.6: Architecture of an Elman neural network. The backward connections copy activations from the hidden layer to the context layer on a one-for-one basis with fixed weights of 1.0.

As the feedback connections are fixed at 1.0, supervised backpropagation learning can be used for the training of the feedforward connections. In the learning phase the network output is compared with a teacher input and backpropagation of error is used to incrementally adjust connection strengths [BJ92].

2.2.5 Hidden Markov Models

The basic theory of hidden Markov models was published in a series of papers by BAUM et all. [BPSW70]. They are the most successful TAPM algorithm for speech recognition based on words or to sub-word units (such as phoneme, diphone or triphone) and they are nowadays used in most speech recognition systems. Motivated by the success of HMMs, they have been combined with ANNs and a number of "HMM-like" implementations of ANNs have been developed [MSC91].

A hidden Markov model is a double embedded stochastic process with an underlying stochastic process that is not directly observable (it is hidden) [ST95]. It consists of a set of distinct states, a state-transition probability distribution, an observation symbol probability distribution and an initial state distribution. Generally the states are interconnected in such a way that any state can be reached from any other state (fully connected HMM). But in speech recognition, the so called left-right model or Bakis model of the HMM is mainly utilised because the underlying state sequence associated with the model has the property that, as time increases, the state index increases or stays at the same state.

In the training phase a labelled *observation sequence* is used to adjust the model parameters, e.g. the state-transition probabilities, the observation symbol probabilities and the initial state distribution. This sequence is called training sequence and is obtained at regularly spaced, discrete times. Usually, for each class a separate HMM is designed. There is no known way to analytically solve for the model parameter set, but iterative procedures such as the Baum-Welch method [Pre01] are used in many applications.

In the classification phase the model is given and a unknown sequence of regularly spaced, discrete observations is fed into each of the hidden Markov models. For each HMM the probability that the observed sequence was produced by the underlying model is computed. Therefore the best state sequence has to be calculated from the observation sequence, which is typically done by using the Viterbi algorithm [Vit67]. The class of the HMM whose probability is the largest is the classification result.

2.2.6 Summary

In the preceding Section the concept of time alignment algorithms and classification of temporal sequences was introduced. Several standard methods for the *pre-processing of features*, e.g. linear time scale normalisation, trace segmentation and feature-set transformations have been intro-

duced. Furthermore, methods which are able to *learn temporal contexts*, e.g. time delay neural networks, partially recurrent neural networks and hidden Markov models have been described. As well, dynamic time warping was explained which allows to be problem specific adjusted by a large number of warping constraints.

Although a large number of TAPM algorithms exists, in speech recognition systems one of the key questions is still, how speech patterns have to be compared to measure their similarity [RJ93]. Most of these algorithms have not been used in this thesis. But many TAPM approaches can be found in Section 4. For example, a linear time scale normalisation technique based on sonograms with a fixed number of sampling points have been introduced in Section 4.2.5. Additionally, a linear temperature normalisation technique is used to lessen the influence of the environment temperature to several signal parameters of the *Orthoptera* (see Section 4.1.6).

However, even temporal contexts have been trained by extracting a set local feature vectors inside a sliding short time window (see Figure 4.2.5). Learning structured classifier outputs is supported by the MDT and the CDT approach proposed in Section 3.5.2.

Chapter 3

Multiple Classifier Systems

This chapter deals with time series classification with multiple classifier systems (MCS). It introduces with static and adaptive fusion schemes and the graduation of MCS by four factors. A presentation of MCS approaches for time series classification based on local classifier decisions is given. This includes three static architectures (see Section 3.4) and three trainable approaches derived from the decision template algorithm (see Section 3.5). Furthermore, learning of decision templates is introduced and discussed in the context of neural network training schemes. In particular, links to the well established methods: The pseudoinverse matrix, the linear associative memory, and naive Bayes decision fusion are given.

3.1 Introduction

The combination of multiple classifiers may generate more precise classification results than single classifiers [KBD01]. Combining classification powers of a team of classifiers is therefore regarded as a general problem in various pattern recognition applications. Several experimental and analytical investigations have been made [Bre96, KHDM98, Kun01, TD01]. The two main paradigms for combining classifiers are [WKB97]:

- **Classifier fusion**.
  In the *ensemble approach* the classifiers are competitive experts. The predictions of the single classifiers are combined to yield a single class prediction [XKS92].

- **Classifier selection**.
  The classifiers are complementary experts (also called *modular approach*). Here a function is defined which dynamically selects for each pattern a single classifier from the team to reveal a class prediction [HAK01, JJNH91]. Two dynamic classifier selection methods based on the general framework of statistical decision theory can be found in [GR00].

Actually in both paradigms it is not accounted what internal structure a classifier has and on what theory and methodology it bases. Rather, in the field of MCS a classifier is simply regarded as a function box that receives an input sample and outputs a classification result [XKS92].

Let $\mathcal{C} = \{\mathcal{C}^1, ..., \mathcal{C}^N\}$ be a set of N classifiers[1] and $\Omega = \{1, ..., L\}$ be a set of L class labels. Each classifier gets as input a feature vector $\mathbf{x}_n \in \mathbb{R}^{D_n}$ from a continuous data input space (often D_n is constant) and assigns it to a class label from Ω. We distinguish 3 types of classifiers according to the levels of information available at the classifier output:

- **Crisp classifiers**. These classifiers $\dot{\mathcal{C}}^n$, $n \in 1, ..., N$ are given through the mapping

  $$\dot{\mathcal{C}}^n : \mathbb{R}^{D_n} \rightarrow \Omega. \tag{3.1}$$

- **Ranked classifiers**. A ranked classifier returns a list of $R \leq L$ disjoint class labels in descending order of a certain similarity score. This is conform to a mapping [Par01a]

  $$\vec{\mathcal{C}}^n : \mathbb{R}^{D_n} \rightarrow (\omega_1, ..., \omega_R) \tag{3.2}$$

  with $\omega_r \in \Omega$, $r = 1, ..., R$.

- **Soft classifiers**. A soft classifier is any classifier which uses fuzzy sets either during its training or during its operation [KBS98]. We divide soft classifiers into two groups: possibilistic classifiers and probabilistic classifiers. The *possibilistic classifier* is given through the mapping

  $$\tilde{\mathcal{C}}^n : \mathbb{R}^{D_n} \rightarrow [0, 1]^L \setminus \{0\}, \quad n \in 1, ..., N. \tag{3.3}$$

  In the term of probabilistic classifier fusion the soft classifier outputs are usually normalised to be interpreted as probabilities [TDvB97, TvBDK00]. This type of classifier is then called *probabilistic classifier* which is conform to a mapping

  $$\mathcal{C}^n : \mathbb{R}^{D_n} \rightarrow \Delta, \quad n \in 1, ..., N \tag{3.4}$$

  where the set $\Delta \in \mathbb{R}^L$ is defined through

  $$\Delta := \{(\mathbf{y}_1, ..., \mathbf{y}_L) \in [0, 1]^L | \sum_{l=1}^{L} \mathbf{y}_l = 1\}. \tag{3.5}$$

[1] These classifiers are usually denoted as *base classifies* as well as *first level classifiers*.

For soft classifiers the classifier output determines the degree of support for the hypothesis that $\mathbf{x}_n$ belongs to the respective class [Kun00b]. Then $\tilde{\mathcal{C}}_l^n(\mathbf{x}_n)$, $l \in \Omega$ is the estimated membership of feature vector $\mathbf{x}_n$ to class l obtained from the n-th classifier in the team. In probabilistic approaches this evidence may be additionally decomposed into the true posterior probability $P(\omega = l|\mathbf{x}_n)$ and the error term $\epsilon_l(\mathbf{x}_n)$ [KHDM98, OT01]

$$\mathcal{C}_l^n(\mathbf{x}_n) = P(\omega = l|\mathbf{x}_n) + \epsilon_l(\mathbf{x}_n). \tag{3.6}$$

By assuming $\epsilon_l(\mathbf{x}_n) = 0$, in probabilistic approaches the outputs of the individual base classifiers are combined (see Section 3.4.2). Hereby the crisp classifier $\dot{\mathcal{C}}$ is considered as a special case of the soft classifier, because the crisp decision can be determined by the maximum membership rule (see Eq. 2.5) [DT00]. But in the context of this thesis, crisp classifier decisions are often considered as binary unit vectors. This allows to use the same *aggregation operator* for both types of classifiers (see Section 3.4). Let $\omega \in \Omega$ be the class estimate of a crisp classifier, then the binary unit vector $(\mathbf{y}_1, ..., \mathbf{y}_L) \in \Delta \cap \{0,1\}^L$ with exactly a single one is given by

$$\mathbf{y}_l := \begin{cases} 1, & \omega = l \\ 0, & \text{otherwise} \end{cases}, \quad l = 1, ..., L. \tag{3.7}$$

The ranked classifier $\vec{\mathcal{C}}$ may also be considered as a special case of the soft classifier. Let $\mathcal{C}^n(\mathbf{x}_n) = (\mathcal{C}_1^n(\mathbf{x}_n), ..., \mathcal{C}_L^n(\mathbf{x}_n))$ be the soft classifier outputs of $\mathcal{C}^n$ given the feature vector $\mathbf{x}_n$. Then we assume these outputs sorted in decreasing order, e.g. there is a sequence $(\tau_l)_{l=1}^L$, $\tau_l \in \Omega$ of indices such that $\mathcal{C}_{\tau_1}^n(\mathbf{x}_n) \geq ... \geq \mathcal{C}_{\tau_L}^n(\mathbf{x}_n)$. For $\mathbf{x}_n \in \mathbb{R}^{D_n}$ the output of the ranked classifier is then given by $\vec{\mathcal{C}}^n(\mathbf{x}_n) = (\tau_1, ..., \tau_R)$ with $R \leq L$. By setting $R = 1$ the ranked classifier is equivalent to the crisp classifier.

Due to the fact that each classifier type may be represented through the soft classifier (see Eq. 3.3) the combination of the individual base classifiers can be applied by a combination of soft decisions. For such a combination in many publications the *decision profile* is considered [Kun00b]. For an input $\mathbf{x}_1, ..., \mathbf{x}_N$ with $\mathbf{x}_n \in \mathbb{R}^{D_n}$, the individual rows of the decision profile contain the outputs of the base classifiers $\mathcal{C}^1, ..., \mathcal{C}^N$. It is given by the $(N \times L)$-matrix

$$\mathcal{P}(\mathbf{x}_1, ..., \mathbf{x}_N) := \begin{bmatrix} \mathcal{C}_1^1(\mathbf{x}_1) & \dots & \mathcal{C}_l^1(\mathbf{x}_1) & \dots & \mathcal{C}_L^1(\mathbf{x}_1) \\ \dots & \dots & \dots & \dots & \dots \\ \mathcal{C}_1^n(\mathbf{x}_n) & \dots & \mathcal{C}_l^n(\mathbf{x}_n) & \dots & \mathcal{C}_L^n(\mathbf{x}_n) \\ \dots & \dots & \dots & \dots & \dots \\ \mathcal{C}_1^N(\mathbf{x}_N) & \dots & \mathcal{C}_l^N(\mathbf{x}_N) & \dots & \mathcal{C}_L^N(\mathbf{x}_N) \end{bmatrix}. \tag{3.8}$$

By assuming the decision profile, fusion schemes may be discriminated in the way in which the soft decisions are combined. For *class-conscious* fusion schemes the classifier outputs of a single class $l \in \Omega, \mathcal{C}_l^1(\mathbf{x}_1), ..., \mathcal{C}_l^N(\mathbf{x}_N)$ are combined through a mapping

$$\mathbf{z}_l := \mathcal{F}(\mathcal{C}_l^1(\mathbf{x}_1), ..., \mathcal{C}_l^N(\mathbf{x}_N)) \tag{3.9}$$

in order to calculate the combined membership estimate $\mathbf{z}_l \in [0, 1]$ for class l. This means that the individual columns of the decision profile are combined independently. But, there is a class of more general fusion schemes called *class-indifferent* where dependencies between class-wise classifier outputs may be considered for the combination of the N base classifiers. The combination is applied through the mapping [Kun00b]

$$\mathbf{z} := \mathcal{F}(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^N(\mathbf{x}_N)), \tag{3.10}$$

where the vector $\mathbf{z} \in [0, 1]^L \setminus \{0\}$ contains the combined membership estimates for each class in Ω. In literature these mappings are termed as *fusion mapping*, *combining rule* or *aggregation operator*.

A simple class-indifferent combining rule, applied in many applications and investigated in several theoretical considerations is *majority voting* [Kun00b], where the class which receives the largest number of votes is selected as the consensus decision. In majority voting the classifier ensemble returns a crisp decision as a binary coded unit vector which is given by

$$\mathcal{F}(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^N(\mathbf{x}_N))_l = \begin{cases} 1 & , l = \underset{\omega \in \Omega}{\operatorname{argmax}} \left(\sum_{n=1}^{N} \mathcal{C}_\omega^n(\mathbf{x}_n) \right) \\ 0 & , \text{otherwise} \end{cases}. \tag{3.11}$$

Unless for class l there is a ensemble output with $\sum_{n=1}^{N} \mathcal{C}_l^n(\mathbf{x}_n) > \frac{L}{2}$, the ensemble outputs of the individual classes have to be compared (see Eq. 3.11) to determine the final classification result.

In addition, classifier ensemble methods are distinguished in the context of their combination strategy. Typical classifier fusion or combination strategies are based on *fixed fusion mappings* like averaging, majority voting or multiplying the classifier outputs [DSP01a, KHDM98]. These combination rules are also called *simple fusion rules* [KM02] and they do not require any time consuming training of the classifier ensemble. However, learning strategies with an *adaptive fusion mapping* are also quite popular and widely used [XKS92]. These learning strategies are also called *trainable fusion strategies*.

For fixed fusion mappings the training of the base classifiers $\mathcal{C}^1, ..., \mathcal{C}^N$ is applied in a single phase by training the individual base classifiers utilising a training data set $\mathcal{D}$. The test of the classifier ensemble is done by using a separate test set $\mathcal{D}^{\mathrm{T}}$ and combining the individual classifier outputs utilising a static mapping which is often denoted as *aggregation rule*.

In contrast to fixed fusion mappings for adaptive fusion mappings the training of the overall classifier system is considered as a two phase learning problem:

1. Building the *classifier layer* consisting of a set of base classifiers where each classifier is trained on a specific feature subset.

2. Training of the *fusion layer* performing a mapping of the classifier outputs (soft or crisp decisions) into the set of desired class labels.

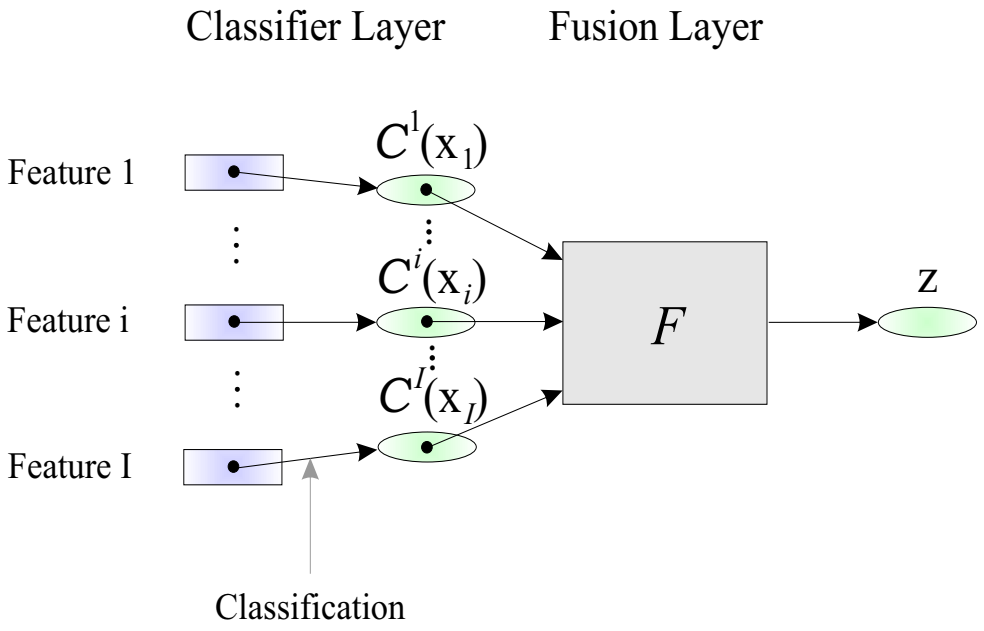


Figure 3.1: Two layer architecture of a MCS consisting of a classifier layer and an additional fusion layer. The combination of the classifier decisions of the individual base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ is accomplished by the fusion layer.

So, the overall classifier system (see Figure 3.1) is a two layer architecture very similar to layered neural networks, with a two layer structure such as multilayer perceptrons and radial basis function networks (see

Section 2.1.1). Whereas all parameters of the multilayer perceptrons are trained simultaneously by a backpropagation-like training algorithm in a one phase learning procedure, this 2-layered classifier architecture is trained by two completely different training phases. This type of classifier training is similar to radial basis function learning procedures, where the adaption of the whole network is performed in two or maybe three learning phases (see Section 2.1.1). Usually, for such multi-phase learning schemes a labelled training data set $\mathcal{D}$ is applied to the RBF network to calculate the radial basis function centers, the radial basis function widths and the weights of the output layer. However, other learning schemes may be applied. For instance, if a large set of unlabelled data points is available, this data can be used to initialise the RBF kernel parameters of the first layer through unsupervised clustering [SD00, SKP01].

Similar to this idea, for the training of the two layer fusion architecture, two labelled data sets ($\mathcal{D}$ and $\mathcal{D}^{\mathrm{V}}$) are applied to train the whole architecture, see Figure 3.2.

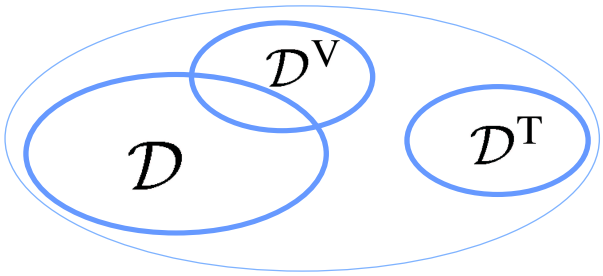


Figure 3.2: The training data set $\mathcal{D}$ is used to train the base classifiers. An additional data set (validation set $\mathcal{D}^{\mathrm{V}}$) is utilised to adapt the classifier ensemble, and $\mathcal{D}^{\mathrm{T}}$ is the disjoint test set to evaluate the performance of the classifier ensemble.

This means, that first the base classifiers are trained by a labelled training set $\mathcal{D}$, and then a labelled validation set $\mathcal{D}^{\mathrm{V}}$ (usually different form $\mathcal{D}$) is applied to the previously trained base classifiers to calculate the individual classifier outputs. These classifier outputs together with the desired class labels are used to train the decision fusion mapping performed by the fusion layer.[2] The basic idea of this approach is to implement prior knowledge of the base classifier based on its characteristic output behaviour into the fusion layer. Hereby it is assumed, that the validation set $\mathcal{D}^{\mathrm{V}}$ reflects the distribution of the pattern space.

[2] In some publications a trainable fusion layer is denoted as second layer classifier.

Such prior knowledge is for example given in the *confusion matrix* calculated by classifying feature vectors from the validation set $\mathcal{D}^{\mathrm{V}}$. Many well known algorithms for the combination of multiple classifiers use confusion matrix data for the combination of multiple classifiers [XKS92] (see Section 3.6). Then $\mathcal{D} \cup \mathcal{D}^{\mathrm{V}}$ may be considered as the training set of the whole classifier ensemble. Different training and validation set compositions may be determined through an overlap parameter

$$\rho = \frac{|\mathcal{D} \cap \mathcal{D}^{\mathrm{V}}|}{|\mathcal{D}^{\mathrm{V}}|} \in [0, 1] \tag{3.12}$$

which determines the fraction of data points of $\mathcal{D}^{\mathrm{V}}$ which are also in $\mathcal{D}$. Hereby $\rho = 0$ denotes $\mathcal{D} \cap \mathcal{D}^{\mathrm{V}} = \emptyset$ and $\rho = 1$ denotes $\mathcal{D}^{\mathrm{V}} \subset \mathcal{D}$.

To give a summary of classifier fusion and selection algorithms with respect to the mentioned factors several of such algorithms are systematised in Table 3.1.

classifier	fusion mapping	
	fixed	adaptive
crisp	Majority voting [CC00]$^{\mathbf{IF}}$	Behaviour knowledge space (BKS) [HS95]$^{\mathbf{IF}}$
		Naive Bayes fusion [XKS92]$^{\mathbf{IF}}$
ranked	Borda count [Ho01]$^{\mathbf{CF}}$	Generalized borda count [Par01a]$^{\mathbf{IF}}$
soft	Percentiles [SDKP03]$^{\mathbf{CF}}$	Neural networks [TW99]$^{\mathbf{IF}}$
	Ordered weighted averaging [CF97]$^{\mathbf{IF}}$	Pseudoinverse matrix [Koh95]$^{\mathbf{IF}}$
	Averaging [TDvB97]$^{\mathbf{CF}}$	Decision templates [KBS98]$^{\mathbf{IF}}$
	Fuzzy integral [CK95]$^{\mathbf{CF}}$	Wernecke's method (BKS+) [Wer92]$^{\mathbf{IF}}$
	Probabilistic function [End01]$^{\mathbf{CF}}$	A priori selection [GR00]$^{\mathbf{S}}$
	Product rule [TvBDK00]$^{\mathbf{CF}}$	A posteriori selection [GR00]$^{\mathbf{S}}$
	Random subspace [Ho98]$^{\mathbf{CF}}$	Gating network [JJNH91]$^{\mathbf{S}}$

Table 3.1: Classifier combination algorithms grouped by the 4 mentioned factors: (1) classifier type (crisp, ranked or soft), (2) combination strategy (fixed or adaptive), (3) fusion mapping (class-conscious (**C**) or class-indifferent (**I**)) and (4) combining type (fusion (**F**) or selection (**S**)).

However, it must be mentioned that for classifier selection methods, e.g. the a priori selection, the a posteriori selection, and the gating network a discrimination in class-conscious and class-indifferent fusion mappings is senseless because the overall decision is given by the selected classifier. Consequently, each of the listed adaptive classifier fusion methods is class-indifferent.

3.2 MCS Design Methodology and Terminology

A large number of classifier fusion and selection algorithms are known and are used to combine classifiers.[3] Independent from the combining algorithm the following *combining paradigms* are distinguished:

1. **Combination of different classifiers**.
   The classifier ensemble contains several classifiers of different architecture [TD01, WJP00] (decision trees, K-nearest neighbour classifiers, neural network approaches, etc.) or classifiers of different model size (multi-layer neural networks with different number of hidden layers, hidden neurons, etc.) or classifiers trained on different initial conditions (e.g. neural networks with random weight initialisation [Ho01]).

2. **Combination of classifiers trained on different data sets**.
   Several classifiers, each usually trained on randomly generated sets produced by re-sampling from a larger training set are combined to perform the classification or regression task. Prominent examples are *stacking* [WM96], *bagging* [Bre96] and *boosting* [Fre95].

3. **Combination of classifiers trained on different feature subsets**.
   Here the input space is divided into several subspaces. The input of each subspace is classified separately, and the combination is accomplished through a fusion scheme (see Table 3.1).

Broadly speaking, we have two large groups of combining paradigms, namely (1) *synthetic and structural paradigms* where non-homogeneous classifiers are combined and (2, 3) *feature based paradigms* where homogenous classifiers are trained on different features.

For each combining algorithm (see Table 3.1) and for each combining paradigm the term *classifier diversity* [KWSD00, OT01, SK02] and *weak classifier* (e.g. weak learner) [Fre95, Sku01] seems to be an important issue to build the final classifier system, the so-called *strong classifier* [MLR01]. Both terms are not clearly defined, but it seems that the combination of multiple classifiers is a trade-off between weak classifiers and the diversity between these classifiers.

Actually, one would expect that the combination of accurate and independent classifiers would offer the best classification performance. Unfortunately, the reported experimental and theoretical results have shown that the creation of accurate and diverse classifiers is a very difficult task

[3] In Table 3.1 several well known classifier fusion and selection algorithms are given.

because the error dependence between the individual base classifiers is typically high. Therefore, it seems to be essential to find a team of error independent weak classifiers. A classifier is called weak because it is not expected to classify the training data very well [MLR01]. We call a classifier weak if:

- Low capacity, simple form [GVB+92, Mel00]
- Low classification accuracy [HS90]
- Variance based (low bias, high variance) [GBD92]
- It can discriminate features only locally in the feature space [Kun00a]
- It uses only a subset of the training data [Bre96]

In particular, it has been shown that the combination of weak classifiers making independent errors can offer dramatic improvements in performance [HS90]. Accordingly, error independence among the individual classifiers is commonly regarded as a requirement for effective classifier fusion. Theoretical considerations can be found in [FHT98, SS99].

In bagging (combining paradigm 2), for example, several base classifiers are weakened by using only a subset of randomly chosen feature vectors for the training of the individual base classifiers. Due to the fact that the base classifiers are trained on different feature vectors the individual classifiers become diverse. Many measures have been proposed to measure the diversity between these classifiers [KW01b][4]. Furthermore, in [SK02] the ensemble accuracy is studied in the context of the diversity between the individual base classifiers.

Thus, the object in designing a multiple classifier system is to build an ensemble of base classifiers with maximal accuracy, or at least higher accuracy than the single best classifier. Building strong base classifiers is certainly not the topic of this research field.

Approaches of the third combining paradigm, the temporal combination of classifier decisions from different feature spaces, are discussed in the following two Sections. In these approaches a set of feature vectors is derived from a local window covering a small range of the time series.

[4] The diversity measures proposed in [KW01b]: Q statistic, correlation coefficient, disagreement measure, double fault measure, entropy measure, measure of difficulty, Kohavi-Wolpert variance, measurement of interrater agreement, generalised diversity and coincident failure diversity.

Then these local feature vectors are classified and a local decision is calculated. These local decisions are combined through a temporal aggregation rule to determine the final decision [DSP01a]. In the case of temporal decision fusion the diversity of the classifier outputs depends on the temporal variation of the feature vectors. This temporal combination problem is involved in many applications, e.g. speech recognition [RJ93], person identification [KHDM98], etc.

3.3 Feature Extraction in Time Series

In principle there are two different types of features which may be extracted from time series [DSP01b]:

1. **Global features**.
   These features are based on global characteristics or information of the whole time series, e.g. the mean frequency, mean energy, etc. (see [DSRP01] and Section 4.2.6).

2. **Local features**.
   These features are derived from subsets of the whole time series (see Figure 3.3), which are usually defined through a local time window $W^\jmath$. In the context of bioacoustic time series the time windows are located at detected pulses (see algorithm SEGMENTATION in Figure 4.15). Moving the window over the whole time series (see Figure 4.16) leads to a sequence of feature vectors called *feature stream*.

Local features may be grouped into two classes: (1) The features are extracted in regular time intervals and (2) the features are extracted in irregular time intervals. For both classes correlations among local observations have been observed in many applications [DPS03].

In the following, the extraction of local features from multiple feature spaces is discussed. Hereby, it must not be concerned if the features are extracted from regular or irregular time intervals. The situation is illustrated in Figure 3.3 where a window $W^\jmath$ covering a small part of the time series $s(t)_{t=1}^{T}$ is moved over the whole time series. Each time series is labelled with its corresponding class label $\omega \in \Omega$. For each window $W^\jmath$, $\jmath = 1, ..., \mathcal{J}$ a set of I features $\mathbf{x}_i(\jmath) \in \mathbb{R}^{D_i}$, $i = 1, ..., I$ and $D_i \in \mathbb{N}$, is extracted from the time series. Typically $\mathcal{J}$, the number of time windows varies from time series to time series.

For a time series $(s(t))_{t=1}^{T}$ this leads to I feature streams $(\mathbf{x}_i(\jmath))_{\jmath=1}^{\mathcal{J}}$ of constant length. Thus, for the temporal integration of the feature streams two approaches may be considered:

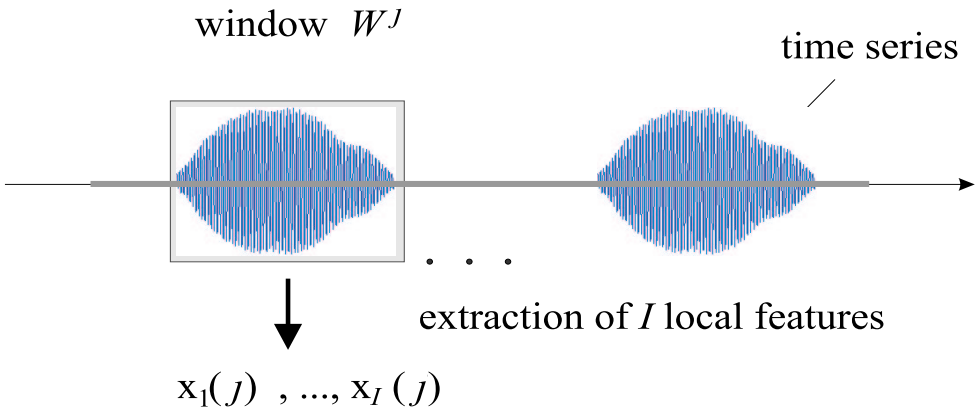


Figure 3.3: A set of I features $X(j) = (x_1(j), ..., x_I(j))$ is extracted from the j-th local time window W^j.

- **Temporal integration of local decisions from single features**.
  Local class decisions of each local feature are temporally integrated to calculate the final decisions for each of the feature streams. The final decisions based on the individual features are combined to calculate the final classification result (see Architecture CTF in Section 3.4).
- **Temporal integration of local decisions from locally combined features**.
  The local decisions for a set of local features are calculated. These local class decisions are temporally integrated over the whole time series in order to calculate the final decision (see Architecture CFT and FCT in Section 3.4).

Hereby for the second approach it must be bear in mind that the number of feature vectors per feature stream must be equal [DSP01a].

3.4 Three Architectures for the Classification of Time Series

Three static architectures for the combination of local classifier decisions are the basis for further research on the temporal combination of features from different feature spaces.

A. Architecture CFT (see Figure 3.4(a))
   In this architecture the classification of the time series is performed in the following three steps:

1.) Classification of single feature vectors (C-step)
For each feature $\mathbf{x}_i(j)$, $i = 1, ..., I$ derived from the local time window W^j a classifier $\mathcal{C}^i$ is given through a mapping (see Section 3.1)

$$\mathcal{C}^i : \mathbb{R}^{D_i} \rightarrow \Delta \tag{3.13}$$

where the set Δ is defined in Eq. 3.5. Thus, for each time window W^j, $j = 1, ..., \mathcal{J}$, I classification results $\mathcal{C}^1(\mathbf{x}_1(j)), ..., \mathcal{C}^I(\mathbf{x}_I(j))$ based on the individual features $\mathbf{x}_1(j), ..., \mathbf{x}_I(j)$ are calculated.

2.) Fusion of the local decisions (F-step)
For each time window W^j the I classification results are combined into a local decision $\mathbf{z}_A^j \in \Delta$ through a fusion mapping $\mathcal{F} : \Delta^I \rightarrow \Delta$

$$\mathbf{z}_A^j := \mathcal{F}(\mathcal{C}^1(\mathbf{x}_1(j)), ..., \mathcal{C}^I(\mathbf{x}_I(j))), \quad j = 1, ..., \mathcal{J} \tag{3.14}$$

which calculates the fused classification result based on I decisions.

3.) Temporal fusion of decisions over the whole time series (T-step)
The combination of the local decisions of the whole set of time windows W^j, $j = 1, ..., \mathcal{J}$ is given through

$$\mathbf{z}_A^o := \mathcal{F}(\mathbf{z}_A^1, ..., \mathbf{z}_A^{\mathcal{J}}) \tag{3.15}$$

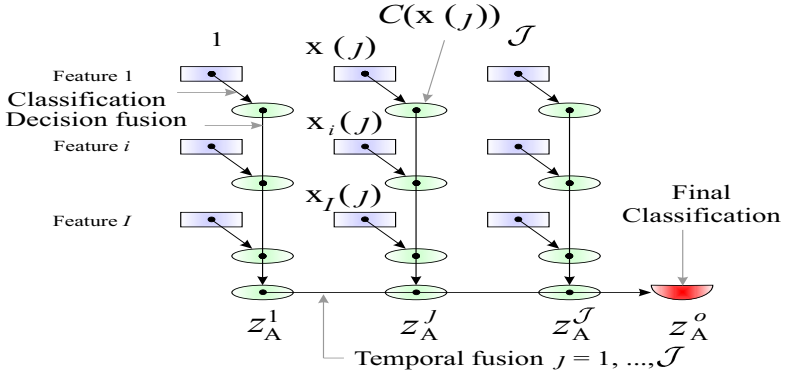
hereby $\mathcal{F} : \Delta^{\mathcal{J}} \rightarrow \Delta$ is an aggregation rule.

B. Architecture FCT (see Figure 3.4(b))
Here the classification of the whole time series is determined through the classification of the combined features and decision fusion over the whole time series:

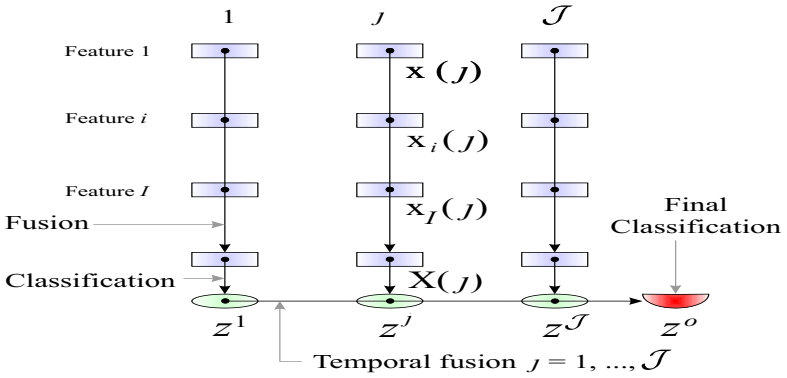
1.) Fusion of feature vectors (F-step)
Here the extracted features $\mathbf{x}_1(j), ..., \mathbf{x}_I(j)$ inside the time window W^j are simply concatenated into a single feature vector $X(j) = (\mathbf{x}_1(j), ..., \mathbf{x}_I(j)) \in \mathbb{R}^{\Phi}$, with $\Phi = \sum_{i=1}^{I} D_i$.

2.) Classification (C-step)
The concatenated feature vector $X(j)$ is classified into $\mathbf{z}_B^j \in \Delta$ using a classifier mapping $\mathcal{C} : \mathbb{R}^{\Phi} \rightarrow \Delta$.

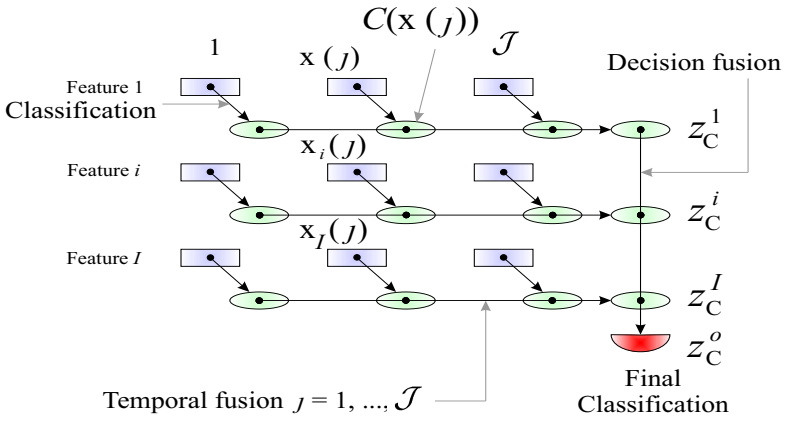
$$\mathbf{z}_B^j := \mathcal{C}(X(j)), \quad j = 1, ..., \mathcal{J} \tag{3.16}$$



(a) CFT - Classification, fusion of decisions, Temporal integration



(b) FCT - Fusion of features, Classification, Temporal integration.



(c) CTF - Classification, Temporal integration, fusion of decisions.

Figure 3.4: Three architectures for the classification of time series.

3.) Temporal fusion of decisions over the whole time series (T-step)
Here the integration of the local decisions is defined through

$$\mathbf{z}_B^o := \mathcal{F}(\mathbf{z}_B^1, ..., \mathbf{z}_B^{\mathcal{J}}) \tag{3.17}$$

again $\mathcal{F} : \Delta^{\mathcal{J}} \rightarrow \Delta$.

C. Architecture CTF (see Figure 3.4(c))
The final classification result is determined through temporal fusion followed by decision fusion.

1.) Classification of each of the I feature vectors $\mathbf{x}_1(\jmath), ..., \mathbf{x}_I(\jmath)$ within $W^{\jmath}$ (C-step)

$$\mathbf{x}_i(\jmath) \rightarrow \mathcal{C}^i(\mathbf{x}_i(\jmath)) \in \Delta \tag{3.18}$$

2.) Temporal fusion of the classifier outputs based on the individual features $i = 1, ..., I$ (T-step)

$$\mathbf{z}_C^i := \mathcal{F}(\mathcal{C}^i(\mathbf{x}_i(1)), ..., \mathcal{C}^i(\mathbf{x}_i(\mathcal{J}))) \in \Delta \tag{3.19}$$

again $\mathcal{F}$ is a fusion mapping $\mathcal{F} : \Delta^{\mathcal{J}} \rightarrow \Delta$.

3.) Fusion of decisions over all I feature spaces $\mathbf{z}_C^i$ (F-step)

$$\mathbf{z}_C^o := \mathcal{F}(\mathbf{z}_C^1, ..., \mathbf{z}_C^I) \tag{3.20}$$

here $\mathcal{F}$ is a fusion mapping $\mathcal{F} : \Delta^I \rightarrow \Delta$.

Different aggregation rules may be assumed for the integration of classifier decisions over time (see Eq. 3.15, 3.17 and 3.19) and the combination of classifier decisions over the feature space (see Eq. 3.14 and 3.20). For simplicity we do not discriminate between fusion over time and fusion over the feature space. Let us assume N, the number of classifiers which is given by the number of classifier decisions to be combined. For the temporal integration $N = \mathcal{J}$ and $\mathcal{F} : \Delta^{\mathcal{J}} \rightarrow \Delta$ is proper fusion mapping whereas for decision fusion over the feature space $N = I$ and $\mathcal{F} : \Delta^I \rightarrow \Delta$ is the applied mapping.

In many applications, the combination of such probabilistic classifier decisions is applied by using a static combining scheme [KB98b]. For this type of combination we consider four types of combining rules: (1) *averaging* (see Eq. 3.21), (2) *probabilistic approaches*, e.g. the probabilistic function (see Eq. 3.28), (3) *percentiles* (see Eq. 3.29) and (4) *voting* (see Eq. 3.30). A general theoretical framework of classifier combination strategies based on the Bayesian decision rule can be found in [KHDM98].

3.4.1 Average Fusion

The combination of the individual classifiers $\mathcal{C}^1, ..., \mathcal{C}^N$ by averaging the classifier decisions is known to be robust and is applied in many applications [TvBDK00]. It is simply given by

$$\mathcal{F}(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^N(\mathbf{x}_N)) := \frac{1}{N} \sum_{n=1}^{N} \mathcal{C}^n(\mathbf{x}_n). \tag{3.21}$$

3.4.2 Probabilistic Fusion

A probabilistic approach to combine N classifiers is to apply the Bayes' rule under the assumption that the classifiers are mutually independent. For a two class problem this approach is proposed by THRUN et al. [TB96].

In order to combine classifier outputs with such a probabilistic approach the outputs of the classifiers are interpreted as estimates of the posterior probabilities $P(\omega = l|\mathbf{x}_n)$, see Eq. 3.6. For this, the combined probability $P(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N)$ of the hypothesis $\omega = l$, given the features $\mathbf{x}_1, ..., \mathbf{x}_N$ has to be calculated. Then the ensemble output for class l

$$\mathcal{F}(\mathbf{x}_1, ..., \mathbf{x}_N)_l := P(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N) + \epsilon_l(\mathbf{x}_1, ..., \mathbf{x}_N) \tag{3.22}$$

is additionally decomposed into the combined belief in the hypothesis $\omega = l$, given the features $\mathbf{x}_1, ..., \mathbf{x}_N$ and the error made by the classifier ensemble $\epsilon_l(\mathbf{x}_1, ..., \mathbf{x}_N)$.

In the following, we derive a probabilistic fusion function for a multi class problem. Let $O(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N)$ be the *posterior odds* as defined by

$$O(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N) := \frac{P(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N)}{P(\omega \neq l|\mathbf{x}_1, ..., \mathbf{x}_N)}, \tag{3.23}$$

where $\omega \neq l$ is the negation of the hypothesis $\omega = l$. The combined belief in the hypothesis $\omega = l$ is then given through [Pea88]

$$\begin{aligned} P(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N) : &= \frac{O(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N)}{1 + O(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N)} \\ &= 1 - \frac{1}{1 + O(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N)}. \end{aligned} \tag{3.24}$$

Furthermore, let $O(\omega = l)$ be the *prior odds* and $\hat{L}(\mathbf{x}_n|\omega = l)$ be the *likelihood ratio* as defined by

$$O(\omega = l) := \frac{P(\omega = l)}{P(\omega \neq l)}, \qquad \hat{L}(\mathbf{x}_n|\omega = l) := \frac{P(\mathbf{x}_n|\omega = l)}{P(\mathbf{x}_n|\omega \neq l)}. \tag{3.25}$$

Thereby $P(\omega = l)$ denotes the a priori class probability for the class $\omega = l$ and $P(\mathbf{x}_n|\omega = l)$ denotes the class-conditional probability density function. By assuming, that the class-conditional probability density functions for $n \neq n'$ are independent of $P(\mathbf{x}_{n'}|\omega = l)$ [The89], we can write

$$O(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N) := \alpha O(\omega = l) \prod_{n=1}^{N} \hat{L}(\mathbf{x}_n|\omega = l) \tag{3.26}$$

where α is a normalising constant, requiring that Eq. 3.28 sum to unity over l [Sch92]. In the case of $L = 2$, this normalising constant can be omitted.

Integrating Eq. 3.26 into Eq. 3.24 leads to

$$P(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N) = 1 - \frac{1}{1 + \alpha O(\omega = l) \prod_{n=1}^{N} \hat{L}(\mathbf{x}_n|\omega = l)}. \tag{3.27}$$

After applying the Bayes' rule which says: $P(\mathbf{x}_n|\omega = l) = \frac{P(\omega=l|\mathbf{x}_n)P(\mathbf{x}_n)}{P(\omega=l)}$, the *probabilistic function* [TB96] is given by

$$P(\omega = l|\mathbf{x}_1, ..., \mathbf{x}_N) = 1 - \left(1 + \alpha O(\omega = l) \prod_{n=1}^{N} \frac{P(\omega = l|\mathbf{x}_n)}{P(\omega \neq l|\mathbf{x}_n)} \frac{P(\omega \neq l)}{P(\omega = l)}\right)^{-1}. \tag{3.28}$$

In the case of equal a priori class probabilities $P(\omega = l) = \frac{1}{L}$, this formula reduces to the product rule [TDvB97].

Although, this probabilistic approach is based on the assumption of independent class-conditional probability density functions, this approach seems to provide reasonable results [DSP01a]. It remains to mention, that this assumption is debatable. However, it is often made in statistics, because the joint probability density functions $P(\mathbf{x}_1, ..., \mathbf{x}_N|\omega = l)$ are difficult to refer [KHDM98].

This assumption is for example commonly made in approaches building occupancy grids for robot navigation [TB96]. A detailed discussion concerning probabilistic classifier fusion can be found in [End01].

3.4.3 Percentiles

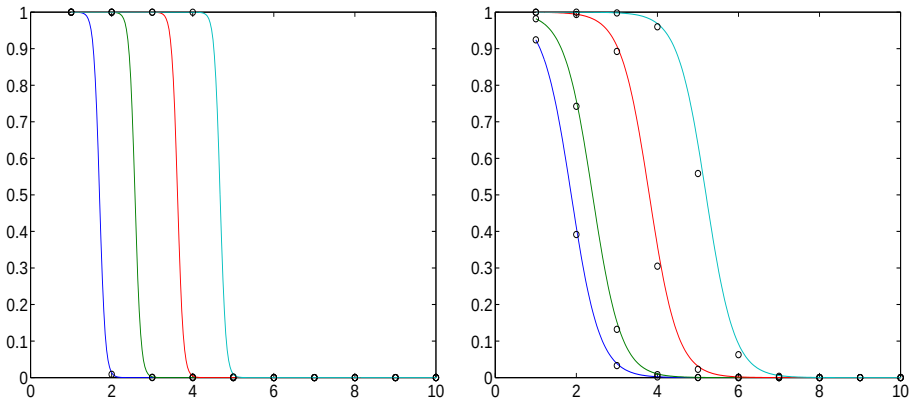
In *percentiles* for an input $\mathbf{x}_1, ..., \mathbf{x}_N$ the classifier outputs $\mathcal{C}_l^n(\mathbf{x}_n)$ for each class $l \in \Omega$ are sorted in decreasing order, e.g. for each class there is a sequence $(\tau_l^n)_{n=1}^N$ of indices, such that $\mathcal{C}_l^{\tau_l^1}(\mathbf{x}_{\tau_l^1}) \geq ... \geq \mathcal{C}_l^{\tau_l^N}(\mathbf{x}_{\tau_l^N})$. Then for $\beta \in [0, 1]$ the fusion mapping is defined by

$$\mathcal{F}_\beta(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^N(\mathbf{x}_N))_l := \mathcal{C}_l^{\tau_l^{\lceil \beta N \rceil}}(\mathbf{x}_{\tau_l^{\lceil \beta N \rceil}}) \tag{3.29}$$

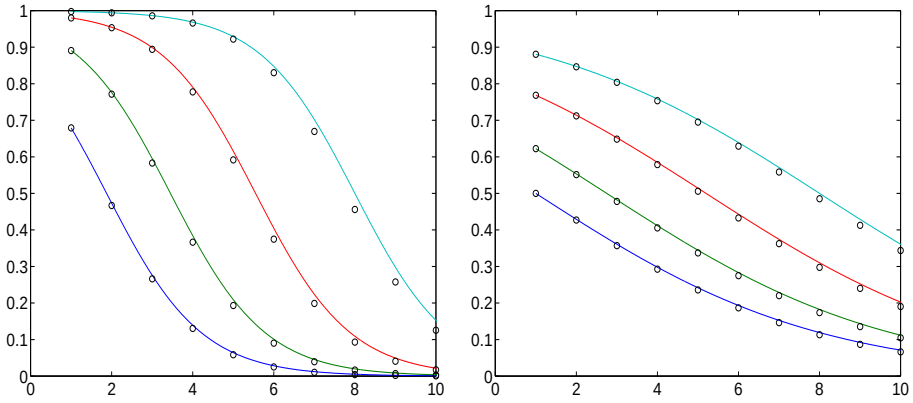
For $\beta = 0$, $\beta = 1$, and $\beta = \frac{1}{2}$ the fusion mapping $\mathcal{F}_\beta(\cdot)_l$ is the minimum, maximum and median of class l [DT00, Kun98].

3.4.4 Voting

Let $\vec{\mathcal{C}}^1, ..., \vec{\mathcal{C}}^N$ be an ensemble of ranked classifiers and $\vec{\mathcal{C}}^n(\mathbf{x}_n) = (\omega_1^n, ..., \omega_R^n)$ be the output of $\vec{\mathcal{C}}^n$ given the feature vector $\mathbf{x}_n$ (see Eq. 3.2). With R_l^n



(a) $\beta = 500$ (b) $\beta = 100$



(c) $\beta = 30$ (d) $\beta = 10$

Figure 3.5: Pay of fusion mappings for the combination of multiple classifiers with voting dependent on the parameters β and θ. In each Figure (a), ..., (d) 4 different parameter settings for θ are shown, whilst β is fixed.

we denote the rank of class l in this sequence, e.g. $R^n_{\omega_l^n} = \omega^n_{R_l^n} = l$ and

$l \in \{\omega_1^n, ..., \omega_R^n\}$. Otherwise we set $R_l^n = \infty$. Typically, in voting a monotonically decreasing pay off mapping is given which rates the ranks of the individual classes $R_1^n, ..., R_L^n$. In [SDKP03] for example a pay of fusion mapping $\mathcal{F}_{\beta,\theta}$ with $\beta > 0$ and $\theta \in [0, 1)$ which is based on the sigmoid function is applied for the temporal combination of classifier decisions. It is given through

$$\mathcal{F}_{\beta,\theta}(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^N(\mathbf{x}_N))_l := \frac{1}{N}\sum_{n=1}^{N}\Big(1 - \frac{1}{1 + \exp(-\beta(\frac{R_l^n - 1}{L-1} - \theta))}\Big). \tag{3.30}$$

In Figure 3.5 the applied pay of functions are shown. Here β determines the slope and θ the location of the inflection point of the signum function. More details about voting and classification results for bioacoustic time series by using the proposed pay off mappings (see Figure 3.5) can be found in [SDKP03]. In these studies radial basis function networks served as first level classifiers.

3.5 Decision Templates for Time Series Classification

In this Section we introduce the concept of multiple decision templates per class for time series classification. Provided that the outputs of the base classifiers show different characteristic patterns, which is a typical behaviour in time series classification, this approach may enhance the classifier performance, because a fusion mapping based on several templates per class leads to an increased expression power of the fusion layer. In our application (see Chapter 4) and our artificial data set (see Chapter 5) this appears for inputs of many classes, which strengthened our presumption to derive algorithms with multiple decision templates per class.

3.5.1 Decision Templates

The concept of decision templates is a simple, intuitive, and robust aggregation idea that evolved from the *fuzzy template* which is introduced by KUNCHEVA, see [KKZ95, Kun01, KBD01]. Let $\Omega = \{1, ..., L\}$ be the set of class labels, and $\mathcal{C} : \mathbb{R}^D \to \Delta$ be a probabilistic classifier (see Eq. 3.4), mapping feature vectors $\mathbf{x} \in \mathbb{R}^D$ into the decision space (see Eq. 3.5). We assume that the classifier mapping $\mathcal{C}$ is trained through a supervised training procedure on a finite training set $\mathcal{D} \subset \mathbb{R}^D \times \Omega$ with patterns $(\mathbf{x}^\mu, \omega^\mu)$ of

feature vectors $\mathbf{x}^\mu$ and corresponding class labels ω^μ. In the classification phase usually a feature vector $\mathbf{x}^\mu$ is presented to the classifier, and the decision is made through the maximum membership detection (see Eq. 2.5).

But decision templates are a trainable fusion scheme, which means, that the previously trained classifier is applied to calculate a second fusion layer (see Figure 3.1). For a trained classifier $\mathcal{C}$, *decision templates* $\mathcal{T}^\omega$ for each class $\omega \in \Omega$ can be calculated by the mean of the classifier outputs $\mathcal{C}(\mathbf{x}^\mu)$ for inputs $\mathbf{x}^\mu$ of class ω [KBD01]:

$$\mathcal{T}^\omega := \frac{1}{|\mathcal{D}_\omega^{\mathrm{V}}|} \sum_{\mathbf{x}^\mu \in \mathcal{D}_\omega^{\mathrm{V}}} \mathcal{C}(\mathbf{x}^\mu) \tag{3.31}$$

Here $\mathcal{D}_\omega^{\mathrm{V}}$ is a finite validation set of $\mathbb{R}^D \times \{\omega\}$ (see Figure 3.2). Then the decision template $\mathcal{T}^\omega \in \Delta$ may be interpreted as characteristic classifier output for the inputs $\mathbf{x}^\mu$ of $\mathcal{D}_\omega^{\mathrm{V}}$. In the context of decision templates the classifier output $\mathcal{C}(\mathbf{x})$ is also called a *decision profile* of classifier $\mathcal{C}$ and input $\mathbf{x}$ [Kun00b, Kun01].

In order to improve the overall performance of the classifier ensemble, particularly for a multi-class pattern recognition problem ($L > 2$), instead of classifying objects based on a single feature, a set of I different features is used. A typical approach to deal with I features is to build I classifiers, i.e. one classifier per feature space (see Figure 3.1), and to combine the classifier outputs into a final decision[DPS03, KBD01]. In the case of I input features with classifier mappings $\mathcal{C}^i : \mathbb{R}^{D_i} \to \Delta$, $i = 1, ..., I$, the *decision template* $\mathcal{T}^\omega$ of class ω is given by a $(I \times L)$-matrix

$$\mathcal{T}^\omega := \begin{bmatrix} \mathcal{T}_1^\omega \\ \vdots \\ \mathcal{T}_I^\omega \end{bmatrix} \in \Delta^I. \tag{3.32}$$

Hereby $\mathcal{T}_i^\omega \in \Delta$ is the decision template of the i-th feature space $\mathbb{R}^{D_i}$ and target class ω. The decision profile $\mathcal{P}$ for an input $X = (\mathbf{x}_1, ..., \mathbf{x}_I)$ is given by the individual classifier outputs of the previously trained base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ (see Eq. 3.8).

Classification of static objects with decision templates works as follows: The base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ are applied to calculate the decision profile $\mathcal{P}$, see step (a) in algorithm DT given in Figure 3.6. Then for each class $\omega \in \Omega$ a class membership value $\mathbf{z}_\omega$ with respect to a *similarity measure* $\mathcal{S}$ between the decision profile $\mathcal{P}$ and the decision template $\mathcal{T}^\omega$ is calculated (see step (b) in algorithm DT, Eq. 3.33 and Eq. 3.34). The class

Algorithm $\omega^* = \mathsf{DT}(X, (\mathcal{T}^\omega)_{\omega=1}^L)$

(a) $\mathcal{P} = [\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^I(\mathbf{x}_I)]^T$

foreach $\omega \in \Omega$

(b) $\mathbf{z}_\omega = \mathcal{S}(\mathcal{P}, \mathcal{T}^\omega)$

end

(c) $\omega^* = \underset{\omega}{\operatorname{argmax}}(\mathbf{z}_\omega)$

—

Figure 3.6: Algorithm DT: Classification of static objects $X = (\mathbf{x}_1, ..., \mathbf{x}_I)$ consisting of I feature vectors with decision templates.

with the maximum membership ω^* is the final decision, see step (c). A numerical illustration of the DT fusion scheme can be found in [KJ00].

In [Kun01, KW01a] KUNCHEVA discussed different similarity measures in the context of decision template classifiers. The most popular similarity measure is based on the normalised squared Euclidean distance (L_2-norm) or more general, it is based on the normalised L_p-norm $||\cdot||_p$ and is defined by

$$\mathcal{S}_p(\mathcal{P}, \mathcal{T}^\omega) := 1 - \frac{1}{2I} \sum_{i=1}^{I} ||\mathcal{P}_{i,\cdot} - \mathcal{T}^\omega_{i,\cdot}||_p \quad \in [0, 1]. \tag{3.33}$$

Another similarity measure also used in applications and theoretical considerations (see Section 3.6) is the normalised correlation

$$\mathcal{S}(\mathcal{P}, \mathcal{T}^\omega) := \frac{1}{I} \sum_{i=1}^{I} \langle \mathcal{P}_{i,\cdot}, \mathcal{T}^\omega_{i,\cdot} \rangle \quad \in [0, 1], \tag{3.34}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product. A special case which allows to compare the decision template combination with static combining paradigms is the *most desirable decision template* [Kun00b, KBD01]. For each class $\omega \in \Omega$ it is given through the $I \times L$ matrix

$$\bar{\mathcal{T}}^\omega_{i,l} := \begin{cases} 1, & l = \omega \\ 0, & \text{otherwise} \end{cases}, \quad i = 1, ..., I, \quad l = 1, ..., L. \tag{3.35}$$

By assuming the most desirable decision template and the normalised

correlation as similarity measure, Eq. 3.34 can be written by

$$\mathcal{S}(\mathcal{P}, \bar{\mathcal{T}}^{\omega}) = \frac{1}{I} \sum_{i=1}^{I} \sum_{l=1}^{L} \mathcal{P}_{i,l} \bar{\mathcal{T}}^{\omega}_{i,l} = \frac{1}{I} \sum_{i=1}^{I} \mathcal{P}_{i,\omega}. \tag{3.36}$$

Since the coefficients of $\mathcal{P}$ are given by the outputs of I classifiers (see Eq. 3.8), the similarity $\mathcal{S}(\mathcal{P}, \bar{\mathcal{T}}^{\omega})$ is equivalent to averaging the classifier decisions of class ω and therefore the combination with the most desirable decision template is equivalent to the static combination through average fusion (see Eq. 3.21). In the numerical experiments this fact was considered to appraise the performance gain of training the decision fusion mapping with decision templates with the static combination with averaging.

3.5.2 Multiple Decision Templates

In this Section the concept of multiple decision templates per class is introduced in the context of time series classification. For the training of the overall classifier system it is assumed that a set of time series $s^{\mu}(\cdot)$, $\mu = 1, ..., M$ is given, each time series is labelled with its corresponding class label $\omega^{\mu} \in \Omega$.

The training procedure of the overall classifier architecture is splitted into the following 4 steps [DSP02]:

1. **Feature extraction**.
   A local sliding window $W^{\jmath}$ covering a small part of the time series (see Figure 3.3) is moved over each time series $s^{\mu}(\cdot)$ in order to extract I feature streams $X^{\mu}(\jmath) = (\mathbf{x}_1^{\mu}(\jmath), ..., \mathbf{x}_I^{\mu}(\jmath))$, $\jmath = 1, ..., \mathcal{J}^{\mu}$ and $\mathbf{x}_i^{\mu}(\jmath) \in \mathbb{R}^{D_i}$. The extracted feature streams are divided into a training set $\mathcal{D}$, and a validation set $\mathcal{D}^{\mathrm{V}}$.

2. **Base classifier training**.
   A set of I base classifier mappings $\mathcal{C}^1, ..., \mathcal{C}^I$ is calculated, where each classifier $\mathcal{C}^i$ is trained with the labelled data of the i-th feature stream, i.e. with the data set $\mathcal{D}_{\cdot,i} := \{((\mathbf{x}_i^{\mu}(\jmath))_{\jmath=1}^{\mathcal{J}^{\mu}}, \omega^{\mu}) | \mu = 1, ..., M^{\mathcal{D}}\}$. Hereby $M^{\mathcal{D}} < M$ is the number of feature streams in $\mathcal{D}$.

3. **Decision template training**.
   The validation set $\mathcal{D}^{\mathrm{V}}$ is used to calculate the base classifier outputs, i.e. for each $X^{\mu}(\jmath) = (\mathbf{x}_1^{\mu}(\jmath), ..., \mathbf{x}_I^{\mu}(\jmath)) \in \mathcal{D}^{\mathrm{V}}$ the base classifier outputs $[\mathcal{C}^1(\mathbf{x}_1^{\mu}(\jmath)), ..., \mathcal{C}^I(\mathbf{x}_I^{\mu}(\jmath))]^T \in \Delta^I$ are computed. From these classifier outputs the decision templates are calculated by the DT, the TDT or the CDT procedure.

4. **Retraining of the base classifiers**.
   The base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ are retrained with $\mathcal{D} \cup \mathcal{D}^{\mathrm{V}}$. For artificial neural networks the retraining of $\mathcal{C}^i$ may be a further weight adaption with $\mathcal{D}_{\cdot,i} \cup \mathcal{D}^{\mathrm{V}}_{\cdot,i}$.

This training procedure leads to I trained classifiers and a set of decision templates. In the following we describe three types of decision template training procedures:

TDT: Temporal decision templates
For each time series in $\mathcal{D}^{\mathrm{V}}_{\omega}$ one decision template is computed through the mean of the decision profiles averaged over all $\mathcal{J}^{\mu}$ time windows. For the μ-th time series (X^{μ}, ω) the decision template is:

$$\mathcal{T}^{\omega,\mu} := \frac{1}{\mathcal{J}^{\mu}} \sum_{\jmath=1}^{\mathcal{J}^{\mu}} \begin{bmatrix} \mathcal{C}^1(\mathbf{x}_1^{\mu}(\jmath)) \\ \vdots \\ \mathcal{C}^I(\mathbf{x}_I^{\mu}(\jmath)) \end{bmatrix} \tag{3.37}$$

This gives for each time series in $\mathcal{D}^{\mathrm{V}}_{\omega}$ a decision template for the associated class $\omega \in \Omega$. Thus $|\mathcal{D}^{\mathrm{V}}_{\omega}|$ defines the number of decision templates for class ω.

DT: One decision template per class
Here the average over all decision templates $\mathcal{T}^{\omega,\mu}$ of class ω is calculated [KBD01]

$$\mathcal{T}^{\omega} := \frac{1}{|\mathcal{D}^{\mathrm{V}}_{\omega}|} \sum_{\mu=1}^{|\mathcal{D}^{\mathrm{V}}_{\omega}|} \mathcal{T}^{\omega,\mu}. \tag{3.38}$$

For constant length of time series $\mathcal{J}^{\mu} = \mathcal{J}$ this is equivalent to the decision templates as defined in Eq. 3.31.

CDT: Clustered decision templates
Here $\hat{k}$ decision templates $\mathcal{T}^{\omega,k}$, $k = 1, ..., \hat{k}$ are calculated for each class ω in Ω by clustering the classifier outputs $(\mathcal{C}^1(\mathbf{x}_1^{\mu}(\jmath)), ..., \mathcal{C}^I(\mathbf{x}_I^{\mu}(\jmath)))$ for each $X^{\mu}(\jmath) \in \mathcal{D}^{\mathrm{V}}_{\omega}$ through a clustering procedure (e.g. $\hat{k}$-means, fuzzy-$\hat{k}$-means, Kohonen's self organised feature map [Bis95]).

The decision templates of class ω certainly depend on the validation set $\mathcal{D}^{\mathrm{V}}_{\omega}$. So multiple decision templates may be computed by re-sampling methods [TS95]. Additionally, it should be emphasised that the templates are also depending on the training set $\mathcal{D}$ of the individual classifiers. This

implies, that for R training sets and R corresponding validation sets there is a set of K decision templates for each class $\omega \in \Omega$ (for TDTs, $K = R|\mathcal{D}_\omega^{\mathrm{V}}|$ and for DTs, $K = R$).

In the CDT approach we have to consider two different ways of clustering:

1. The classifier outputs of all R validation sets are clustered by a single clustering procedure, which leads to $K = \hat{k}$ decision templates per class.

2. The clustering procedure is applied to the classifier outputs for each of the R validation sets separately, which leads to $K = R\hat{k}$ decision templates per class.

In our numerical evaluation the first approach is applied, utilising the $\hat{k}$-means clustering algorithm. Hereby R training- and validation sets are determined by re-sampling.

The classification of time series by the multiple decision template approach is accomplished by the CFTBYMDT algorithm that is a special case of the CFT fusion architecture. It is given in Figure 3.7. The overall classifier architecture consisting of I base classifiers, K decision templates per class, and temporal decision fusion is depicted in Figure 3.8. Provided the I base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ are trained, and a set of $\omega = 1, ..., L$ decision templates $\mathcal{T}^\omega = \{\mathcal{T}^{\omega,1}, ..., \mathcal{T}^{\omega,K}\}$ is given, a time series $s(\cdot)$ can be classified.

As in the feature extraction procedure in the first step I feature streams $X(\jmath) = (\mathbf{x}_1(\jmath), ..., \mathbf{x}_I(\jmath))$, $\jmath = 1, ..., \mathcal{J}$ are extracted from the time series. Then, for feature vector $X(\jmath) = (\mathbf{x}_1(\jmath), ..., \mathbf{x}_I(\jmath))$ the classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ are applied to calculate the decision profile $\mathcal{P}^\jmath$ at time $\jmath$, see step (a). In step (b) the similarity values $\mathcal{S}(\mathcal{P}^\jmath, \mathcal{T}^{\omega,k})$ between the decision profile $\mathcal{P}^\jmath$ and all decision templates $\mathcal{T}^{\omega,k}$, $\omega = 1, ..., L$ and $k = 1, ..., K$ are calculated. For each class ω the $S \leq K$ best matches $(\mathcal{T}^{\omega,\tau_1}, \ldots, \mathcal{T}^{\omega,\tau_S})$ of the K decision templates are computed, e.g. the sequence $(\tau_k)_{k=1}^K$ is determined such that $\mathcal{S}(\mathcal{P}^\jmath, \mathcal{T}^{\omega,\tau_1}) \geq ... \geq \mathcal{S}(\mathcal{P}^\jmath, \mathcal{T}^{\omega,\tau_S}) \geq ... \geq \mathcal{S}(\mathcal{P}^\jmath, \mathcal{T}^{\omega,\tau_K})$. This sequence is given through the set valued mapping $\mathcal{N}_S(\mathcal{P}^\jmath, \mathcal{T}^\omega)$, which defines the S decision templates of class ω with descending similarity to $\mathcal{P}^\jmath$.

Then in step (c) the centroid of the best matches

$$\hat{\mathcal{T}}^\omega = \frac{1}{S} \sum_{k=1}^{S} \mathcal{T}^{\omega,\tau_k} \tag{3.39}$$

is determined for each class $\omega \in \Omega$, see step (c). We call $\hat{\mathcal{T}}^\omega$ *virtual decision template*. Subsequently, for each class ω the local class membership $\mathbf{z}_\omega^\jmath$ is set

Algorithm $\omega^* = \text{CFTBYMDT}((X(\jmath))_{\jmath=1}^{\mathcal{J}}, (\mathcal{T}^\omega)_{\omega=1}^{L})$

for $\jmath = 1, ..., \mathcal{J}$

(a) $\mathcal{P}^\jmath = [\mathcal{C}^1(\mathbf{x}_1(\jmath)), ..., \mathcal{C}^I(\mathbf{x}_I(\jmath))]^T$

foreach $\omega \in \Omega$

(b) $\mathcal{N}_S(\mathcal{P}^\jmath, \mathcal{T}^\omega) = (\tau_1, ..., \tau_S)$

(c) $\hat{\mathcal{T}}^\omega = \frac{1}{S} \sum_{k=1}^{S} \mathcal{T}^{\omega,\tau_k}$

(d) $\mathbf{z}_\omega^\jmath = \mathcal{S}(\mathcal{P}^\jmath, \hat{\mathcal{T}}^\omega)$

end

$\mathbf{z}^\jmath = \mathbf{z}^\jmath / \sum_{\omega=1}^{L} \mathbf{z}_\omega^\jmath$

end

(e) $\mathbf{z}^o = \mathcal{F}(\mathbf{z}^1, ..., \mathbf{z}^{\mathcal{J}})$

$\omega^* = \underset{\omega \in \Omega}{\text{argmax}}(\mathbf{z}_\omega^o)$

—

Figure 3.7: Algorithm CFTBYMDT: Classification of time series with multiple decision templates and temporal decision fusion.

to $\mathbf{z}_\omega^\jmath = \mathcal{S}(\mathcal{P}^\jmath, \hat{\mathcal{T}}^\omega)$, see step (d) and finally the normalised local class memberships $\mathbf{z}^\jmath \in \Delta$ are determined. For the $\mathcal{J}$ decision vectors $\mathbf{z}^1, ..., \mathbf{z}^{\mathcal{J}} \in \Delta$ the temporal combination (see step (e)) is then made through the average of the normalised local class memberships $\mathcal{F}(\mathbf{z}^1, ..., \mathbf{z}^{\mathcal{J}}) := \frac{1}{\mathcal{J}} \sum_{\jmath=1}^{\mathcal{J}} \mathbf{z}^\jmath$. The final decision is then calculated through the maximum membership rule which returns the class ω^* [KHDM98].

Another way to calculate the class membership $\mathbf{z}_\omega^\jmath$ for each class $\omega \in \Omega$ is to determine the set of S decision templates in such a way, that the similarity $\mathcal{S}(\mathcal{P}^\jmath, \hat{\mathcal{T}}^\omega)$ is maximised. In the CFTBYMDT algorithm these decision templates are denoted through $\mathcal{N}_S(\mathcal{P}^\jmath, \mathcal{T}^\omega)$, see step (b). In order to find $\mathcal{N}_S(\mathcal{P}^\jmath, \mathcal{T}^\omega)$ for each of the $\binom{K}{S}$ possible subsets of S decision templates the virtual decision templates (see Eq.3.39) and the corresponding similarities $\mathcal{S}(\mathcal{P}^\jmath, \hat{\mathcal{T}}^\omega)$ have to be calculated. For linear similarity measures (see Eq. 3.34) both methods are equivalent.

It remains the consideration of the fact, that under particular circumstances the decision template fusion scheme is equivalent to average fusion in the temporal domain. Due to the fact that according to Eq. 3.36 fusion with decision templates is equivalent to average fusion if the decision templates are most desirable and the normalised correlation is used

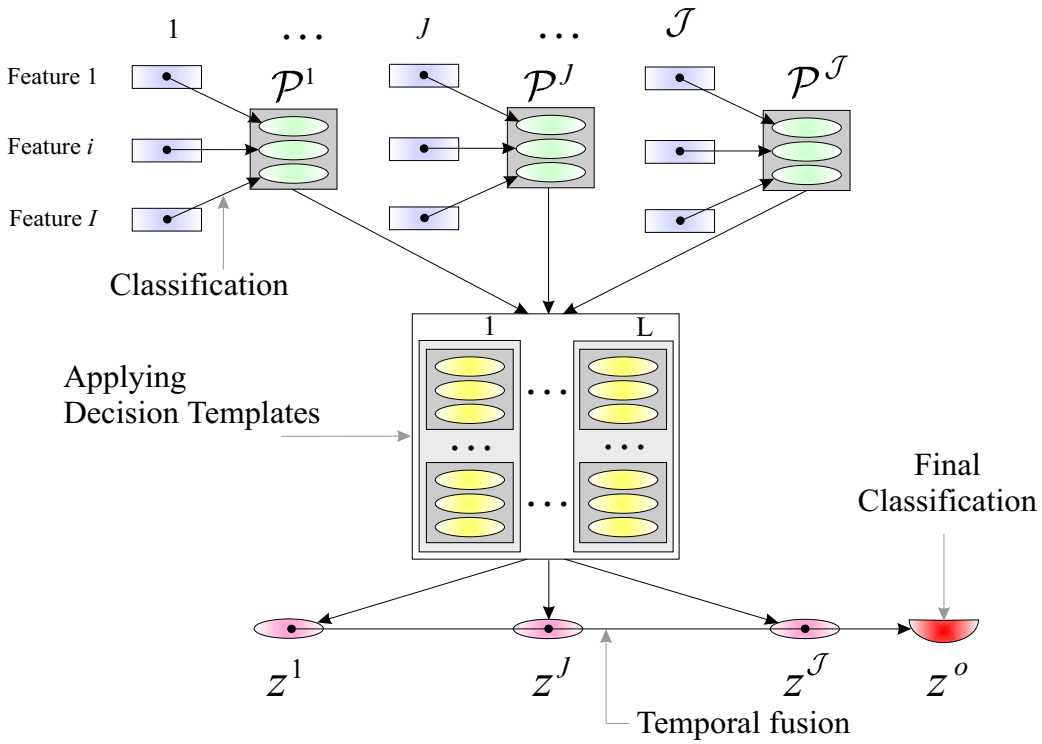


Figure 3.8: Classifying I feature streams $X(\jmath) = (\mathbf{x}_1(\jmath), ..., \mathbf{x}_I(\jmath))$, $\jmath = 1, ..., \mathcal{J}$ with decision templates over time, $\mathbf{z}^1, ..., \mathbf{z}^{\mathcal{J}}$ with $\mathbf{z}^\jmath \in \Delta$ is the sequence of local decisions.

as similarity measure, the local decisions of the CFTbyMDT algorithm and the CDT fusion schem are equivalent if additionally averaging is used for the combination over the feature space (see CDT architecture and Eq. 3.14). By using for both fusion schemes the same aggregation rules for the temporal integration the overall decision is equivalent as well (see Eq. 3.15 and algorithm CFTBYMDT step (e)). Hence, in the numerical experiments the CDT architecture may be considered as static reference architecture for the DT, the TDT and the CDT fusion scheme.

The proposed DT fusion schemes show similarities to prototype based classifiers. In particular, the calculation of the decision templates in the DT approach is equivalent to the calculation of the prototypes for the minimum distance classifier [DH73]. But, also the multiple decision template approaches show similarities to a version of the fuzzy-K-nearest neighbour classifier, which picks the K nearest neighbours from each class (see Section 2.1.3 - Type 2).

3.6 Links to Neural Network Training Schemes

In this Section we consider supervised neural network learning schemes to train the classifier fusion mapping of the two layer architecture (see Figure 3.1). It is assumed, that the base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ have been trained through a supervised learning procedure by a labelled training set $\mathcal{D}$. This corresponds to phase one of the two phase learning scheme discussed in Section 3.1.

Let $\mathcal{C}^i(\mathbf{x}_i^\mu) \in \Delta$ be the classifier output of the i-th classifier $\mathcal{C}^i$ given the input feature vector $\mathbf{x}_i^\mu \in \mathcal{D}^{\mathrm{V}}$ of the i-th feature space. Then for each classifier the outputs of the validation set $\mathcal{D}^{\mathrm{V}}$ are given by a $(L \times M)$-matrix C_i, $i = 1, ..., I$ where $M = |\mathcal{D}^{\mathrm{V}}|$ and the μ-th column of C_i contains the classifier output $\mathcal{C}^i(\mathbf{x}_i^\mu)^{\mathrm{T}}$. Hereby the superscript T denotes the transposition. The desired classifier outputs $\omega^\mu \in \Omega$ of inputs $\mathbf{x}_i^\mu \in \mathcal{D}^{\mathrm{V}}$ are given by the $(L \times M)$-matrix Y defined by the 1 of L encoding scheme for class labels

$$Y_{l,\mu} = \begin{cases} 1, & l = \omega^\mu \\ 0, & \text{otherwise} \end{cases}. \tag{3.40}$$

Corresponding to C_i the μ-th column $Y_{\cdot,\mu} \in \Delta$ contains the binary coded target output of feature vector $\mathbf{x}_i^\mu \in \mathcal{D}^{\mathrm{V}}$.

In the following we discuss different approaches to combine the classifier outputs $\mathcal{C}^i(\mathbf{x}_i)$, $i = 1, ..., I$ into an overall classifier decision

$$\mathbf{z} := \mathcal{F}(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^I(\mathbf{x}_I)). \tag{3.41}$$

Four different learning schemes namely *linear associative memory*, *decision template*, *pseudoinverse matrix* and *naive Bayes* are introduced to calculate the decision fusion mapping $\mathcal{F} : \Delta^I \to \Delta$ (see Eq. 3.41 and Figure 3.1). In all these methods the fusion mapping $\mathcal{F}$ is realised through $(L \times L)$-matrices $V^1, ..., V^I$, calculated through a certain training algorithm. Hereby the training of the matrices V^i is based on the outputs of the first level classifiers C_i and the desired target output Y (see Eq 3.40). The overall classifier system is depicted in Figure 3.9 in more detail.

3.6.1 Linear Associative Memory

A linear decision fusion mapping $\mathcal{F}$ may be realised through an associative matrix memory whose information storage capacity and error-correcting properties have been investigated in several numerical experiments and theoretical investigations [Koh95, Pal80, SP99]. In order to calculate

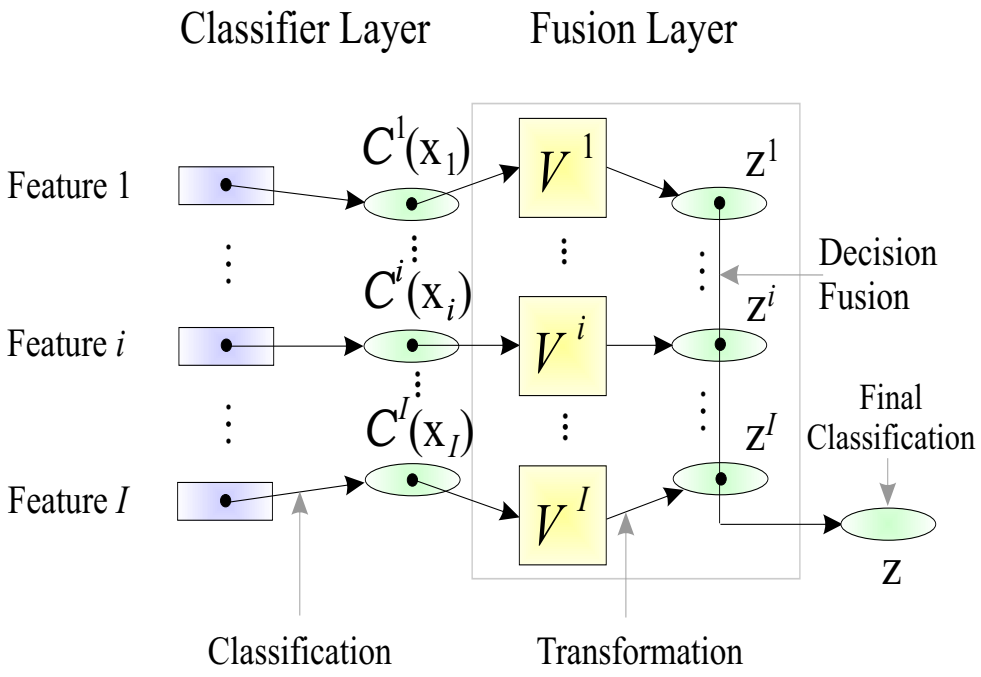


Figure 3.9: Two layer architecture of a MCS consisting of a classifier layer and an additional fusion layer. The combination of the classifier outputs $\mathcal{C}^i(\mathbf{x}_i)$, $i = 1, ..., I$ is accomplished through a fusion mapping $\mathcal{F}(\mathcal{C}^1(\mathbf{x}_1), ..., \mathcal{C}^I(\mathbf{x}_I))$. In this Section we restrict separable linear fusion mappings, where the classifier outputs $\mathcal{C}^i(\mathbf{x}_i)$ are multiplied by matrices V^i, $i = 1, ..., I$. The multiplied decisions $\mathbf{z}^1, ..., \mathbf{z}^I$ are combined with decision fusion.

the *memory matrix* V^i for each classifier $\mathcal{C}^i$ the stored classifier outputs C_i are calculated through a Hebbian learning rule [RMS91]. The associative memory matrix is given as the product of the transposed classifier outputs C_i and the desired classifier outputs Y:

$$V^i = \underbrace{YC_i^{\mathrm{T}}}_{W^i}. \tag{3.42}$$

In the case of crisp classifiers (see Eq. 3.1), the matrix V^i equals the confusion matrix of classifier $\mathcal{C}^i$, where V^i_{ω,ω^*} is equal to the number of samples of the validation set of target class label ω and were assigned by $\mathcal{C}^i$ to class ω^* [XKS92]. For soft classifiers the ω-th row of V^i contains the cumulated soft classifier decisions of $\mathcal{C}^i(\mathbf{x}_i^\mu)$ for the feature vectors $\mathbf{x}_i^\mu \in \mathcal{D}_\omega^{\mathrm{V}}$.[5] After

[5] Independent from the classifier type we denote W^i as confusion matrix. It is defined to compare the individual fusion schemes (see Eq. 3.42,3.47, 3.48, 3.49 and 3.50).

this second training phase the two layer classifier system consisting of a classifier layer and a fusion layer is trained. Whereas the classifier layer is defined by the classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$, the fusion layer is defined by the set of linear matrix operators $V^1, ..., V^I$.

In the classification phase these matrices are then used to combine the individual classifier decisions to calculate the overall classifier decision (see Eq 3.41). For a feature vector $X = (\mathbf{x}_1, ..., \mathbf{x}_I)$, the classifier outputs $\mathcal{C}^i(\mathbf{x}_i)$ are applied to the linear associative memories V^i, $i = 1, ..., I$ and the outputs $\mathbf{z}^i \in \mathbb{R}^L$ are given by

$$\mathbf{z}^i := \mathcal{C}^i(\mathbf{x}_i)V^i. \tag{3.43}$$

The combined class membership estimate is then given by the average of the outputs of the associative memories

$$\mathbf{z} := \sum_{i=1}^{I} \mathbf{z}^i = \sum_{i=1}^{I} \mathcal{C}^i(\mathbf{x}_i)V^i. \tag{3.44}$$

The final decision just depends on the combined class membership estimate $\mathbf{z}$ and is determined by the maximum membership rule

$$\omega := \underset{l \in \Omega}{\operatorname{argmax}}(\sum_{i=1}^{I} \mathbf{z}_l^i). \tag{3.45}$$

3.6.2 Decision Templates

Section 3.5.1 introduces the concept of decision templates which has been proposed by KUNCHEVA [Kun01]. Hereby the decision templates are calculated by the mean of the classifier outputs of a specific class $\omega \in \Omega$ (see Eq. 3.31). In the case of I input features for each class a decision template $\mathcal{T}^\omega$ is given by a $(I \times L)$-matrix (see Eq. 3.32). In order to align the decision template combining scheme in such a way that the combination is applied as proposed in Figure 3.9, for each feature space a linear matrix operator V^i has to be calculated by the decision template algorithm. Let $\mathcal{T}_i^\omega \in \Delta$ be the decision template of the i-th feature space and target class ω as defined in Eq. 3.31. Then for each feature space $i = 1, ..., I$, a $(L \times L)$-*decision template operator* V^i is simply given by the decision templates of the i-th feature space

$$V^i := \begin{bmatrix} \mathcal{T}_i^1 \\ \vdots \\ \mathcal{T}_i^L \end{bmatrix} \in \Delta^L. \tag{3.46}$$

It can be computed similar to the associative memory matrix (see Eq. 3.42) by using the stored classifier outputs and the desired classifier outputs

$$V^i := (YY^{\mathrm{T}})^{-1} \underbrace{(YC_i^{\mathrm{T}})}_{W^i}. \tag{3.47}$$

Multiplying $(YY^{\mathrm{T}})^{-1}$ with the confusion matrix W^i equals the row wise normalisation of W^i by the number of training patterns per class $|\mathcal{D}_\omega^{\mathrm{V}}|$.

By assuming the normalised correlation (see Eq. 3.34) as similarity measure [DSP02, Kun01] the combination of the classifier outputs with decision templates (see algorithm DT in Figure 3.6) is equivalent to the combination of classifier outputs with the linear associative memory. Thus, for a set of input vectors $X = (\mathbf{x}_1, ..., \mathbf{x}_I)$ and a set of classifier outputs $\mathcal{C}^i(\mathbf{x}_i)$, $i = 1, ..., I$ the combined class membership estimate is given by Eq. 3.44.

3.6.3 Pseudo-Inverse Solution

Another linear decision fusion mapping $\mathcal{F}$ can be calculated for each feature space by the optimal least squares solution between the stored classifier outputs C_i and the desired classifier outputs Y. Such a mapping is given by the *pseudoinverse solution* which is another type of the linear associative memory [Hay94]. The linear matrix operator V^i given by the pseudoinverse solution (also called generalised inverse matrix) is given by

$$V^i := \lim_{\alpha \to 0} \underbrace{(YC_i^{\mathrm{T}})}_{W^i} (C_i C_i^{\mathrm{T}} + \alpha I)^{-1}. \tag{3.48}$$

Provided, the inverse matrix of $C_i C_i^{\mathrm{T}}$ exists, which is always the case for full rank matrices C_i, the pseudoinverse solution is given for $\alpha = 0$

$$V^i = (YC_i^{\mathrm{T}})(C_i C_i^{\mathrm{T}})^{-1}. \tag{3.49}$$

Due to the fact, that for crisp classifiers the matrix product $C_i C_i^{\mathrm{T}}$ is always a diagonal matrix, Eq 3.48 can be solved with $\alpha = 0$ if $V^i_{l,l} > 0$ for each $l \in \Omega$. As for the linear associative memory and the decision template, the matrix operators $V^1, ..., V^I$ determined by the pseudo-inverse solution are applied for a linear combination of the classifier outputs. Therefore, for a set of classifier outputs $\mathcal{C}^i(\mathbf{x}_i)$, $i = 1, ..., I$ the combined class membership is given by Eq. 3.44 as well.

3.6.4 Naive Bayes Decision Fusion

In the naive Bayes fusion scheme the classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ are assumed to be crisp and mutually independent [Kun00b], therefore in [XKS92] it is called *Bayes combination*. In the case of I feature spaces for each feature space a $(L \times L)$-matrix operator is calculated based on the stored classifier outputs C_i and the desired classifier outputs Y. These matrix operators are then given through

$$V^i := \underbrace{(YC_i^{\mathrm{T}})}_{W^i}(C_iC_i^{\mathrm{T}})^{-1}. \tag{3.50}$$

In contrast to decision templates (see Eq. 3.47) the confusion matrices W^i, $i = 1, ..., I$ are normalised column wise and V^i is called *label matrix*. Due to the fact that the classifier $\mathcal{C}^i(\mathbf{x}_i)$ is typically error-bearing, the confusions within the classifier decisions can be used to calculate the uncertainty of $\mathcal{C}^i$ to misclassify $\mathbf{x}_i$. Therefore, the matrix coefficient V^i_{l,l^*} is an estimate of the conditional probability

$$\hat{P}(\omega = l|\mathcal{C}^i(\mathbf{x}_i) = l^*), \tag{3.51}$$

that the true class label is l given that $\mathcal{C}^i$ assigns the feature vector $\mathbf{x}_i$ to the crisp class label $l^* \in \Omega$.

After learning, the label matrices $V^1, ..., V^I$ are applied to calculate the final decision of the classifier ensemble. The classification works as follows: Let $\mathcal{C}^1(\mathbf{x}_1) = \omega_1, ..., \mathcal{C}^I(\mathbf{x}_I) = \omega_I$ be the class decisions of the individual classifiers by assigning a set of feature vectors $X = (\mathbf{x}_1, ..., \mathbf{x}_I)$ to them. Then by the independence assumption [Jay96], the estimate of the probability that the true class label is l, is calculated by [XKS92]

$$\mathbf{z}_l := \alpha \prod_{i=1}^{I} \hat{P}(\omega = l|\mathcal{C}^i(\mathbf{x}_i) = \omega_i), \quad l \in \Omega \tag{3.52}$$

where α is a normalising constant that ensures $\sum_{l=1}^{L} \mathbf{z}_l = 1$. This type of Bayesian combination and various methods for uncertainty reasoning have been studied extensively in [Pea88].

3.6.5 Discussion

All four combining schemes are realised by a set of $(L \times L)$-matrices $V^1, ..., V^I$, where the coefficients are modified by a certain training algorithm. All these matrices are calculated by using the classifier outputs $C_1, ..., C_I$ of

the individual first level classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ and the desired classifier ensemble output Y.

By considering the equations for the calculation of these matrices it is conspicuous that the confusion matrix is the main ingredient in all combining schemes, as the confusion matrix $W^i = YC_i^{\mathrm{T}}$ (see Eq. 3.42, 3.47, 3.48 and 3.50) is normalised for each combining scheme in a different way. Hereby the confusion matrix describes the uncertainty of the error-bearing classifier which is considered for the training of the fusion layer. From another point of view, the confusion matrix W^i could be regarded as prior knowledge of the regarding classifier [XKS92].

In three of the four combining schemes the combination of the classifier decisions is a matrix-vector product (see Eq. 3.44). The only exception is the combination with the naive Bayes fusion scheme which is based on probability theory.

By assuming the normalised correlation as similarity measure, fusion with decision templates is very similar to the combination with linear associative memories. Both methods lead to equal class decisions (see Eq. 3.45) if the validation set $\mathcal{D}^{\mathrm{V}}$ contains for each class the same number of feature vectors, e.g. $|\mathcal{D}_1^{\mathrm{V}}| = ... = |\mathcal{D}_L^{\mathrm{V}}|$. Let $\kappa \in \mathbb{N}$ be the number of feature vectors for each class in $\mathcal{D}^{\mathrm{V}}$. Then the normalisation term of the decision template $(YY^T)^{-1}$ (see Eq. 3.46) can be written by

$$(YY^T)^{-1} = \mathtt{diag}(\kappa, ..., \kappa)^{-1} = \mathtt{diag}(\frac{1}{\kappa}, ..., \frac{1}{\kappa}). \tag{3.53}$$

As a consequence, the decision vector $\mathbf{z}$ of the decision template is multiplied by $\frac{1}{\kappa}$ and therefore, the class decisions of both fusion schemes are equal, as they are calculated with the maximum membership rule (see Eq. 3.45).

Fusion with the naive Bayes combination is very similar to the pseudoinverse matrix solution regarding the coefficients of the matrix operators $V_1, ..., V_I$, particularly if crisp classifiers are applied to classify the feature vectors in $\mathcal{D}^{\mathrm{V}}$. In this case the term $C_iC_i^{\mathrm{T}}$ (see Eq. 3.48 and Eq. 3.50) is a diagonal matrix containing the number of feature vectors in $\mathcal{D}^{\mathrm{V}}$ which have been assigned to the individual classes. Let κ^ω be the number of feature vectors in $\mathcal{D}^{\mathrm{V}}$ which have been assigned by $\mathcal{C}^i$ to class ω. Then the normalisation term is given by

$$C_iC_i^{\mathrm{T}} = \mathtt{diag}(\kappa^1, ..., \kappa^L). \tag{3.54}$$

If $\kappa^\omega > 0$ for each $\omega \in \Omega$, the inverse of $C_iC_i^{\mathrm{T}}$ is given through

$$(C_iC_i^{\mathrm{T}})^{-1} = \mathtt{diag}(\kappa^1, ..., \kappa^L)^{-1} = \mathtt{diag}(\frac{1}{\kappa^1}, ..., \frac{1}{\kappa^L}) \tag{3.55}$$

and therefore Eq. 3.48 can be solved for $\alpha = 0$. But, for the naive Bayes combination and the pseudoinverse solution the class decisions of both fusion schemes usually differ, since two different combining schemes are used to combine the outputs of the base classifiers.

3.7 Statistical Evaluation

Because we have to deal with limited data sets the cross-validation method [Bis95] is used to evaluate the proposed classification algorithms. In the $\tilde{k}$-fold cross-validation testing procedure the data set is divided into $\tilde{k}$ disjoint subsets. Then the classifier ensemble (the base classifiers and the fusion layer) is trained $\tilde{k}$ times, each time using a version of the data set omitting exactly one of the $\tilde{k}$ subsets. Such a data set is the training set for the whole classifier ensemble, which is the union of the training set $\mathcal{D}$ which is used to train the base classifiers $\mathcal{C}^1, ..., \mathcal{C}^I$ and the validation sets $\mathcal{D}^{\mathrm{V}}$ which is used to calculate the fusion layer (see Figure 3.1).

The omitted subset $\mathcal{D}^{\mathrm{T}}$ is then used to test the overall classifier system. Finally, the achieved classification results are averaged over all $\tilde{k}$ classifier tests. In the numerical evaluation the cross-validation procedure has been performed using $\tilde{k} - 1$ time series per class to train the classifier ensemble, while one time series was used for testing.

3.8 Summary

The preceding Section introduces with static and adaptive fusion schemes for the combination of multiple classifiers. Learning strategies for the overall multiple classifier system training are discussed and a graduation of MCS by four factors is presented.

In particular, this Section deals with MCS approaches for time series classification by using local classifier decisions and temporal decision fusion. Three fusion architectures: the CDT, the FCT and the CTF architecture have been presented and a new fusion approach called multiple decision template (MDT) has been derived from the decision template (DT) algorithm. In order to calculate multiple decision templates per class, two new adaptive approaches called temporal decision templates (TDTs) and clustered decision templates (CDTs) have been suggested. These new methods lead to an increased expression power of the fusion layer, as they are able to deal with different characteristic patterns within the classifier decisions. Similarities between the proposed DT fusion schemes and the

CDT architecture are discussed in Section 3.5.2. The motivation of the CDT approach is derived from the structure of the bioacoustic data set and is presented in Section 4.7. Furthermore, the motivation of the TDT fusion scheme can be found in Section 5.3.

By using the normalised correlation as similarity measure, learning of decision templates is introduced and discussed in the context of neural network training schemes. In particular, links to well established methods: The pseudoinverse matrix, the linear associative memory and naive Bayes decision fusion are given (see Section 3.6).

Chapter 4

Orthoptera Bioacoustics

This Chapter gives an overview about bioacoustics and automated song analysis. The sound production mechanisms of the *Orthoptera* are reviewed and the corresponding sound structures are inspected. Section 4.2 deals with filtering and segmentation of the acoustic signals and hierarchical extraction of discriminative features in four different time scales. A data set description refers to the equipment (mainly microphones and analog and digital tape recorders) which have been used for the sound recordings. The evaluation of the individual features and the automated selection of features is applied to determine discriminative feature sets for the experiments. Additionally, the automatically selected feature sets are discussed in the context of sound production and morphology of the *Orthoptera* and traditional bioacoustics. Experimental results for the proposed fusion schemes are given in Section 4.6.

4.1 Introduction

Animal bioacoustics is an interdisciplinary field of research that focuses on how different species produce, receive and process sound. Although studies of animal sounds have been conducted throughout history, these studies have only recently been viewed as a coherent field [III98]. Many strategies are used by biologists to study the sounds of animals:

1. **Anatomy**. Anatomical data serve to construct theoretical models of the production and reception of animal sounds. It also can provide important clues to find out how systems might work [RKH90].

2. **Signal analysis**. Recent advances in signal processing allows a detailed analysis of sounds of animals [HHM99, SH97]. Furthermore, signal analysis gives important clues about how the sounds are produced (see Section 4.1.3).

3. **Behavioural analysis**. A variety of experimental and observational approaches are applied to study the acoustic capabilities of animals

[JWK92, SBK+99]. For example, playback studies have been used extensively to investigate how the animals respond to sounds. In some of the more recent playback studies artificial sounds are generated and sound parameters are varied to study the reaction of the animal to different features of the sound [Jat95].

4. **Neuroscience**. Neurobiological methods (e.g. single electrode recordings) were used to investigate the neural mechanisms and coding schemes of the auditory system of animals [MPS+01, Stu99]. For example, a detailed exploration of the auditory frequency tuning by comparing tympanal vibrations with the response of auditory receptor neurons is given in [MMH02].

5. **Computational modelling**. The computational models will likely play an increasingly important role in future studies [III98]. Additionally, they reveal hidden assumptions of the underlying theoretical models and can potentially help to understand the neural information processing [MSP+01].

This Section investigates the acoustic abilities of the insect order *Orthoptera* whose species are a sensitive indicator for habitat quality in tropical ecosystems [Rie98]. Figure 4.1 shows the *systematics* of the insect order of the *Orthoptera*, including the suborders, superfamilies, families and subfamilies.

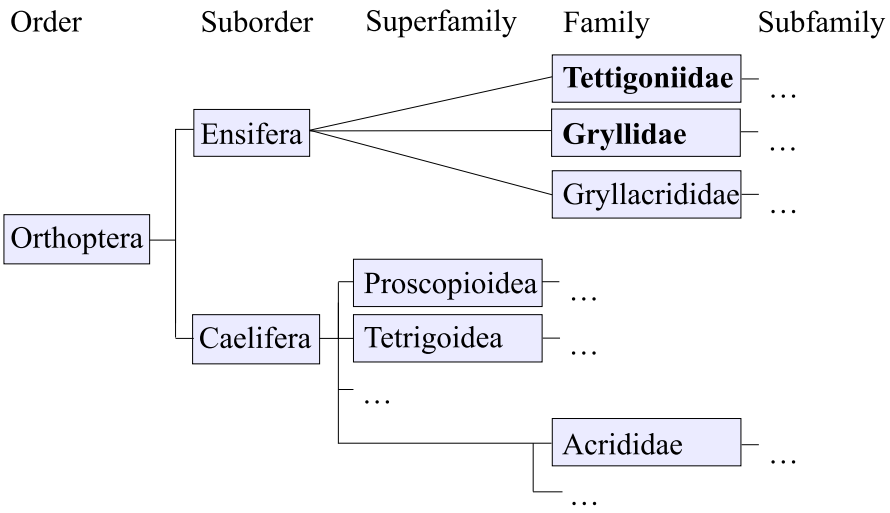


Figure 4.1: The insect order *Orthoptera* including the suborders *Ensifera* and *Califera*.

The name *Orthoptera* (in German: Geradflügler) goes back to OLIVIER (1789) [IK98]. This insect order contains the two suborders *Ensifera* (long-horned grasshoppers: *Gryllidae*, *Tettigoniidae* and the *Gryllacrididae*) and the *Califera* (short-horned grasshoppers, containing a large number of superfamilies). In this Section we mainly focus on the *Gryllidae* (crickets) producing calls with a simple melodic pattern, and the *Tettigoniidae* (katydids) generating calls with a temporal structure that is much more complex [KKK90].

4.1.1 Automated Song Analysis

Bioacoustic song analysis already forms an integral part within species descriptions of "new"*Orthoptera* [Ing97]. The graphic representation of recordings is a prerequisite for the analysis of the temporal structure and frequency composition of animal songs. Historically, it began with the use of the spectrograph and temporal analysis was applied by examination of oscilloscope tracks [Rie98]. The advance of computer technologies allowed *digital analysis*, mainly using the fast Fourier transform (FFT) for the spectral analysis. Most of these early analyses were based on recordings that did not cover the full bandwidth of the production range of the animal [III98]. During the last decade, expensive hardware and software has been replaced by personal computers (PCs) with moderately priced soundcards and cheap, but fully functional software, making this technology financially accessible even for low research budgets. However, these studies relied heavily on aural perception of sound signals and visual inspections of sound spectra, which relies on agreement between human experts [KM98, Rie93].

In this thesis we study automated bioacoustics, where even feature extraction and classification of the individuals is applied automatically by a PC. For this pattern recognition task the knowledge from traditional bioacoustics plays a significant role to find discriminative features and applicative classification methods. The knowledge from traditional bioacoustics is particularly valuable in the context of this thesis, because the data set used in the numerical experiments has been already analysed by the biologists who have recorded the individual sounds.

By catching voucher specimens, NISCHK [Nis99] has been shown that cricket songs recorded from one tropical location in Ecuador could be separated qualitatively within a parameter space of carrier frequency and impulse intervals [Rie98]. This is shown by clusters of manually extracted feature vectors that are produced by distinct species [Nis99]. A deeper

insight into the analysis of *Orthoptera* songs is given by HELLER, who examined the acoustic properties, the stridulatory apparatus and the wing movements of a large number of species of the *Tettigoniidae* and the *Gryllidae* [Hel88]. Furthermore, preliminary work was applied by INGRISCH who analysed the anatomy, the behaviour, the life cycle and the song structure of the *Orthoptera* [IK98]. The data sets of INGRISCH, HELLER and NISCHK (see Section 4.3) have been used for the numerical experiments presented in Section 4.6.

Besides the mentioned features used by traditional, descriptive bioacoustics, other signal parameters may be analysed in order to classify animal vocalisations. Such parameters may be derived from parameters used in speech recognition [RJ86] and speaker identification, e.g. linear predictive coding (LPC) (LPC coefficients [ST95] and LPC-smoothed spectra [VHH98]) and time encoded signals (TES) parameters [KG78] (see Section 4.2.5). In addition, energy contours and frequency contours (see Section 4.2) exposed to discriminate the *Orthoptera* at species level.

4.1.2 Sound Production and Morphology of the *Orthoptera*

Many insects produce sounds, either for defence or sexual communication. Males of crickets, katydids and grasshoppers *(Acrididae)* sing by using specialised structures on different parts of the legs, hind-body *(abdomen)* and fore-wings *(tegmina)* as stridulatory apparatus [Rie98].

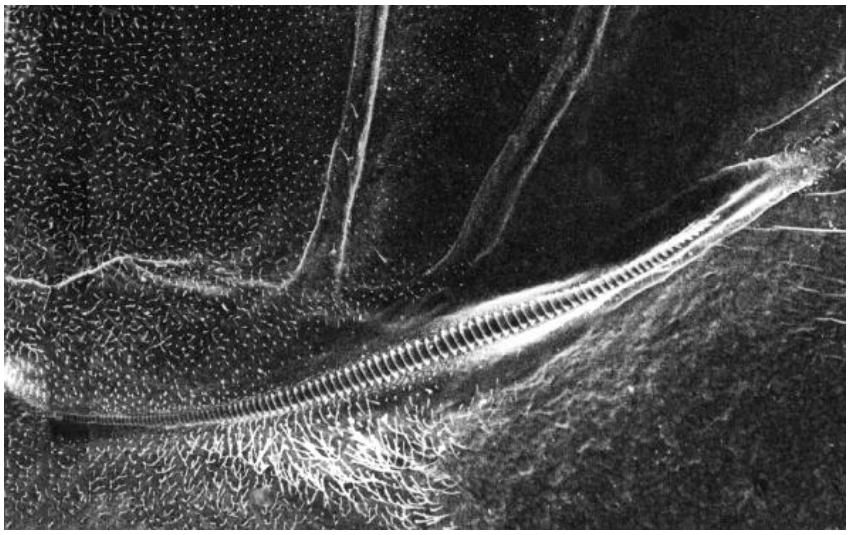


Figure 4.2: Stridulatory file *(pars stridens)* of the species *Zvenella yunnana*.

The most common sound production mechanism of the *Gryllidae* and *Tettigoniidae* is the *elytro-elytral* stridulation. During evolution they have developed two specialised regions: (1) the *stridulatory file* and (2) the *scraper* on their fore-wings which are frictionally used for sound production. The stridulatory file *(pars stridens)* is a Cu2 vein that differs from typical wing veins in that their lower surfaces are supplied with knobs or teeth. Complementary, the scraper *(plectrum)* consists of sclerotised areas which may be a single ridge or hard projection such as a raised wing vein on the anterior medial margin of each tegmen.

In the *Tettigoniidae*, the wings are unequal; the stridulatory file (see Figure 4.2) is located at the ventral side of the left elytron and the *scraper* is located at the fronto-median edge of the right elytron [CFS98]. During calling (stridulation), the pars stridens of the left upper elytron is rhythmically rubbed against the plectrum situated at the right lower elytron. With each capture occurring at opening and closing strokes a sequence of of acoustic transients called *impulses* is generated.

In contrast the wings of the *Gryllidae* are usually symmetrical; the stridulatory file is located at the ventral side of both fore-wings as well the scraper is located at the fronto-median edge of both fore-wings (see Figure 4.3). During calling (stridulation), the pars stridens of the right upper elytron is rhythmically rubbed against the plectrum situated at the inner edge of the left lower elytron. With each capture occurring at the closing stroke the Cu2 vein is set into vibration, whereas the opening phase remains silent [WC75]. Since the Cu2 vein of each wing forms one border of a triangular wing cell, known as the *harp*, vibrations of the Cu2 excite the harp to vibrate and radiate sound [Pre00]. A distinct membrane area, which includes a *mirror cell* [1] acts as a resonator, amplifying the fundamental carrier frequency produced by the interactions between the stridulatory file and the plectrum [DG95]. Thus, each pulse approximates a pure tone whose frequency is determined by the tooth-strike rate, in interaction with the carrier frequency of the sound radiator [BC96].

The vibratory energy transmitted by the calling insect can be divided into two components as perceived by the receptor system (see Section 4.1.4): *airbone sound* and *substrate vibration* [KKK90]. Both components are an essential feature of the communication and localisation process which includes the transition of a temporal image. Whereas airbone sound is transmitted through the air by frequencies often extending into the high sonic

[1] The mirror cell of crickets is extremely thin, which is a physical requirement for maintaining carrier frequencies in the cricket-specific range (*Cycloptiloides canariensis*: thickness 0.2 μm).

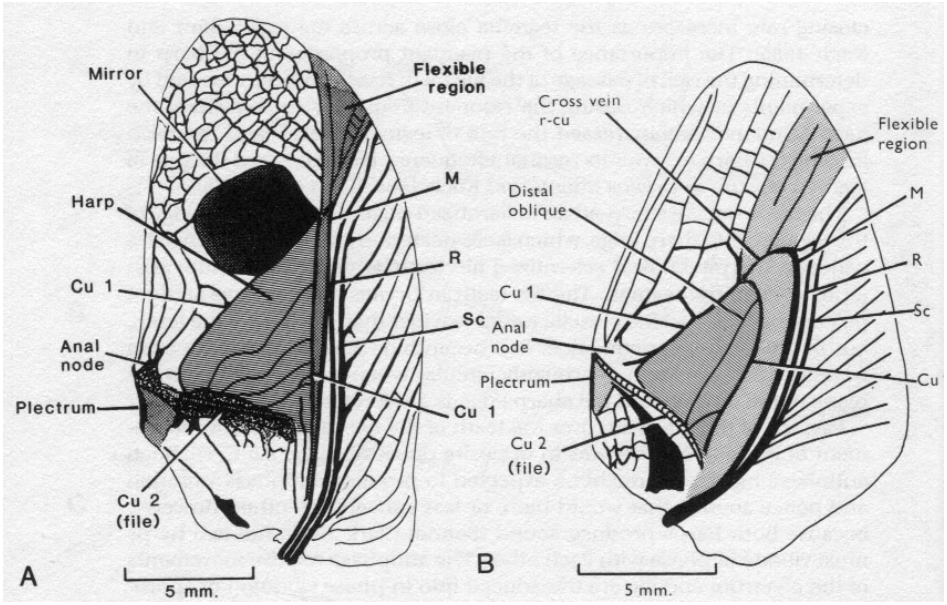


Figure 4.3: Cricket male tegmina, seen from below, showing important structures for the sound production. (A) *Gryllus campestris*, (B) *Gryllotalpa vineae* (adapted from [HML89]).

(10-20 kHz) or even ultrasonic range, substrate vibration (1 Hz to 5000 Hz) contains additional information conveyed over a shorter range by the substrate on which the animal sits. Usually even the temporal pattern between these signals differs and is produced by different mechanisms [SMLH02]. The factors that attenuate and distort airbone-sound signals fall into six groups (1) atmospheric absorption, (2) spherical attenuation (3) accumulation of reverberation from reflections from objects near the path of transmission (4) attenuation by scattering, (5) accumulation of irregular amplitude fluctuations as a result of diffraction from non-stationary turbulence in the atmosphere and (6) diffraction of sound by temperature and other velocity gradients in the environment [NW01]. These factors produce changes in the sound which are often greater in high and ultrasonic frequency ranges because the ground and low vegetation tends to act as a low-pass filter [KKK90].

Most species, particularly crickets use 3 different types of songs [IK98]:

- **Calling songs.** Species-specific songs used to call females from far away.
- **Courtship songs.** Mostly used if male and female are close or preferably if they have antennal contact.

- **Aggressive songs.**[2] This type of song is used by one male telling all other males to keep their distance.

As the number of teeth on the pars stridens used for stridulation differs only slightly between species (60-90 teeth) and the total length of the pars stridens ranges from 1500 μm to 3500 μm, important differences are observed in the course and structure of the pars stridens [Hel88]. This provides species-specific characters particularly in the impulse structure of the *Tettigoniidae* [Jat95]. The relation between wing movement and sound structure is different for crickets and katydids, and thus, these differences are discussed in more detail in Section 4.1.3.

4.1.3 Acoustic Structure of the *Orthoptera* Sound

The nomenclature for the elements of the stridulatory signals differs in publications [Bro63]. Thus, we define the notations of the sound structure of the *Orthoptera* by the following terms (see Figure 4.4 and 4.6):

- **Impulse.** Damped acoustic transient which may have several oscillations before decaying. An impulse is produced by a single tooth-impact of the pars stridens on the plectrum [Hel88].

- **Pulse.** In contrast to an impulse, a pulse is a damped acoustic transient that is produced by one or more tooth-impacts.

- **Syllable.** Acoustic signal produced by one upward and downward movement of the pars stridens over the plectrum. In the *Ensifera*, one upward and downward stroke of the wings is producing an opening and a closing syllable. In the *Acrididae* an opening and closing syllable is produced by one upward and downward movement of the hind-legs [IK98].

- **Chirp.** Syllables occurring in repeated short sequences. The chirps are separated by pauses.

- **Trill.** Multiple consecutive syllables (no inter-syllable pause) which can be produced for some considerable time [CFS98].

- **Verse.** Multiple syllables which are isolated through inter-syllable pauses.

[2] Aggressive songs are also termed *battle calls*.

In the following, we focus on the sound production of the *Gryllidae* and the *Tettigoniidae* and distinguish between these families because the production of the individual impulses differs with respect to the wing movement.

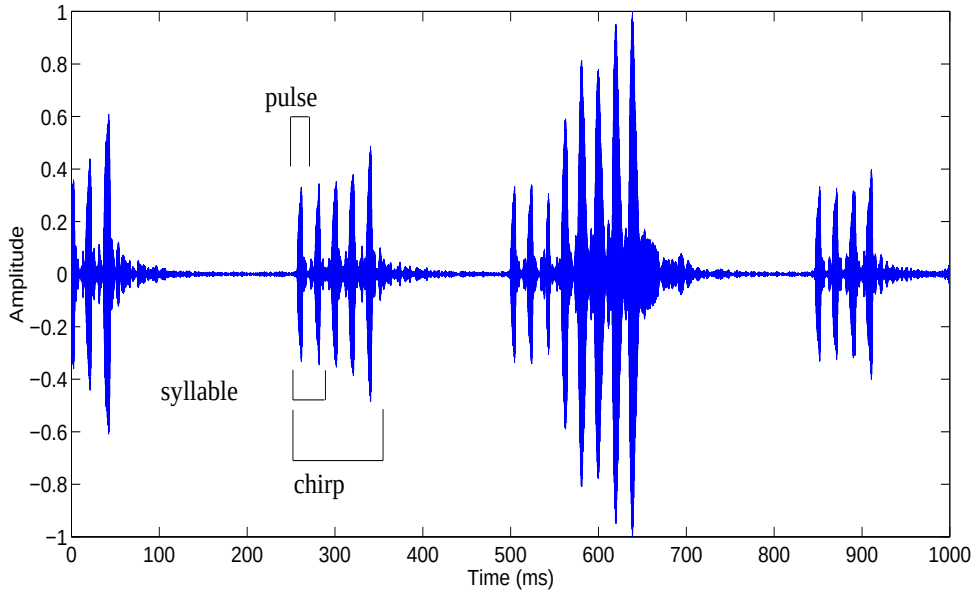
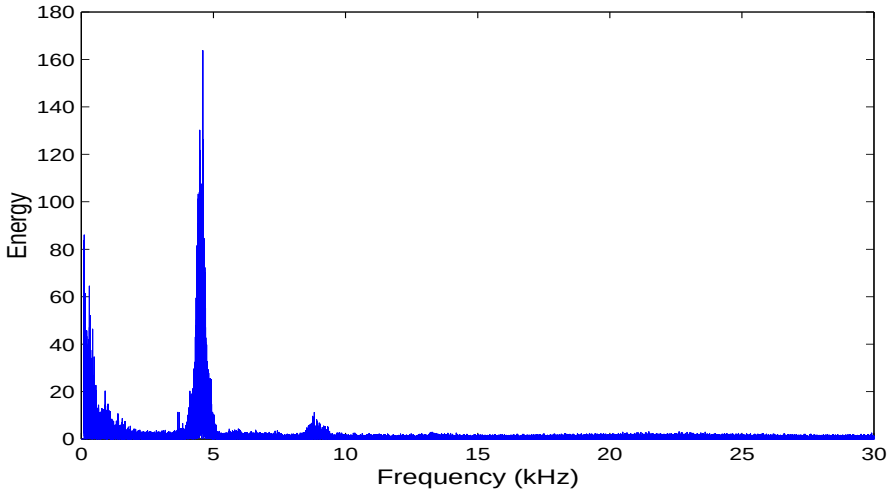


Figure 4.4: Time amplitude pattern of the acoustic signals of the cricket species *Homoeoxipha lycoides*.

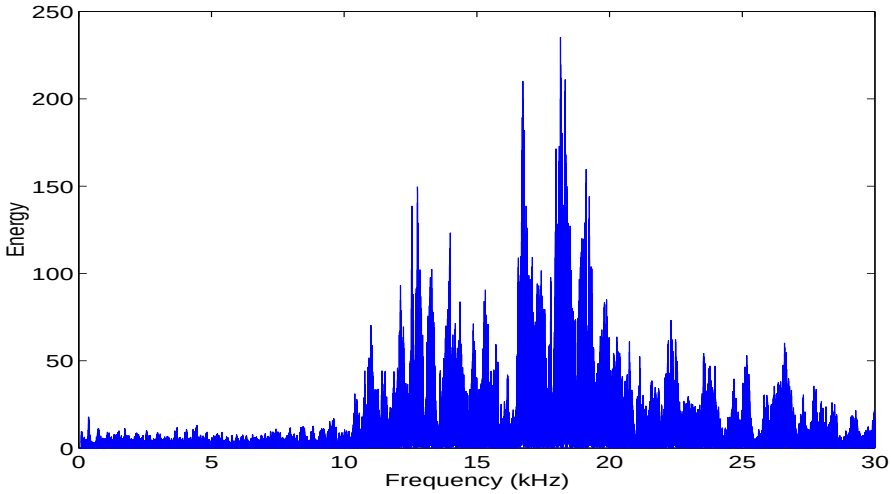
For the *Gryllidae* a single pulse is produced by a downward movement of the wings. Due to the fact that the tooth-impact rate interacts with the carrier frequency of the sound radiator, the frequency range inside pulses is narrow [Nis99]. This type of sound production is called *resonant sound production*. Therefore, the resulting spectrum is narrow (see Figure 4.5(a)).

Gryllidae, in particular, are well known for their production of pure tone impulses between 2 and 11 kHz [IK98, RD94, Nis99], i.e. well within the human auditory range. The temporal structure is species-specific and highly organised on time scales of different orders of magnitude. Besides sonograms, biologists use three features to label species-specific song parameters: frequency (2 - 11 kHz), pulse repetition rate (10 - 150 Hz) [3] and chirp repetition rate (0.01 - 2 Hz) [4], often irregular [Ott92]. Female crickets are attracted by songs of conspecific males. This so-called *phonotaxis*

[3] Impulse distance: 7 - 100 ms
[4] Chirp distance: 0.5 - 10 s



(a) *Homoeoxipha lycoides*



(b) *Ephippiger ephippiger*

Figure 4.5: Typical power spectra of the acoustic signals of the cricket species *Homoeoxipha lycoides* and the katydid species *Ephippiger ephippiger*.

has been studied extensively for several species of *Gryllus* spp. [HML89]. SCHILDBERGER [Sch94] has shown for these species, that the pulse repetition rate is one of the principal features for song recognition, and found neural correlates of female phonotactic behaviour.

In contrast to species of the *Gryllidae*, most of the *Tettigoniidae* species produce their sounds by the *non-resonant sound production* mechanism. Here the tooth-impact rate is much lower than the carrier frequency of the sound radiator and the oscillation is faded away before the stimulation by the next tooth-impact follows [Jat95]. This leads to complex broad frequency calls, often with ultrasonic components over 100 kHz (see Figure 4.5(b)). The wide spectra are due to the large number of impulses produced by each movement of the fore-wings. The frequency range for large katydid species (e.g. *Tettigonia*, *Decticus*, *Ephippiger*) often reveals a lower maximum (between 8 and 16 kHz) and a higher maximum (between 25 and 40 kHz). For small katydid species (e.g. *Conocephalus*, *Metrioptera* and *Leptophyes*) the low frequency components are between 15 and 20 kHz [IK98]. The impulse repetition rate ranges from 500 Hz (slow singers) up to 8 kHz (fast singers) [5] [JSSK94].

In the *Tettigoniidae* each tooth impact produces an short damped oscillation (*impulse*) [CFS98]. Each opening and closing movements of the wings produces a series of sound impulses termed *opening syllable* and *closing syllable*. For most species the closing syllable contains more energy and a higher amplitude than the opening syllable [Ewi89] (see Figure 4.6(b)).

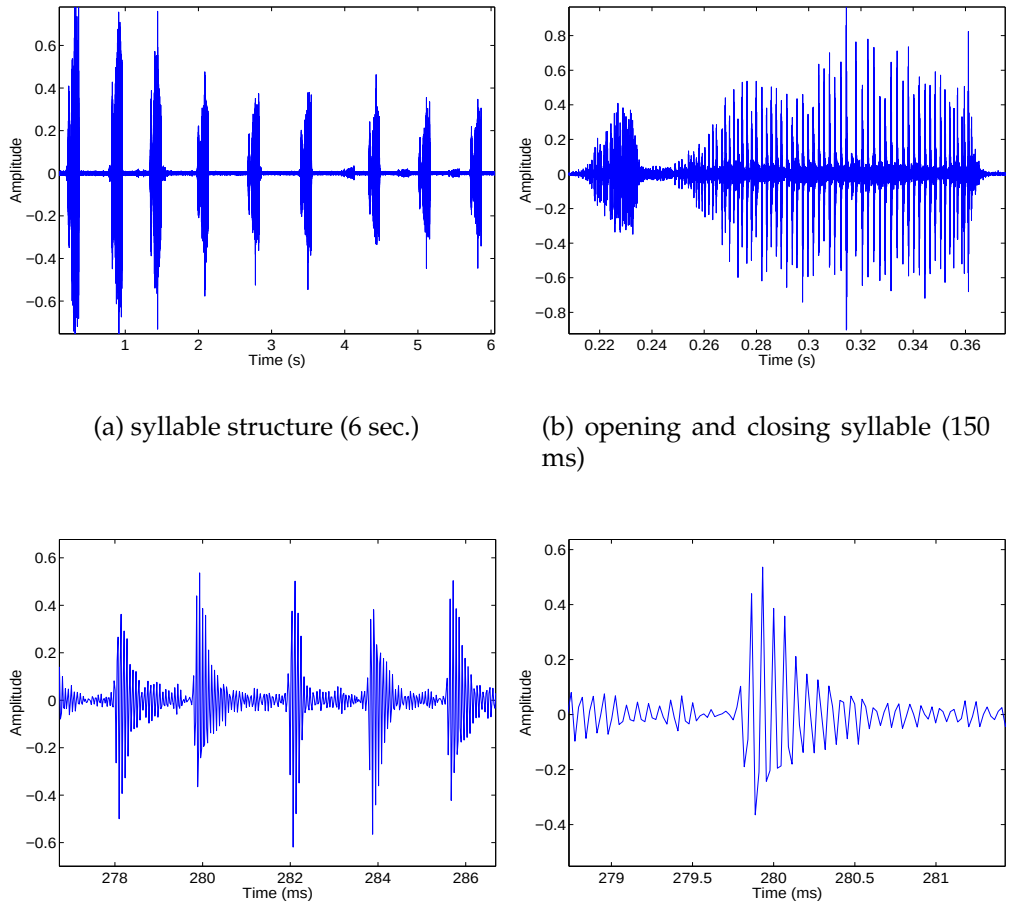
A species-specific number of opening and closing syllables form a verse [JSSK94]. In the *Tettigoniidae* we distinguish between *slow singing species* and *fast singing species* [Jat95]. For slow singing species the velocity of the stridulatory movement of the fore-wings is low, the teeth are scrapping slowly against the plectrum. Most of the impulses of the closing syllable are clearly separated and can be identified individually (see Figure 4.6(c)). On the other hand, for fast singing species the teeth are scrapping fast against the scraper due to a high movement velocity, thus the impulses of the closing syllable are conflating to *impulse groups* [Hel88].

In the *Acrididae*, one movement of the stridulatory mechanism may produce more than one impulse of sound. Acridid grasshoppers generate chirping sound patterns by rasping both hind-legs across their fore-wings at the same time. This leads to complex sounds with frequencies between 5 and 40 kHz [MSP+01].

4.1.4 The Acoustic Receptor System

Acoustic *intraspecific communication* requires a match between signals and receivers, as undetected signals convey no information [MMH02]. Hence, members of the insect order *Orthoptera* have an auditory system with ex-

[5] Impulse distance: slow singers $\approx$ 2 ms; fast singers $\approx$ 0.125 ms



(a) syllable structure (6 sec.)

(b) opening and closing syllable (150 ms)

(c) impulse structure (10 ms)

(d) single impulse (2 ms)

Figure 4.6: Typical time amplitude pattern of the stridulatory signals of the katydid species *Ephippiger ephippiger* in 4 different time resolutions, syllable structure (Figure (a) and (b)) and impulse structure (Figure (c) and (d)).

cellent high-frequency hearing, broad range, and high temporal resolution [JSSK94]. Usually, this high-frequency hearing is even sensitive to the frequency of the acoustic signals.

In the *Orthoptera* the most important receptor organs for the perception of sound and vibration are the complex *tibial organs* of the fore-legs. These tibial organs consist of the *tympanal organ* and the *subgenual organ*. The tympanal organs are only located at the fore-legs and are the essential structures to detect airborne sound. Generally, the structure of the tympa-

nal organs of crickets and katydids differs in that way, that for crickets the eardrums are connected by a tracheal tube through sound waves can propagate. Consequently the vibration of the eardrum reflects the combination of direct sound and delayed sound travelling from the other ear. The relative phase of the waves reflects the direction of the sound source [WH00]. In contrast to the tympanal organs, smaller subgenual organs are also found in the mid and the hind-legs and are used to recognise substrate vibration [IK98].

The morphology of the receptor organs, and the arrangement of the corresponding receptor cells are well known and many studies have been examined. In particular, the frequency response properties of the auditory receptors have been studied [KKK90, RKH90].

Behavioural experiments show that female crickets and katydids are able to locate the species-specific male song. This behaviour is for example mimicked by a "cricket robot" developed by WEBB [Web96]. The robot is able to move toward a sound source by detecting the direction of the sound and correcting its course as it moved toward its destination. Therefore it uses two miniature microphones to measure the *phase delay* in the sound. The control mechanism is based on the cricket neurophysiology [WS00] and follows the hypothesis that the onset times of the spikes generated by auditory interneurons are compared and thus the cricket turns to the side on which the sound is closer.

4.1.5 Acoustic Characteristics

The acoustic signals of the *Orthoptera* are influenced by parameters which warp the sound production and the sound transmission of the airborne species-specific song:

- **Temperature.** In [Wal74], WALKER has shown, that the wing-cycle rate of cricket species is dependent on the temperature. He experimentally determined a linear function for the wing-cycle rate of the cricket species *Atlanticus gibbosus* through linear regression. For the *Atlanticus dorsalis* he found that the wing-cycle rate during stridulation is an exponential function of temperature. This function was determined through exponential regression.

- **Habitat.** Especially the high frequency components of the airborne-sound of the *Tettigoniidae* are affected by attenuation of vegetation and ground [WR78]. This attenuation is the result of a number of different phenomena (reflection, diffraction, multiple scattering and reverberation). These phenomena become particularly important when

the wavelength of the sound corresponds to the size of the intervening structures. In [Hel88] HELLER has found, that the high frequency components of sounds of *Tettigoniids* recorded in plastic cages are damped.

- **Song-breeds.** Several species show unusual variation in the song structure throughout their *song-breed* [Dui90]. One famous example is the katydid species *Ephippiger ephippiger* which produces sounds with one to nine syllables.
- **Age.** Examinations in [Wal74] revealed that the wing-cycle rate depends on the age of the individual. However, this dependence is species-specific.
- **Sex.** The size, number and spacing density of pegs varies according to the individual's sex. Due to the fact that most cricket females remain silent especially species of the *Tettigoniidae* provide sex-specific characters [Jat95].
- **Local time.** Investigations of INGRISCH have shown time dependent temporal variations of songs of the *Podoscirtinae* [Ing97]. He found, that the stridulation patterns produced by the same male of the species *Truljala versicolor* and the same male of the species *Zvenella geniculata* vary depending on the time of the night.
- **Distance.** For songs of the *Tettigoniidae* it has been shown that the relative intensities of low and high frequency components change with the distance between the animal and the microphone [RB86]. Sound spectra at different distances from the sound source are discussed in [KKK90].

In order to suppress the influence of the environmental temperature in the numerical experiments, signal parameters, e.g. pulse distances, syllable distances, pulse length and syllable length (see Eq. 4.25 and Eq. 4.27) are linearly normalised as if they were recorded at 20 °C. For this type of temperature normalisation the environmental temperatures which have been measured during the sound-recording of the individual sound files are used.

4.1.6 Normalisation of the Temperature Influence

Because most insects are *ectotherms* (= animals in which the body temperature is the same as the temperature of its surroundings), the temperature

influences attributes of the acoustic signals that are controlled by the neuromuscular system [GH02]. For crickets and katydids the temperature particularly influences the wing-stroke rate which influences the pulse length and the pulse distance of cricket sounds and the syllable length and the syllable distance of katydid sounds, as shown in many publications [Jat95, Wal73, LW79].

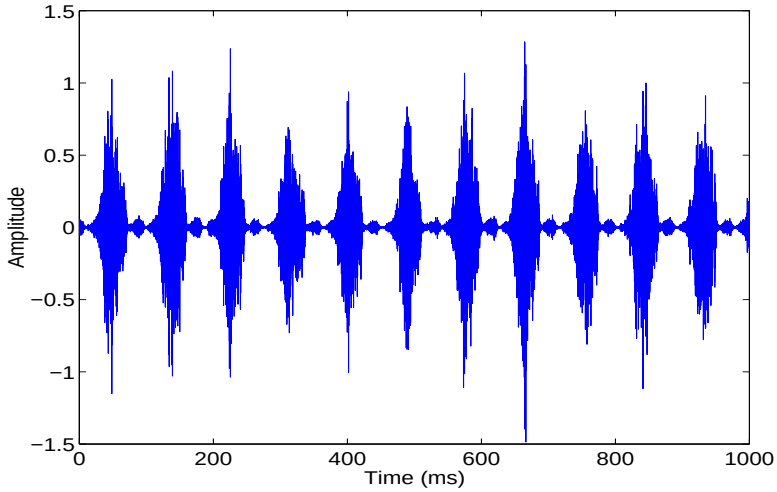
Several models have been developed to explain the temperature influence on these parameters [Pre00]. But, there are also models that explain, that the temperature has no influence on these parameters. The best known model is the *clockwork cricket* (escapement) *model* [EK85, KESK87] which explains this paradox by invoking parallels between escapement in clocks and stridulating crickets. Escapements are devices that regulate energy input and motion. In mechanical clocks, they alternately lock and release the mechanism that moves the clock's hands. The rhythm of lock and release is, in turn, controlled by a regulating oscillator such as a pendulum or balance spring [Pre00]. In the clockwork cricket analogy, the escapement is composed of the stridulatory structures with the harps (see Section 4.1.2) acting as the regulating oscillator.

A katydid species which is often examined in this context is the *Tettigonia cantans*. For this species the syllable length and the syllable distances are highly dependent on the temperature of the environment. Figure 4.7 shows the time amplitude patterns of two individuals of our data set which have been recorded at two different environmental temperatures.

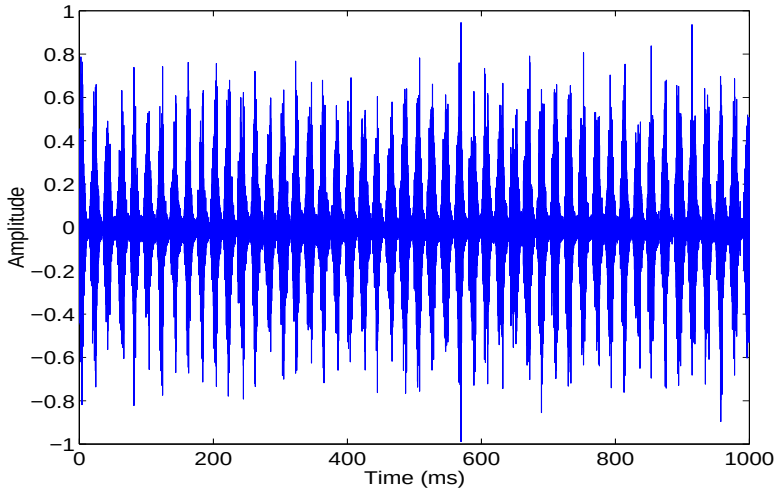
To show this temperature dependence in more detail, the syllable distances and the syllable length for six recordings of the *Tettigonia cantans* recorded at temperatures between 14°C and 26°C including regression lines are depicted in Figure 4.8. The regression lines in Figure 4.8 correspond to the results of JATHO [Jat95] who has examined the temperature influence to signal parameters such as the syllable length and the syllable distance of the species *Gampsocleis gratiosa*, *Tettigonia cantans* and *Tettigonia viridissima*.

To compensate the influence of the environment temperature, signal parameters have to be normalised to a specific temperature. For the *Orthoptera* songs we therefore linearly normalise temperature dependent signal parameters, e.g. the syllable distance (**D**) (see Eq. 4.25) and the syllable length (**L**) (see Eq. 4.27) to a temperature of 20 °C which is in the average temperature range of the recordings of the DORSA data set. But first, we empirically determined the percentage decrease of syllable distances $p_{20}^{\mathbf{D}}$ and syllable length $p_{20}^{\mathbf{L}}$ from the regression lines of the individuals in our data set.

As an example, Figure 4.8 shows the regression lines of the species



(a) 14 °C: (11 syllables per second)



(b) 26 °C: (50 syllables per second)

Figure 4.7: Time amplitude patterns of two individuals of the *Tettigonia cantans* recorded at two different temperatures.

Tettigonia cantans. For this species the percentage decrease of syllable distances p_{20}^{D} and syllable length p_{20}^{L} are determined by considering the regression lines of the *Tettigonia cantans*. Let $\hat{d}(\theta)$ be a syllable distance and $\hat{l}(\theta)$ be the syllable length measured at the regression line at θ degree. Then the

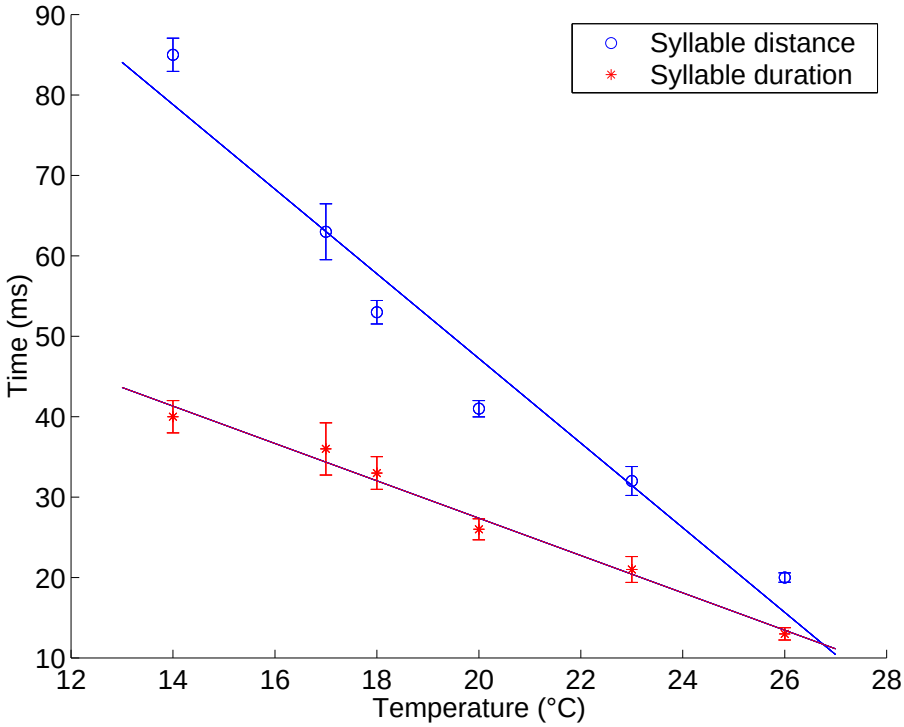


Figure 4.8: Syllable distance (o) and syllable length (*) and the corresponding regression lines of the species *Tettigonia cantans* dependent on the environment temperature.

percentage decrease $p_{20}^{\mathbf{L}}$ and $p_{20}^{\mathbf{D}}$ is simply given by

$$p_{20}^{\mathbf{D}} := \frac{\hat{d}(21) - \hat{d}(20)}{\hat{d}(20)} \qquad p_{20}^{\mathbf{L}} := \frac{\hat{l}(21) - \hat{l}(20)}{\hat{l}(20)}. \tag{4.1}$$

Unfortunately $p_{20}^{\mathbf{D}}$ and $p_{20}^{\mathbf{L}}$ are species dependent. For the *Tettigonia cantans* $p_{20}^{\mathbf{D}} = 0.11$ and $p_{20}^{\mathbf{L}} = 0.085$. But for most species in our data set (see Section 4.3) the syllable distances and the syllable length are not as dependent from the environmental temperature and therefore the parameters $p_{20}^{\mathbf{D}}$ and $p_{20}^{\mathbf{L}}$ are much smaller.

However, this analysis allows to linearly normalise any signal parameter if the environment temperature and the corresponding percentage decrease is given. Let ψ be the signal parameter measured at 20 °C and p_{20}^{ψ} be the species-specific percentage decrease of ψ. By assuming that the signal parameter ψ is linearly dependent of the environment temperature the signal parameter ψ at ϑ degree ψ^{ϑ} is simply given by

$$\psi^{\vartheta} = \psi - \psi(\vartheta - 20)p_{20}^{\psi}. \tag{4.2}$$

Conversely, the linearly normalised signal parameter at 20 °C is given through

$$\mathcal{O}^{\vartheta}_{p^{\psi}_{20}}(\psi^{\vartheta}) := \frac{\psi^{\vartheta}}{1-(\vartheta-20)p^{\psi}_{20}}. \tag{4.3}$$

Hereby $\mathcal{O}^{\vartheta}_{p^{\psi}_{20}}(\cdot)$ is a normalising operator which linearly normalises the signal parameter ψ^{ϑ}, recorded at ϑ degree with respect to p^{ψ}_{20}. Exceptions for this type of linear temperature normalisation are some katydid species, in which singing itself can generate heat in the stridulatory muscles that raises the animal's body temperature above ambient temperature [GH02].

4.2 Hierarchical Feature Extraction in Bioacoustic Time Series

For the classification of *Orthoptera* songs, the sounds must be transformed into feature vectors suited for the pattern recognition system. A pattern recognition system can be considered as a two stage system: (1) the extraction of features from the time signal and (2) the classification of the extracted feature vectors [BJ92]. In the feature extraction, different *feature detectors* are defined in order to calculate a set of characteristic properties of the signal which are used as input for the classifier. In the context of the hierarchical classification of *Orthoptera* species we developed a more complex system including signal pre-processing and filtering, feature extraction at two levels, time series classification over a set of time scales and fusion of the individual time scales (see algorithm HOC in Figure 4.9).

Algorithm HOC receives a waveform $(s(t))_{t=1}^{T}$ and additionally the corresponding environmental temperature ϑ and the family inside the *Ensifera* $\omega_1 \in \Omega^1$ (see Figure 4.1) as input. The result of algorithm HOC are two class labels from two different levels: (1) the family level ω_1 and (2) the species level ω_2 of the individual to be classified.

The hierarchical *Orthoptera* classification works as follows: After preprocessing the signals (see step (a) and Section 4.2.1), onset and offset positions $\lambda = (\lambda_1, ..., \lambda_J)$ and $\mu = (\mu_1, ..., \mu_J)$ of the individual pulses are calculated from the pre-processed signal $(\bar{s}(t))_{t=1}^{T}$ by a signal segmentation algorithm (see step (b) and Section 4.2.4). First level feature extraction (step (c) and Section 4.2.2) and first level time series classification (see step (d) and Section 3.4) to family level is applied if the family ω_1 is not given. In the feature extraction a set of feature streams $(\hat{X}(j))_{j=1}^{\mathcal{J}}$ (see Figure 4.16) is

Algorithm (ω_1, ω_2) = HOC$((s(t))_{t=1}^{T}, \vartheta = 20, \omega_1 = \emptyset)$

(a) $\bar{s}$ = PREPROCESSING(s)
if $\omega_1 == \emptyset$
for $\upsilon \in \{1, ..., \Upsilon\}$
(b) (λ, μ) = SEGMENTATION($\bar{s}$, υ)
(c) $(\hat{X}(j))_{j=1}^{\mathcal{J}}$ = FEATUREEXTRACTION($\bar{s}$, ϑ, υ, λ, μ)
(d) $\mathbf{z}^{\upsilon}$ = CLASSIFICATION $((\hat{X}(j))_{j=1}^{\mathcal{J}})$
end
(e) $\mathbf{z} = \mathcal{F}(\mathbf{z}^1, ..., \mathbf{z}^{\Upsilon})$
$\omega_1 = \underset{l \in \Omega^1}{\text{argmax}}(\mathbf{z}_l)$
end
(a′) $\tilde{s}$ = FILTERING($\bar{s}$, ω_1)
for $\upsilon \in \{1, ..., \Upsilon_{\omega_1}\}$
(b′) (λ, μ) = SEGMENTATION($\tilde{s}$, υ, ω_1)
(c′) $(X(j))_{j=1}^{\mathcal{J}}$ = FEATUREEXTRACTION ($\bar{s}$, ϑ, υ, λ, μ, ω_1)
(d′) $\mathbf{z}^{\upsilon}$ = CLASSIFICATION $((X(j))_{j=1}^{\mathcal{J}})$
end
(e′) $\mathbf{z} = \mathcal{F}(\mathbf{z}^1, ..., \mathbf{z}^{\Upsilon_{\omega_1}})$
$\omega_2 = \underset{l \in \Omega^2_{\omega_1}}{\text{argmax}}(\mathbf{z}_l)$
—

Figure 4.9: Algorithm HOC (Hierarchical *Orthoptera* classification): The feature extraction and classification to the family level (level 1) is accomplished in step (a) to step (e). Similar processing steps (step (a′) to step (e′)) are given for the feature extraction and classification to the species level (level 2).

extracted from Υ different time scales from the pre-processed waveform $\bar{s}$ to determine the family of the individual. After classifying feature vectors of the individual time scales to family level the local classifier decisions for the whole time series $\mathbf{z}^1, ..., \mathbf{z}^{\Upsilon}$, with $\mathbf{z}^{\upsilon} \in \Delta$ are combined through a fixed mapping in step (e). Then the family is calculated by the maximum membership rule. In our application $\Omega^1 = \{1, 2\}$, whereas $\omega_1 = 1$ denotes that the individual is a cricket species and $\omega_1 = 2$ denotes that the individual is a katydid species.

From now on the processing steps are family-specific and depend on

the family of the individual given in ω_1. At first, a family-specific filter is applied to reduce environmental noise (see step (a′) and Section 4.2.3). For each of the Υ_{ω_1} time scales (see Section 4.2.7) onset and offset positions $\lambda = (\lambda_1, ..., \lambda_J)$ and $\mu = (\mu_1, ..., \mu_J)$ are computed from the filtered signal $\tilde{s}$ (see step (b′) and Section 4.2.4). Further local features are extracted (see step (c′) and Section 4.2.5) and the individual feature streams are classified (see step (d′) and Section 3.4 and Section 3.5). Hereby $\mathbf{z}^{\upsilon} \in \Delta$ (see Eq. 3.5) is the probabilistic classification result of a whole time series in the υ-th time scale. These probabilistic classification results are combined (see step (e′)) and the class of the individual $\omega_2 \in \Omega^2_{\omega_1}$ is determined through the maximum membership rule. In our application for $\omega_1 = 1$, Ω^2_1 contains the class labels of the species in the cricket data set (see Table D.2). For $\omega_1 = 2$, Ω^2_2 contains the class labels of the species in the katydid data set (see Table D.1).

For both feature extraction levels different *feature detectors* are defined in order to calculate the characteristic signal properties which are used as input features for the classifiers of the current level. In Section 4.2.2 and Section 4.2.5 the evaluated features are described and their calculation is given:

1. Filter-bank energies (**B**) (see Section 4.2.2)
2. Pulse distances (**D** and $\bar{\mathbf{D}}$) (see Section 4.2.5)
3. Pulse lengths (**L**) (see Section 4.2.5)
4. Pulse frequency contours (**C**) (see Section 4.2.5)
5. Pulse frequencies (**F**) (see Section 4.2.5)
6. Energy contours of pulses (**E** and $\hat{\mathbf{E}}$) (see Section 4.2.5)
7. Time encoded signals of pulses (**T**) (see Section 4.2.5)
8. Maximal amplitudes of pulses (**A**) (see Section 4.2.5)
9. Parzen density functions of pulse distances (**P**) (see Section 4.2.6)

We use boldface upper-case letters for the acoustic features and distinguish between (1) *spectral features* (**B**, **C**, **F** and **T**) which are calculated in the frequency domain and (2) *temporal features* (**D**, $\bar{\mathbf{D}}$, **L**, **E**, $\hat{\mathbf{E}}$, **A** and **P**) that are extracted from the time course of the signal's amplitude. Beside the filter-bank energies (**B**) all other features depend on the pulse positions determined by algorithm SEGMENTATION. Parameter settings for the first

level feature extractors are given in Table C.1, whilst Table C.2 and Table C.3 contains the settings for the second level feature extractors. The evaluation of features for the first and the second hierarchy level is given in Section 4.4.

4.2.1 Pre-processing

In the pre-processing stage the signal $s(\cdot)$ is resampled to a common sample rate and the signal's amplitude is normalised to suppress the influence of the sound volume.

Resampling is applied by a polyphase implementation of fractional sampling rate conversion (FSRC) as proposed in [GGF99]. In the case of rational resampling by a factor $R = \frac{q}{Q}$ with $q \in \mathbb{N}$, $Q \in \mathbb{N}$ and $q < Q$, the input signal has to be expanded by q, bandlimited to compensate for imaging and aliasing effects and then compressed by Q.

Let $\hat{s}(t), t = 1, ..., T$, be the finite resampled signal containing the sound pattern of a single individual. To prevent the influence of the sound volume for the following feature extraction procedure, the signal its amplitude is normalised by

$$\bar{s}(t) = \frac{\hat{s}(t)}{||\hat{s}||_\infty} \tag{4.4}$$

where $||\hat{s}||_\infty = \max\{|\hat{s}(t)| : t = 1, ..., T\}$ denotes the maximum norm.

4.2.2 First Level Feature Extraction

This Section deals with the extraction of features to classify to family level. Hereby individuals of both families (crickets and katydids) are separated (see algorithm HOC, level 1). Whereas the calculation of features to classify to family level is given in this Section, features applied in both feature extraction levels (see algorithm HOC, level 1 and 2) are only mentioned and their calculation is given in Section 4.2.5.

Filter-Bank Energies (B)

Due to the differences in the frequency composition of the stridulatory signals of the two families (see Figure 4.5) a linear uniform filter-bank is utilised to extract *filter-bank energy contours* (**B**). This type of filter-bank is the most common type of filter-bank in speech recognition systems [RJ93]. One reason for this is, that it is quite robust to noise and reverberation.

In contrast to all other local features proposed in this thesis the filter-bank energy contours are not based on the positions determined by algorithm SEGMENTATION. For this feature the sound of the animal is located by determining the positions of the signal-segments containing maximal energy (see Eq. 4.10).

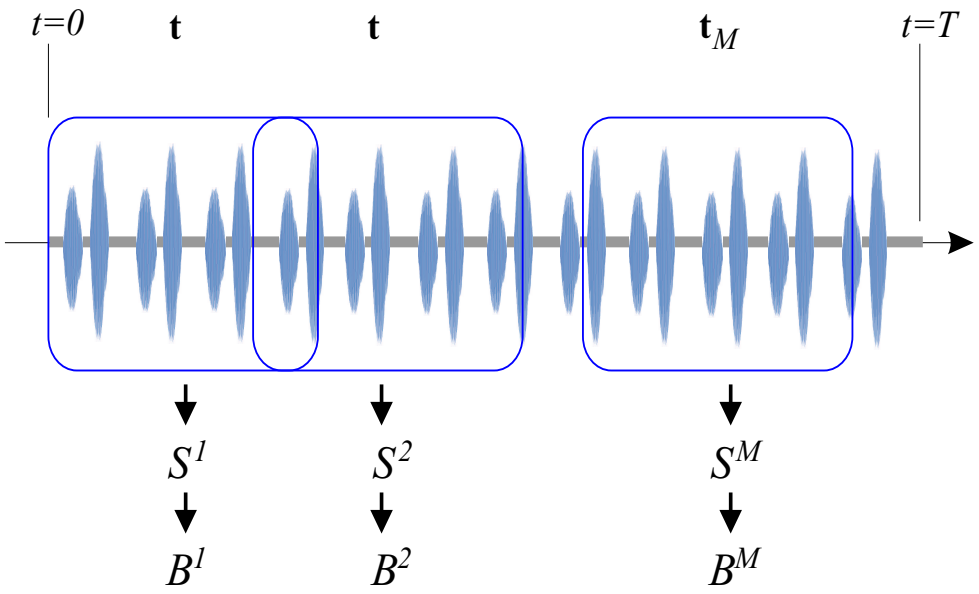


Figure 4.10: From the signal $s(t)_{t=1}^{T}$ a sequence of local spectra $S^1, ..., S^M$ is extracted from M equally spaced short time windows. From each local spectra a filter-bank energy contour $\mathbf{B}^m \in \mathbb{R}^D$, $m = 1, ..., M$ is calculated.

Let $\bar{s}(t), t = 1, ..., T$ be the resampled and normalised signal which contains the sound of the animal. The m-th short time spectrum S^m is then given by applying the Fourier transformation $\mathcal{F}$ [VHH98] to the windowed signal $\bar{s}$. It is given by

$$S^m(\omega) = \mathcal{F}\{\bar{s}(t)\mathcal{W}_{\mathbf{t}_m}^{V,1}(t)\}, \quad m = 1, ..., M \tag{4.5}$$

where $\mathbf{t} = (\mathbf{t}_1, ..., \mathbf{t}_M)$ with $M = \lfloor \frac{T-V}{\beta V} \rfloor + 1$ is a sampling vector which determines the positions of the individual windows (see Figure 4.10). In order to determine window positions with equal distances between two consecutive windows the sampling vector $\mathbf{t}$ is defined through

$$\mathbf{t}_m := \frac{V}{2} + (m-1)\beta V. \tag{4.6}$$

Hereby βV determines the constant movement between two consecutive windows and

$$\mathcal{W}_t^{V,\kappa}(\tau) = \begin{cases} 1 - \frac{2(1-\kappa)|t-\tau|}{V}, & \tau \in [t - \frac{V}{2}, t + \frac{V}{2}] \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

is a symmetrical window of size V at time t (see Figure 4.11).

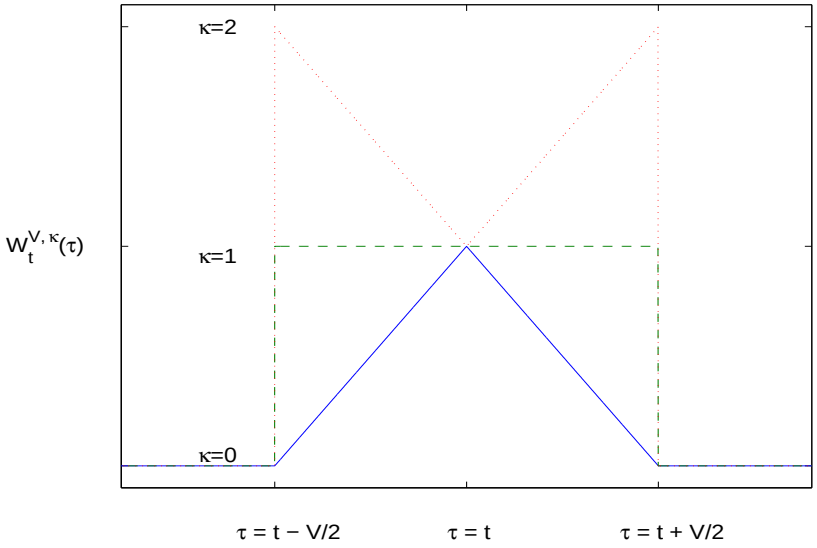


Figure 4.11: A symmetrical window $\mathcal{W}_t^{V,\kappa}(\tau)$ of size V at time t. Three windows for three different settings for κ are shown ($\kappa = 0$, $\kappa = 1$ and $\kappa = 2$).

For this window the parameter $\kappa \in [0, \infty)$ determines $\mathcal{W}_t^{V,\kappa}(\tau) = \kappa$ at $|t - \tau| = \frac{V}{2}$. Therefore, for $\kappa = 0$ the window is triangular (solid line) and for $\kappa = 1$ the window is rectangular (dashed line).

The spectral energy inside the d-th frequency channel of the m-th short time spectrum (see Eq. 4.5) is then given through

$$\mathbf{B}_d^m := \sum_{\omega=f_d-\frac{b}{2}}^{f_d+\frac{b}{2}} |S^m(\omega)| \underbrace{\frac{1}{2}\left(1 - \cos\left(2\pi \frac{\omega - (f_d - \frac{b}{2})}{b}\right)\right)}_{H_d(\omega)}. \tag{4.8}$$

Hereby $H_d(\omega)$ is the Hann window [Ham77] which covers the d-th frequency channel and b is the constant frequency range of the individual frequency channels whose center frequencies are defined by f_d, $d = 1, ..., D$.

To calculate the individual center frequencies, let f_1 be the first center frequency, f_D be the last center frequency and D be the number of frequency channels. Then all other center frequencies are given by

$$f_d = f_1 + (d-1)\frac{f_D - f_1}{D-1}, \quad d \in \{2, ..., D-1\}. \tag{4.9}$$

Usually, the individual frequency channels do overlap in frequency, e.g. $b > f_d - f_{d-1}$ (see Figure 4.12) and the frequency range of interest is given through $[f_1 - \frac{b}{2}, f_D + \frac{b}{2}]$.

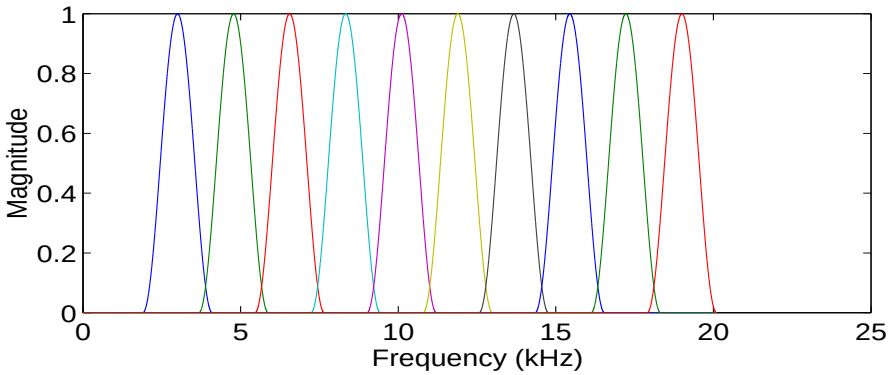


Figure 4.12: Window sequence of the linear filter-bank consisting of D = 10 frequency channels (f_1 = 3000 Hz, f_D = 19000 Hz and b = 2000 Hz).

To extract $J \leq M$ characteristic filter-bank energy contours from $\bar{s}$ we consider the J filter-bank energy contours containing maximal energy, e.g. there is a sequence $\tau_1, ..., \tau_M$ such that

$$\hat{B}^{\tau_1} \geq ... \geq \hat{B}^{\tau_J} \geq ... \geq \hat{B}^{\tau_M} \tag{4.10}$$

where $\hat{B}^m = \sum_{d=1}^{D} \mathbf{B}_d^m$. The sequence of feature vectors is then given by $\mathbf{B}^{\tau_1}$, ..., $\mathbf{B}^{\tau_J}$ with $\mathbf{B}^{\tau_j} \in \mathbb{R}^D$ (see Eq. 4.8 and algorithm HOC step (c)). But the overall filter-bank spectrum analysis also performs segmentation of the acoustic signals (see algorithm HOC step (b)) and returns a sequence of onsets $\lambda_{\tau_j} := \mathbf{t}_{\tau_j} - \frac{R}{2}$ and offsets $\mu_{\tau_j} := \mathbf{t}_{\tau_j} + \frac{R}{2}$, $j = 1, ..., J$.

Features for Both Feature Extraction Levels

The filter-bank energy contours are combined with features which are used in the first and the second feature extraction level. In particular, the *pulse distances* and the *pulse length* (see Section 4.2.5) are used in both feature extraction levels.

In the numerical experiments the features in the first hierarchy level are derivated by using the parameter settings given in Table C.1.

4.2.3 Signal Filtering

To suppress the influence of environmental noise the normalised and resampled raw time signal $\bar{s}$ is filtered. In the signal filtering we distinguish between the families crickets and katydids due to differences in the frequency composition (see Figure 4.5 and Section 4.1.2):

Crickets produce signals restricted to a narrow band (see Figure 4.13(b)) because the sounds are strongly resonant [PW81]. To get rid of the sounds of the environment the signal is filtered with a highly selective bandpass-filter, the passband can be determined automatically at the frequency range of the cricket.

The automated filtering works as follows: First the frequency with the highest intensity is determined through

$$\omega^* = \underset{\omega}{\operatorname{argmax}} |\bar{S}(\omega) W(\omega)| \tag{4.11}$$

where $W(\omega)$ is a probability density function defined in the frequency range of the cricket's sounds and $\bar{S}(\omega) = \mathcal{F}\{\bar{s}(t)\}$ is the frequency spectrum calculated through Fourier transform of the normalised signal $\bar{s}$ (see Eq. 4.4). In the numerical experiments $W(\omega)$ is given through the Gaussian bell function [Pea88]

$$W(\omega) := \frac{1}{f_\sigma \sqrt{2\pi}} \exp\Big(-\frac{(\omega - f_{max})^2}{2 f_\sigma^2}\Big). \tag{4.12}$$

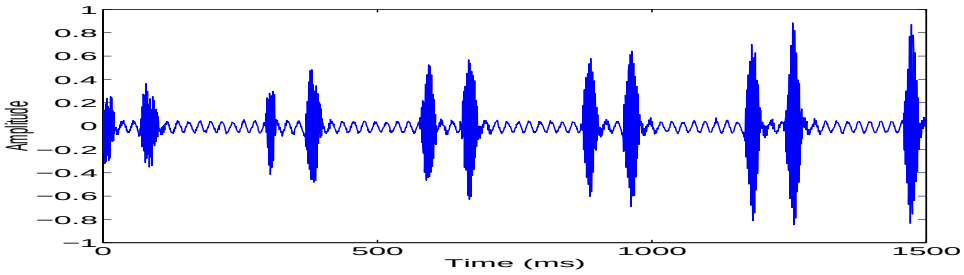
For ω^* the transfer function (or frequency response) of the highly selective bandpass-filter [Ham77] is defined through

$$H_\gamma(\omega) = \begin{cases} 1, & |\omega| - |\omega^*| < \gamma \\ 0, & \text{otherwise} \end{cases} \tag{4.13}$$

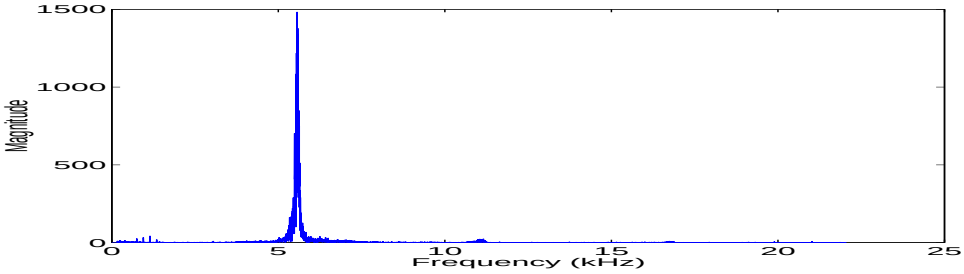
where 2γ is the size of the passband. In order to calculate the filtered signal (see Figure 4.13(c)), the bandpass-filter $H_\gamma(\omega)$ is applied in the frequency domain

$$\tilde{s}(t) = \mathcal{F}^{-1}\{\bar{S}(w) H_\gamma(\omega)\}. \tag{4.14}$$

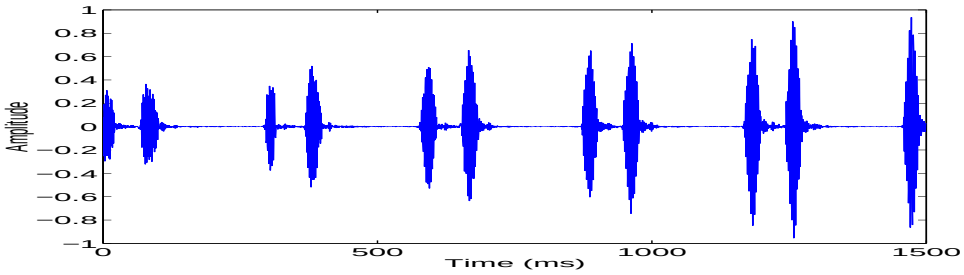
Hereby $\mathcal{F}^{-1}\{\cdot\}$ denotes the inverse Fourier transform [VHH98]. The filtered signal $\tilde{s}(t)$ is limited to the frequency range $[\omega^* - \gamma, \omega^* + \gamma]$, which is assumed to contain the sounds of the cricket [DSRP01]. After resampling, normalisation and filtering the signal $\bar{s}(t)$ with $H_\gamma(\omega)$, the filtered signal $\tilde{s}(t)$ is free from background sounds, such as the voices of birds.



(a) The waveform $\bar{s}(t)$ after normalisation



(b) The Fourier-spectrum $\bar{S}(\omega)$



(c) The filtered signal $\tilde{s}(t)$

Figure 4.13: A short song of a cricket from the species *Noctitrella glabra* (time window 1500 ms).

Most of the **Katydids** produce wide band spectra, often up to the ultrasonic range (see Section 4.1.2). Here a bandpass-filter is applied to suppress the low frequency components (often environmental noise) and higher frequency components which are sensitive to the influences of the surroundings (see Section 4.1.2) and the equipment used for the sound recordings (see Section 4.3).

In contrast to cricket sounds (see Eq. 4.14), the katydid sounds are filtered with a bandpass-filter whose passband is wide and constant. It is given by the Gaussian bell function which is used as symmetrical transfer function

$$W'(\omega) := \exp\left(-\frac{(|\omega| - f_{max})^2}{2f_\sigma^2}\right). \tag{4.15}$$

Then the frequency spectrum $\bar{S}(\omega) = \mathcal{F}\{\bar{s}(t)\}$ of the normalised and resampled signal $\bar{s}(t)$ (see Eq. 4.4) is filtered in the frequency domain

$$\tilde{s}(t) = \mathcal{F}^{-1}\{\bar{S}(w)W'(\omega)\}. \tag{4.16}$$

Again the inverse Fourier transform $\mathcal{F}^{-1}\{\cdot\}$ is applied to calculate the signal in the time domain. For both filters (see Eq. 4.14 and 4.16) the filtered signal is mainly used for the signal segmentation through the short time energy (see Section 4.2.4). The computational cost and the desired memory was reduced in the numerical experiments by applying the filters to signal segments of two seconds.

4.2.4 Signal Segmentation Through the Signal's Energy

The unaided human ear neither resolves the temporal structure nor the full frequency range of insect songs [Rie98]. But in *automated bioacoustics* the temporal structure of the pulses is an important feature for the classification of the *Orthoptera* to species level [Nis99]. Furthermore, these pulse positions are significant for the classification to the family level. Therefore the detection of pulses is a significant issue in the following feature extraction and classification task. However, the detection of pulses may be considered as a general problem in signal processing, because the detection of the presence of a signal within a background of noise is important in many applications. This problem is often referred as the *end point location problem* [RS75]. In speech recognition systems end point detection is used to get the relevant parts of the speech signal where the speaker is active. We use end point detection to locate the insect's pulses.

RABINER and SAMBUR [RS75] published an end point detection algorithm for speech segmentation using the short time signal energy and the zero-crossing-rate of the signal to localise the onset and offset positions of a speech utterance. This algorithm can be used in almost any background environment with a signal-to-noise ratio of at least 30 dB.

However, there are differences between human speech and animal vocalisations, and the different conditions under which they are recorded.

These differences are significant and must be taken into account [KM98]. For pulse segmentation we therefore use an algorithm which determines the onset and offset positions of the pulses of *Orthoptera* songs. The algorithm is similar to that of RABINER and SAMBUR, but with significant modifications, particularly for the calculation of the threshold function, the silence energy and the peak energy.

The result of the end point detection algorithm are the onsets and the offsets of the individual pulses. These pulse positions are then used to derive local features within these pulses (see Section 4.2.5).

For the pulse detection algorithm the total energy E of the previously filtered signal $\tilde{s}$ (crickets: see Eq. 4.14 and katydids: see Eq 4.16) is defined by [RS75]

$$E = \sum_{\tau=-\infty}^{+\infty} |\tilde{s}(\tau)| \tag{4.17}$$

and the *short time energy* of the signal $\tilde{s}$ inside a rectangular window $\mathcal{W}_t^{U,1}$ of size U at time t (see Eq. 4.7) is given through

$$E_U(t) = \sum_{\tau=-\infty}^{+\infty} |\tilde{s}(\tau)| \mathcal{W}_t^{U,1}(\tau). \tag{4.18}$$

In addition, a *threshold function* $F_a(t)$ is used to determine the pulses of the signal of the *Orthoptera*. In speech recognition this threshold $F_a(t)$ is typically a constant function [RS75]. But for insect vocalisations we observed that for several species the short time energy function $E_U(t)$ increases inside chirps and syllables. In Figure 4.14 a typical energy function $E_U(t)$ is shown where the minimal energy between two consecutive pulses gets higher within chirps. Therefore, a dynamic threshold function

$$F_a(t) = \theta(t) + \frac{a}{\xi} \underbrace{\sum_{\tau=-\infty}^{+\infty} |\tilde{s}(\tau)| \mathcal{W}_t^{V,1}(\tau)}_{E_V(t)} + \zeta \tag{4.19}$$

with minimum threshold $\theta(t)$ (see Eq 4.22), parameter $a > 0$, scaling parameter $\xi > \frac{V}{U}$, rectangle window size $V > U$ and minimum energy $\zeta > 0$ is applied for the detection of the onsets. In many applications, such a threshold function only depends on the short time energy function of the filtered time signal $\tilde{s}$ in a finite rectangular time window $\mathcal{W}_t^{V,1}$.

But, this function also depends on the short time silence energy and the short time peak energy. The *silence energy* $E_{min}(t)$ and the *peak energy*

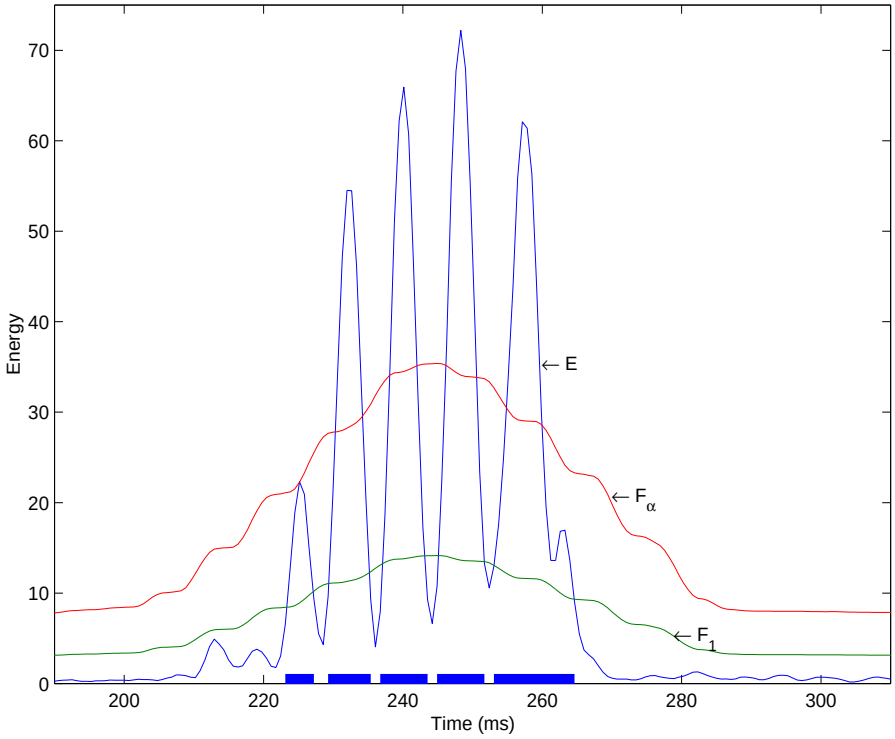


Figure 4.14: Detection of 5 pulses inside a single chirp of the species *Zvenella transversa*. The Figure shows two threshold functions F_1, F_α and the local energy function E_U. Onsets and offset positions of the detected pulses are indicated by the bars at the bottom.

$E_{\max}(t)$ inside two rectangular windows $\mathcal{W}_t^{W,\kappa_{\min}}$ and $\mathcal{W}_t^{W,\kappa_{\max}}$ with $\kappa_{\min} = \kappa_{\max} = 1$ and $W > U$ are given through

$$E_{\min}(t) = \min_{\tau}(E_U(\tau)\mathcal{W}_t^{W,\kappa_{\min}}(\tau)) \tag{4.20}$$

$$E_{\max}(t) = \max_{\tau}(E_U(\tau)\mathcal{W}_t^{W,\kappa_{\max}}(\tau)). \tag{4.21}$$

In order to imply smoothing to the local peak energy and the local silence energy a triangular window ($\kappa_{\max} \in [0, 1)$) is applied to calculate $E_{\max}(t)$ and a mirrored triangular window ($\kappa_{\min} \in (1, 2]$) is applied to calculate $E_{\min}(t)$. The minimal threshold $\theta(t)$ (see Eq. 4.19 and [RS75]) is then defined by

$$\theta(t) = E_{\min}(t) + \gamma(E_{\max}(t) - E_{\min}(t)) \tag{4.22}$$

with $0 < \gamma < 1$.

```
Algorithm λ = SEGMENTATION(E_U, F_1, F_α)
      τ = 0; j = 0; A =TRUE
(a)   for t = V/2, ..., T − V/2
(b)       if (E_U(t) ≥ F_1(t))
(c)           if (τ > 0)
                  if (A =TRUE )
(d)                   if (E_U(t) ≥ F_α(t))
                          j = j + 1; λ_j = t*; A =FALSE
                      else
(e)                       if (τ > ψ) τ = 0
                          else τ = τ + 1 end
                      end
                  end
              else
(f)               τ = τ + 1; t* = t
              end
          else
(g)           τ = 0; A =TRUE ;
          end
      end
—
```

Figure 4.15: Algorithm SEGMENTATION: Searching the onsets $(\lambda_j)_{j=1}^{J}$ of the acoustic signal $\tilde{s}(\cdot)$ by using a a short time energy function E_U and two threshold functions F_1 and F_α.

The estimation of onsets $\lambda = (\lambda_1, ..., \lambda_J)$ through the energy function $E_U(t)$ and $F_\alpha(t)$ (see Eq. 4.19) is accomplished by algorithm SEGMENTATION (see Figure 4.15). Two dynamic threshold functions (see Eq. 4.19): The *lower energy threshold* $F_1(t)$ and the *upper energy threshold* $F_\alpha(t)$ with $\alpha > 1$ are used, obviously $F_1(t) < F_\alpha(t)$. By passing in discrete time steps (see step (a)) through the energy function E_U and both threshold functions F_1 and F_α, in each time step it is decided if the current position is an onset (see step (d)) or not (see step (e), (f) and (g)).

The detection of onsets works as follows: To detect an onset, the short time energy E_U has to exceed the lower energy threshold F_1 (see step (b)). Then the condition in step (c) determines if the current position at time t is really an inflection point between $E_U(t)$ and $F_1(t)$. For $\tau = 0$ this condition

is true and the position of the inflection point is preliminarily assigned to t^* (see step (f)), unless the energy falls below $F_1(t)$ before it rises $F_\alpha(t)$ (see step (g)). If $E_U(t)$ also exceeds the upper energy threshold $F_\alpha(t)$ (see step (d)) within a specified time interval $\tau \leq \psi$, the previously determined inflection point t^* is assigned to the position of the j-th onset λ_j.

A similar search procedure is used to estimate the offset locations $\mu = (\mu_1, ..., \mu_J)$ of the pulses. Both algorithms are only based on short time energy measurements (see Eq. 4.18 and 4.19). The results of the signal segmentation algorithm is J, the number of pulses in the signal, the sequence of onsets $\lambda = (\lambda_1, ..., \lambda_J)$ and the sequence of offsets $\mu = (\mu_1, ..., \mu_J)$ of the detected pulses. These positions are the bases to extract $\mathcal{J} \leq J$ local feature vectors as described in the following section.

In the numerical experiments the pulse segmentation algorithm is used with the parameters listed in Tables C.1, C.2 and C.3. It should be noticed that these parameters are completely different to the parameters used in speech recognition systems. For example, the window size U is much smaller than the window size used in speech recognition systems ($U \approx$ 10 ms) [RJ93].

4.2.5 Local Features From Sequences of Pulses

The general framework for the extraction of local features from time series (see Figure 3.3) describes the extraction of a set of local features inside a sliding window which usually moves over the whole time series. In the following we refine this framework for the derivation of local features from sequences of pulses typical for the structure of *Orthoptera* songs (see Figure 4.4 and 4.6).

For a sequence of onsets $\lambda = (\lambda_1, ..., \lambda_J)$ and offsets $\mu = (\mu_1, ..., \mu_J)$, $\mathcal{J} = J - \Lambda + 1$ multivariate local features $(\mathbf{x}_i(\jmath))_{\jmath=1}^{\mathcal{J}}$, $i \in \{1, ..., I\}$ are extracted inside sliding windows $W^\jmath$, $\jmath = 1, ..., \mathcal{J}$ covering Λ consecutive pulses (see Figure 4.16). Hereby each multivariate local feature $\mathbf{x}_i(\jmath)$ is a composition of pulse features, extracted inside single pulses within the window $W^\jmath$ which covers the $\jmath$-th pulse to the $(\jmath+\Lambda-1)$-th pulse. The compositions of these pulse features is applied by a D-tuple encoding scheme (see Eq. 4.25, 4.26, 4.27, 4.36 and 4.42) producing D-dimensional feature vectors or by averaging the individual pulse features (see Eq. 4.35, 4.37, 4.39 and 4.41). If the pulse features are combined with averaging, $G \leq \Lambda$ defines the number of pulses within $W^\jmath$ which are used for the calculation of the feature vectors.

The parameter settings for the individual features can be found in the

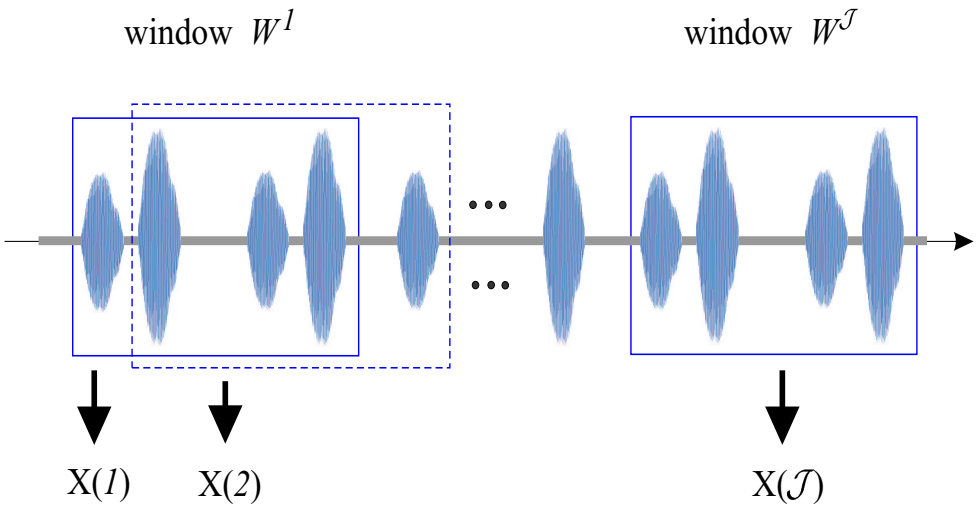


Figure 4.16: A set of I features $X(\jmath) = (\mathbf{x}_1(\jmath), ..., \mathbf{x}_I(\jmath))$ is extracted from the $\jmath$-th local time window $W^\jmath$ covering Λ adjacent pulses. The first, second (dashed line) and the last window is depicted.

Tables C.1, C.2 and C.3.

Pulse Distances (D, $\bar{\text{D}}$)

The pulse distances are derived by using the onsets of the individual pulses $\lambda = (\lambda_1, ..., \lambda_J)$. Then the distance between the j-th and the $(j+1)$-th pulse is simply given by

$$\delta_j = \lambda_{j+1} - \lambda_j, \quad j \in \{1, ..., J-1\}. \tag{4.23}$$

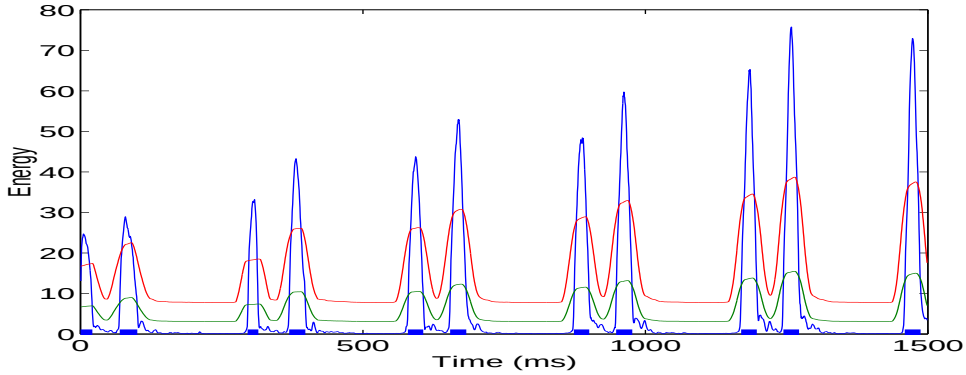
In Figure 4.17(b) a typical sequence of pulse distances of the species *Noctitrella glabra* is depicted. Additionally, the short time energy functions for the pulse detection based on algorithm SEGMENTATION are shown in Figure 4.17(a). The corresponding amplitude signal is given in Figure 4.13.

To derive pulse distances from $D + 1 \leq \Lambda$ consecutive pulses, these features are extracted by using a $D-$tuple encoding scheme producing $\mathcal{J}$ feature vectors $\hat{D}^\jmath \in \mathbb{R}^D$

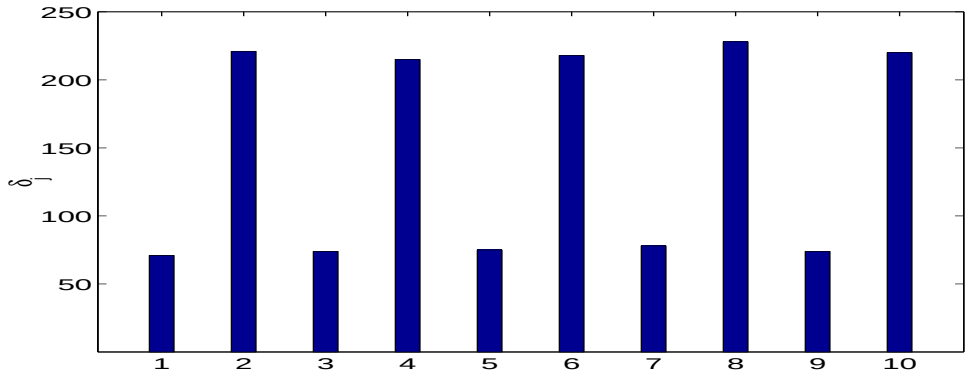
$$\hat{D}^\jmath := (\delta_\jmath, \delta_{\jmath+1}, ..., \delta_{\jmath+D-1}), \quad \jmath = 1, ..., \mathcal{J} \tag{4.24}$$

which may be used for the automated classification. For syllable distances of katydid species and pulse distances of crickets an additional temperature normalisation (see Section 4.1.6) by

$$\mathbf{D}^\jmath := (\mathcal{O}^\vartheta_{p^{\mathrm{D}}_{20}}(\delta_\jmath), \mathcal{O}^\vartheta_{p^{\mathrm{D}}_{20}}(\delta_{\jmath+1}), ..., \mathcal{O}^\vartheta_{p^{\mathrm{D}}_{20}}(\delta_{\jmath+D-1})) \in \mathbb{R}^D, \quad \jmath = 1, ..., \mathcal{J} \tag{4.25}$$



(a) The energy functions for signal segmentation.



(b) The corresponding pulse distances.

Figure 4.17: Pulse detection and corresponding distances between pulses of the species *Noctitrella glabra* (signal length 1.5 sec). Whereas in (a) the two energy thresholds $F_1(t)$, $F_\alpha(t)$, the energy function $E_U(t)$ and the pulse positions are depicted, (b) shows the distance between two consecutive pulses δ_j.

allows to calculate species-specific feature vectors if the environment temperature ϑ and the percentage decrease p_{20}^{D} is given (see Eq. 4.3). In our numerical experiments this type of temperature normalisation improved the classification performance of this feature.

But, the influence of the temperature can be also decreased by normal-

ising the individual pulse distances inside each window by

$$\bar{\mathbf{D}}^{\jmath} := \frac{\hat{D}^{\jmath}}{\sum_{k=\jmath}^{\jmath+D-1} \delta_k}, \quad \jmath = 1, ..., \mathcal{J}. \tag{4.26}$$

Both features are describing the temporal image which depends on the teeth structure of the stridulatory file (see Figure 4.2) and the wing movement (see Section 4.1.3). In particular the teeth structure of katydids is highly species-specific [Hel88].

Pulse Lengths (L)

For the j-th onset λ_j and the j-th offset μ_j the pulse length of the j-th pulse is simply given by $\mu_j - \lambda_j$. The pulse length feature vector $\mathbf{L}^{\jmath} \in \mathbb{R}^D$, $\jmath = 1, ..., \mathcal{J}$ is calculated by the D-tuple coding scheme. It is given by

$$\mathbf{L}^{\jmath} := (\mathcal{O}^{\vartheta}_{p^{\mathbf{L}}_{20}}(\mu_{\jmath} - \lambda_{\jmath}), \mathcal{O}^{\vartheta}_{p^{\mathbf{L}}_{20}}(\mu_{\jmath+1} - \lambda_{\jmath+1}), ..., \mathcal{O}^{\vartheta}_{p^{\mathbf{L}}_{20}}(\mu_{\jmath+D-1} - \lambda_{\jmath+D-1})). \tag{4.27}$$

Again $\mathcal{O}^{\vartheta}_{p^{\mathbf{L}}_{20}}(\cdot)$ linearly normalises the pulse length to a environment temperature of 20 °C (see Section 4.1.6).

Sonograms

Sonograms may not provide the best input features for classifying with artificial neural networks or other statistical classifier schemes [MMR98b], because the number of features is too high. But they allow accurate measurement of the length and frequencies of notes and illustrate the way in which the frequency and amplitude of sound change in time [EE94]. Therefore, sonograms provide a good basis to extract features like frequencies and energy contours of a sound pattern, e.g. a syllable or a pulse.

We distinguish two types of sonograms: (1) sonograms with a *fixed time-resolution* and (2) sonograms with a *fixed number of sampling points*. Let $\bar{s}(t)$ be the signal which contains the sound of the animal. Then the amplitude signal of the j-th pulse is given by

$$u_j(t) = \begin{cases} \bar{s}(t), & t \in [\lambda_j, \mu_j] \\ 0, & \text{otherwise} \end{cases}. \tag{4.28}$$

For simplicity we set $u := u_j$, $\mu := \mu_j$ and $\lambda := \lambda_j$. Then the sonogram of a single pulse u is calculated by D short time Fourier transformations defined as

$$E_{vd} = \sum_{\tau=-\infty}^{+\infty} u(\tau) \mathcal{W}^{V,1}_{\mathbf{t}_d}(\tau) e^{-i\frac{2\pi}{V}\tau v}, \quad d = 1, ..., D \tag{4.29}$$

where $v = 0, ..., \frac{V}{2}$ is the frequency band [RJ93] and $\mathcal{W}_{\mathbf{t}_d}^{V,1}$ is a rectangular window of size V (see Eq. 4.7). The sampling vector $\mathbf{t} = (\mathbf{t}_1, ..., \mathbf{t}_D)$ is given by

$$\mathbf{t}_d \begin{cases} \lambda - (\frac{1}{2} - \alpha)V, & d = 1 \\ \mathbf{t}_1 + (d-1)\beta V, & \text{otherwise} \end{cases} \tag{4.30}$$

for $d = 1, ..., D$. Here the overlap between two consecutive sampling windows is $(1 - \beta)V$ and αV is the overlap between the first window $\mathcal{W}_{\mathbf{t}_1}^{V}$ and the time signal u (see Figure 4.18).

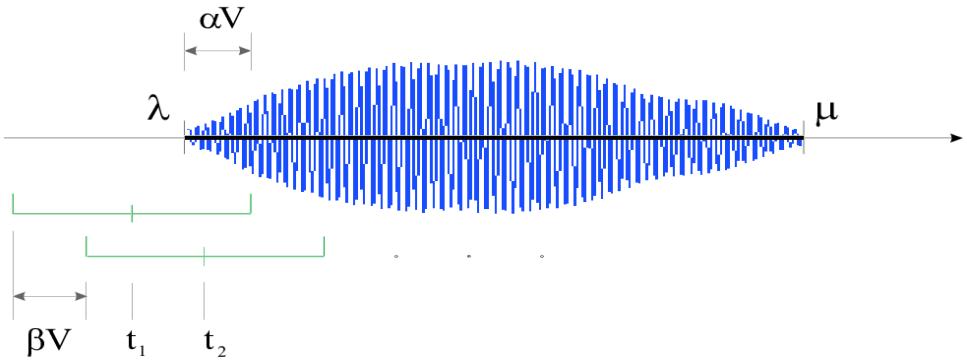


Figure 4.18: The first and the second sampling window to extract a sequence of frequency spectra from a single pulse.

If the parameters α, β and V are fixed, the length of the sampling vector $\mathbf{t}$ which determines the number of spectra depends on the length of the analysed signal part. It is given by

$$D = \left\lceil \frac{\mu - \lambda + (1-\alpha)V - \alpha V}{\beta V} \right\rceil . \tag{4.31}$$

This is the common type of sonogram which is used in many applications, e.g. speech recognition and bioacoustics [KM98].

However, sonograms with a constant number of spectra are of interest in order to calculate feature vectors of constant length D. These sonograms can be calculated by modifying the sampling vector $\mathbf{t}$ (see Eq. 4.30). For window size V and signal length $\mu - \lambda$, the window movement βV (see Figure 4.18) is given by

$$\beta = \frac{\mu - \lambda + (1 - 2\alpha)V}{(D-1)V}. \tag{4.32}$$

Both types of sonograms are utilised for the extraction of acoustic features (e.g. frequency contours and energy contours) and are determined by the parameters V, α and β (fixed time resolution) or V, α and D (fixed number of sampling points). In the context of time alignment it has to be mentioned, that sonograms with a fixed number of sampling points have been linearly normalised in time by calculating the window movement β for each pulse separately (see Eq. 4.32 and Section 2.2.1).

Pulse Frequency Contours (C)

The frequency contour (frequency-time course) of vocalisations is a signal parameter which is used in many systems for the classification of animal sounds [PJ00]. For instance, MURRAY et al. [MMR98b] extracted the frequency contour by *fundamental frequency analysis* in order to classify whistles of killer whales. One method to extract the frequency contour f from a discrete spectrum is to determine the frequency band with maximal energy [VHH98].

For a sonogram of a single pulse consisting of D spectra (see Eq 4.29), let ω_v be the mean frequency of the v-th frequency band and E_{vd} be the energy of the v-th frequency band within the d-th time window. Then

$$v_d^* = \operatorname*{argmax}_v(E_{vd}) \tag{4.33}$$

is the frequency band with the maximal energy and $\omega_{v_d^*}$ is the corresponding frequency. Including adjacent frequency bands and time windows is sharpening the result. Therefore, we do not use $\omega_{v_d^*}$, the frequency band with the maximal energy, but the weighted average

$$f_d := \sum_{v=v_d^*-\phi}^{v_d^*+\phi} \sum_{k=d-\sigma}^{d+\sigma} \frac{\omega_v E_{vk}}{\sum_{v=v_d^*-\phi}^{v_d^*+\phi} \sum_{k=d-\sigma}^{d+\sigma} E_{vk}}. \tag{4.34}$$

Here the parameters ϕ and σ determine the size of the averaging window in the frequency and the time domain.

The frequency contour of a single pulse $\mathbf{C} \in \mathbb{R}^D$ is then given by $\mathbf{C} = (f_1, ..., f_D)$. Now we assume a sequence of J pulses, each represented through a sonogram of fixed length D. Then we denote with $\mathbf{C}^j \in \mathbb{R}^D$ the frequency contour of the j-th pulse. The sequence of averaged frequency contours within $G \leq \Lambda$ adjacent pulses is then used as feature for the classification. Is is given through

$$\mathbf{C}^\jmath = \sum_{k=\jmath}^{\jmath+G-1} C^k \quad , \jmath = 1, ..., \mathcal{J}. \tag{4.35}$$

For the parameters used in the numerical evaluation see Table C.2 and Table C.3.

Pulse Frequencies (F)

Let $f^j \in \mathbb{R}^D$, $j = 1, ..., J$ be the frequency contour of the j-th pulse and $\bar{f}^j = \frac{1}{D}\sum_{d=1}^{D} f_d^j$ be the average frequency of this pulse. Then the feature vector $\mathbf{F}^\jmath \in \mathbb{R}^{D'}$, $\jmath = 1, ..., \mathcal{J}$ which is used as input for the classifier is given by the D'-tuple coding scheme

$$\mathbf{F}^\jmath := (\bar{f}^\jmath, \bar{f}^{\jmath+1}, ..., \bar{f}^{\jmath+D'-1}). \tag{4.36}$$

In contrast to the frequency contour (see Eq. 4.35) which contains the averaged frequency contour of G consecutive pulses, the pulse frequency (see Eq. 4.36) contains the D'-tuple coded pulse frequencies of D' consecutive pulses (see Figure 4.16).

Energy Contours of Pulses (E, $\hat{\mathbf{E}}$)

For the extraction of the energy contour the spectral energies (see Eq. 4.29) can be used.[6] In contrast to the frequency contour (see Eq. 4.35), the energy contour is extracted from sonograms with a fixed number of sampling points (type 1) and sonograms with a fixed time resolution (type 2). Then the total energy of the d-th spectrum E_d (see Eq. 4.29) is simply given by $E_d = \sum_{v=0}^{V/2} E_{vd}$.

Again we assume a sequence of J pulses, each represented through a sonogram where the number of spectra depends on the pulse length. Then we denote E_d^j as the energy of the d-th frequency band inside the j-th pulse and $E^j = (E_1^j, ..., E_{D_j}^j)$ as the energy course of this pulse, where D_j defines the number of spectra. For sonograms with a fixed number of sampling points

$$\hat{\mathbf{E}}^\jmath := \sum_{k=\jmath}^{\jmath+G-1} E^k, \quad \jmath = 1, ..., \mathcal{J} \tag{4.37}$$

may be used as a feature vector because $D_j = D$. For energy contours with a fixed time resolution of size D_j, we set

$$\bar{E}^j := \begin{cases} (E_1^j, ..., E_D^j), & D \leq D_j \\ (E_1^j, ..., E_{D_j}^j, 0, ..., 0), & D > D_j, \end{cases} \tag{4.38}$$

[6] Due to the theorem of the *spectral representation of energy*, the energy in the frequency domain is equal to the energy in the temporal domain [VHH98].

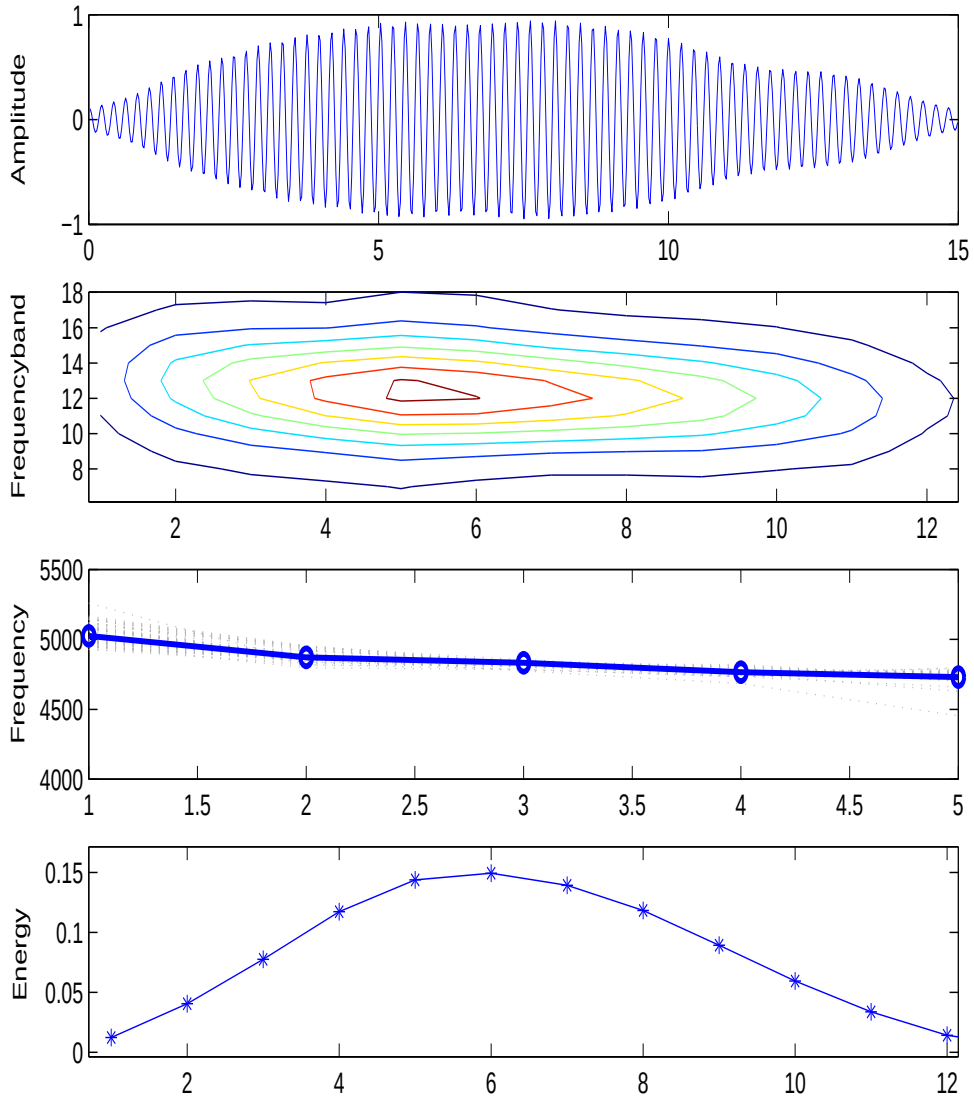


Figure 4.19: Typical features extracted from a single pulse of the species *Noctitrella plurilingua*: (a) time signal, (b) sonogram as contour plot, (c) pulse frequency contour and (d) energy course. Whereas for Figure (b) the x-axis corresponds to the respective of spectrum in Figure (c) and (d) the x-axis corresponds to the feature dimension.

where D is the desired number of spectra. The discrete feature vector $\mathbf{E}^j \in$

$\mathbb{R}^D$ is again given by the average of $G \leq \Lambda$ adjacent energy courses

$$\mathbf{E}^{\jmath} := \sum_{k=\jmath}^{\jmath+G-1} \bar{E}^k, \quad \jmath = 1, ..., \mathcal{J}. \tag{4.39}$$

A typical energy course of the species *Noctitrella plurilingua* is shown in Figure 4.19(d).

Time Encoded Signals of Pulses (T)

Time encoded signals (TES) is a simplified signal description according to general sampling theory first represented by KING and GOSLING [KG78] on coding speech for transmission. Later this method was applied highly effective in speaker verification applications and bioacoustics [CFS98]. The TES signals are based on a precise mathematical description of waveforms, involving polynomial theory that shows how a bandlimited signal may be completely described in terms of the locations of its zeros and further signal parameters [KP98].

This means, that in TES the signal is segmented into epochs and local signal parameters are extracted within these epochs. For this stream of signal parameters, fixed-size histograms which present specific signal properties are calculated.

Figure 4.20 exemplifies the signal segmentation into a stream of epochs. An epoch (positive epoch or negative epoch) is the region between two adjacent zero-crossings, for each epoch a set of parameters is extracted. Examples for such parameters are: The *duration* (number of samples spanned), the *shape* (number of significant positive minima or negative maxima) and the *magnitude* (e.g. the minimal or maximal sample magnitude encountered).

Often these parameters are more economically mapped non-linearly onto a smaller set of numerical descriptors called *symbol alphabet*[7] or these parameters are converted into a sequence of histogram matrices. Hereby the parameters extracted from the individual epochs are application dependent and therefore, it follows that these parameters have a significant influence to the discrimination power.

For the signal $(\bar{s}(t))_{t=1}^{T}$, a sequence of signal parameters $\mathbf{v} = (\mathbf{v}^m)_{m=1}^{M}$ with $\mathbf{v}^m = (\mathbf{v}_1^m, ..., \mathbf{v}_Q^m)$ is calculated from each of the individual epochs. Hereby M is the number of epochs and Q is the number of parameters extracted within each of the individual epochs. We define a Q-dimensional

[7] A detailed description of the coding process is contained in [HHK86].

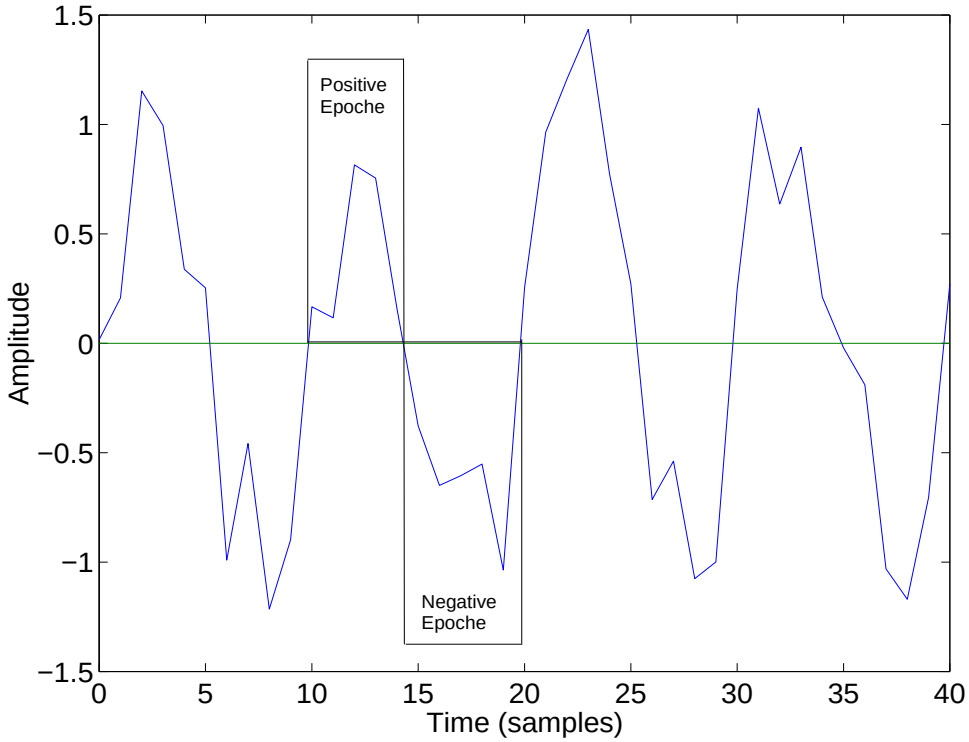


Figure 4.20: In the TES coding process the signal is segmented into a sequence of negative and positive epochs. A set of local signal parameters is extracted within each of these epochs.

histogram by considering a sequence of ranges $R = (\mathbf{r}_1, ..., \mathbf{r}_Q)$ where $\mathbf{r}_q \in \mathbb{R}^{\Phi_q}$ contains the ranges of the individual bins for the q-th signal parameter. For example, if in q-th feature dimension 5 linearly spaced bins in the range of 0.0 and 1.0 should be determined Φ_q is set to 6 and $\mathbf{r}_q = (0.0, 0.2, 0.4, 0.6, 0.8, 1.0)$. The *histogram matrix* $H \in \mathbb{R}^{\Phi_1 \times, ..., \times \Phi_Q}$ is then given by

$$H_{\kappa_1,...,\kappa_Q} = \sum_{m=1}^{M} \begin{cases} 1, & \mathbf{r}_{1,\kappa_1-1} \leq \mathbf{v}_1^m < \mathbf{r}_{1,\kappa_1}, ..., \mathbf{r}_{Q,\kappa_Q-1} \leq \mathbf{v}_Q^m < \mathbf{r}_{Q,\kappa_Q} \\ 0, & \text{otherwise} \end{cases} \tag{4.40}$$

for each $\kappa_1 = 2, ..., \Phi_1, ..., \kappa_Q = 2, ..., \Phi_Q$.

For the feature extraction the TES histogram matrices (see Eq. 4.40) are derived within G adjacent pulses. Let H^j be a histogram matrix extracted from the j-th pulse, then the averaged histogram matrix covering $G \leq \Lambda$

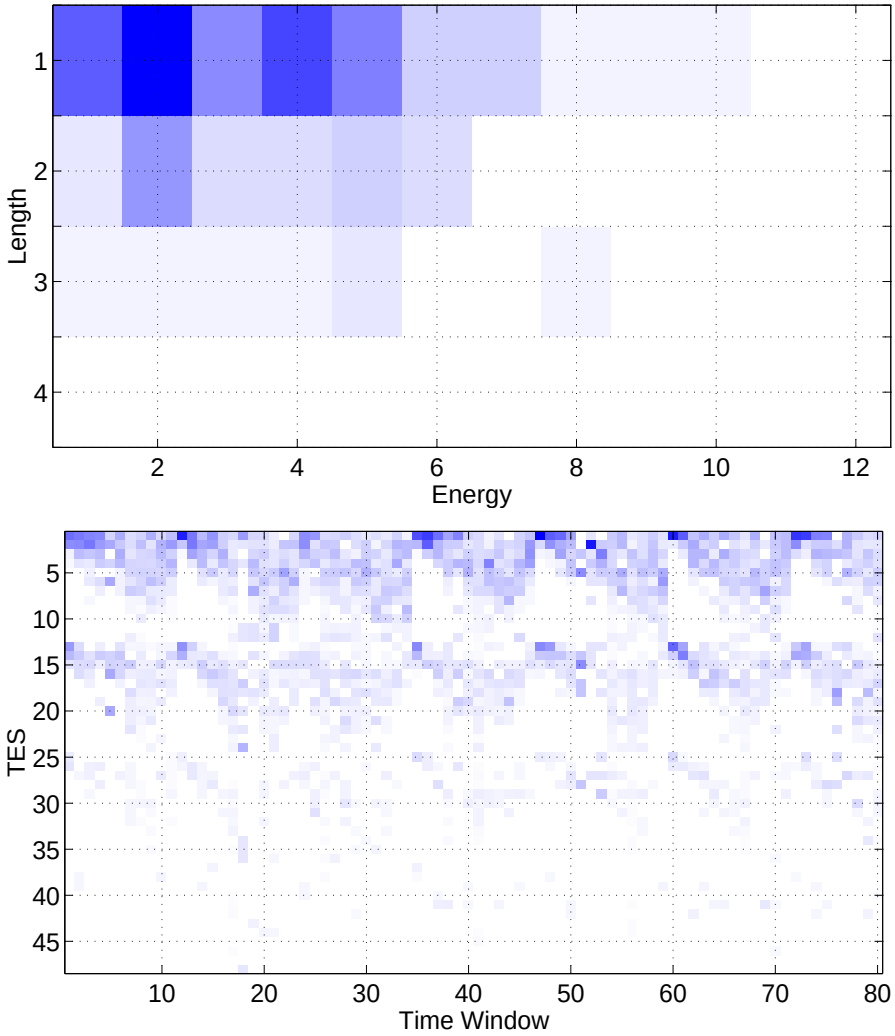


Figure 4.21: Feature extraction by using the TES-matrices of the katydid species *Eupholidoptera chabrieri*: (a) A single histogram matrix $H \in \mathbb{R}^{4\times 12}$ and (b) the TES feature stream from $\mathbf{T}^{\jmath} \in \mathbb{R}^{48}$, $\jmath = 1, ..., \mathcal{J}$ for $\mathcal{J} = 80$ time windows.

consecutive pulses $\hat{H}^{\jmath}$ is defined through

$$\hat{H}^{\jmath} := \frac{1}{G} \sum_{k=\jmath}^{\jmath+G-1} H^{k}, \quad \jmath = 1, ..., \mathcal{J}. \tag{4.41}$$

To extract feature vectors the averaged TES matrices are linearised. This leads to a sequence of $\mathcal{J}$ feature vectors $\mathbf{T}^{\jmath} \in \mathbb{R}^{D}$ with $D = \Phi_1 \cdot, ..., \cdot \Phi_Q$.

In our numerical experiments we calculated two-dimensional ($Q = 2$) histogram matrices by using the *epoch length* and the *energy within an epoch* as signal parameters. Figure 4.21 shows a typical TES matrix extracted within a single window and the feature stream containing the linearised TES matrices of a sequence of 80 time windows. Each feature vector $\mathbf{T}^j$ describes the energy and frequency composition of the acoustic signals.

Well evaluated and hardware-implemented on an derivative of the 8-bit Intel 8051 microcontroller which performs robust speech recognition is the combination of time encoded signal processing and artificial neural networks (TESPAR). Recent practical experiences confirmed that TESPAR enables the implementation of very powerful classification procedures [KP98].

Maximal Amplitude of Pulses (A)

Let $u_j(t)$ be the amplitude signal of the j-th pulse as defined in Eq. 4.28. Then the *maximal amplitude* of this pulse is given by $a_j := \max_t |u_j(t)|$. In order to derive feature vectors of the $\mathbb{R}^D$ the D-tuple coding rule is applied

$$\mathbf{A}^j = (a_j, a_{j+1}, ..., a_{j+D-1}), \quad j = 1, ..., \mathcal{J}. \tag{4.42}$$

Figure 4.22 shows a one-dimensional maximal amplitude contour for four

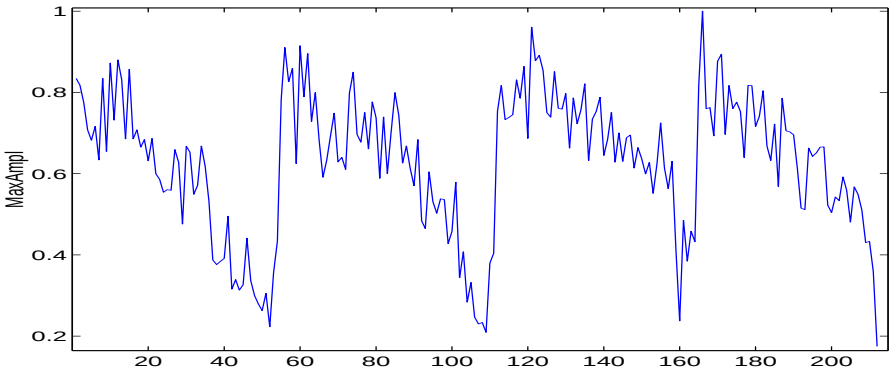


Figure 4.22: The one-dimensional maximal amplitude contour for four syllables of the katydid species *Isophya modestior*.

syllables of the species *Isophya modestior*. Hereby it was observed that the course of the individual syllables shows a similar structure. The corresponding amplitude signal is given in Figure 4.24(a).

4.2.6 Global Features

In [DSRP01] we studied the combination of global and local features for the classification of crickets to species level. Here we found, that in particular the combination of local features is very promising. A combination of global and local features through classifier fusion did not significantly improved the classification performance. But in these studies the *Parzen density* of pulse distances ($\mathbf{P}$) emerged as an important feature which has been overlooked by traditional neurobiologists [Gla96]. Furthermore, these density functions are crucial, particularly for the description of the acoustic signals.

To determine a probability density function over the distances between two consecutive pulses a histogram procedure can be used [Par62]. The basis to extract these classification vectors are the distances between the j-th and the $(j+1)$-th pulse δ_j, $j = 1, ..., J-1$ (see Eq. 4.23). Based on these distances a one-dimensional density function $\mathbf{P}_\delta$ with the Gaussian density as kernel function [KB98b] and variance $\sigma^2 > 0$ is estimated. It is given by

$$\mathbf{P}_\delta(t) := \frac{1}{(J-1)\sigma\sqrt{2\pi}} \sum_{j=1}^{J-1} \exp\Big(-\frac{(t-\delta_j)^2}{2\sigma^2}\Big). \tag{4.43}$$

Hereby the Gaussian probability density function is used as kernel function under the assumption that the estimated density function $\mathbf{P}_\delta$ is continuous [Rus88]. To approximate a discrete feature vector $\mathbf{P} \in \mathbb{R}^D$, the function $\mathbf{P}_\delta(t)$ is sampled with linear increasing time steps $\mathbf{t} = (\mathbf{t}_1, ..., \mathbf{t}_D)$ where $\mathbf{t}_d$ is defined through

$$\mathbf{t}_d := \begin{cases} 0, & d = 1 \\ \gamma \mathbf{t}_{d-1} + \phi, & \text{otherwise} \end{cases}. \tag{4.44}$$

Hereby D is determined in such a way that $\mathbf{t}_D$ is approximately 500 ms depending on the parameters of γ and ϕ. A typical probability density function of the species *Noctitrella glabra* is depicted in Figure 4.23. The appendant amplitude signal is given in Figure 4.13.

Classification results, confusion matrices for the individual features and more details about fusion of local and global features can be found in [DSRP01].

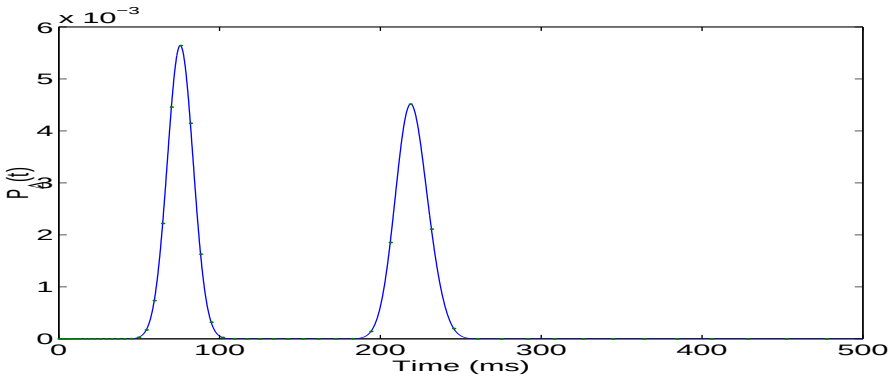


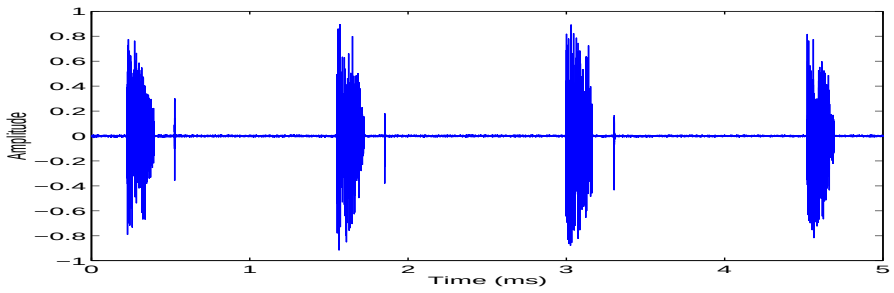
Figure 4.23: The probability density function of the pulse distances of the species *Noctitrella glabra* shows clusters of pulse distances with centers at 74 ms and 220 ms.

4.2.7 Feature Extraction in Time Scales

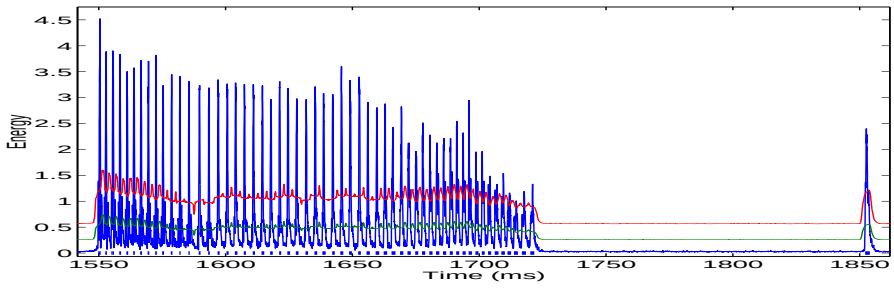
All local features given in Section 4.2.5 depend on onset and the offset positions calculated by algorithm SEGMENTATION. Due to the fact that these positions can be extracted in different time scales, each local feature can be grouped into the time scale, in which it has been extracted. This is particularly important for slowly singing katydid species, producing easily detectable impulses and syllables (see Section 4.1.2 and Figure 4.6). The knowledge about typical impulse distances and impulse length and typical syllable distances and syllable length within the katydid family allows to determine the positions of impulses and syllables with algorithm SEGMENTATION by using specified parameter settings for the pulse segmentation algorithm. In order to extract features in four time scales TS1, ..., TS4, each time scale is defined by the window size U to calculate the short time energy (see Eq. 4.18), two window sizes V and W, a set of parameters to calculate the threshold function (see Eq. 4.19) and the number of pulses Λ covered by the short time window W^J (see Figure 4.16).[8] Hereby the time scales TS1 and TS2 differ mainly regarding the number of pulses covered by the local time window W^J.

An example for the pulse detection in two time scales is given in Figure 4.24. Whereas Figure 4.24(b) shows the detection of the individual impulse positions (TS1), Figure 4.24(c) shows the detection of syllable positions (TS4).

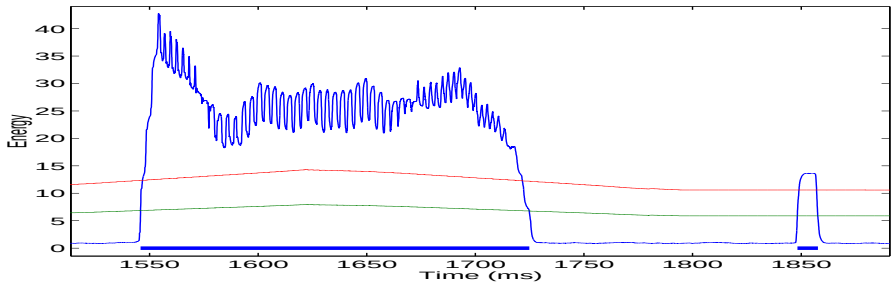
[8] Parameter settings for the pulse segmentation: crickets (one time scale TS1, see Table C.2) and katydids (four time scales TS1, ..., TS4, see Table C.3).



(a) The waveform $\tilde{s}(t)$



(b) Impulse detection (TS1)



(c) Syllable detection (TS4)

Figure 4.24: A short song of the katydid species *Isophya modestior* (signal length 5000 ms), see Figure (a). In Figure (b) the impulse segmentation by algorithm SEGMENTATION is shown. Figure (c) shows the segmentation of the individual syllables. Onset and offset positions of the detected signal segments are indicated by the bars at the bottom.

4.3 Data Set Description

The data set used in the numerical experiments in this thesis consists of 386 sound patterns of 53 different *Orthoptera* species. For each species it contains up to 12 recordings of different individuals. Whereas 185 of these recordings are cricket songs from 31 different species (see Table D.2) 201 recordings are katydid songs from 22 different species (see Table D.1). For each song the environmental temperature has been measured and the songs have been labelled with a specimen code which allows to refer to the voucher specimen. Additionally, the specimen codes have been used to guarantee that the individual songs of a certain species are always recordings from different individuals.

The data set is part of the DORSA database containing 7432 *Orthoptera* songs (1057 cricket songs from 96 different species and 6375 katydid songs from 336 different species). All sounds were recorded with an analog or digital tape recorder and were sampled via an analog-digital converter to be analysed with a signal processing software. The files were converted to the standard WAV-format with a sampling accuracy of 16 Bit and different sampling frequencies in the range of 44.100 Hz to 500.000 Hz. The sound files were provided by:

- Humboldt-University Berlin (110 recordings),
- K.-G. HELLER (5873 recordings),
- G. H. SCHMIDT (360 recordings),
- F. NISCHK (458 recordings) and
- S. INGRISCH (631 recordings).

The recordings from K.-G. HELLER [Hel88] are mainly recorded in Europe (1978-1999), using the following tape recorders: Sony WM-D3, Uher 4200IC and Racal, and the microphones : Sony ECM-121, Uher M516, Uher M645, B&K 1/2", B&K 1/4"and the Bat-detector. Sound sampling was performed with Amadeus on a Mac-computer and a custom-built DSP card by using the program: DSP-Control (version 2.3.0.0) by T. JAUMANN, Erlangen with sampling rates of 250 and 500 kHz. After digitising the sampling rate has been reduced to five different sampling rates in the range of 44.100 Hz to 500.000 Hz.

SCHMIDT used the Uher 4000 IC tape recorder and the microphones: Uher M 53 and the Bat-detector for his recordings.

Cricket songs provided by F. NISCHK have been recorded in Ecuador (1995-1999) for the experiments of his doctoral thesis [Nis99]. He used a portable DAT-recorder (Sony TCD-D7) and two Sennheiser microphones: ME 67 and MKE 2-60 with pre-amplifier K6 for the sound recordings. Sound sampling was performed using Soundscope 8 (44.100 Hz sampling frequency).

The cricket songs from Thailand were provided by INGRISCH [IK98] (1986-1997) utilising one of the following tape recorders: KX-5010, Sony TCD-D7, Sony WM-D3 and the Aiwa DAT recorder. He used the microphones: Aiwa, Sennheiser and the Black Fire 541. The sounds have been sampled with Soundscope with a sampling frequency of 50.000 Hz. By resampling the sampling rate has been reduced to the standard sampling rate of compact discs, 44.100 Hz.

4.4 Feature Evaluation

This Section deals with the evaluation of single features by utilising a histogram method which we call *rank distribution*. The method works for time series as follows:

Let $(\mathbf{x}_i(j))_{j=1}^{\mathcal{J}}$ with $\mathbf{x}_i \in \mathbb{R}^{D_i}$ be a feature stream from the i-th feature space to be evaluated. The final classification of this time series is then calculated through a temporal fusion mapping (see Section 3.4) which combines the local class decisions over the whole time series

$$\mathbf{z} := \mathcal{F}(\mathcal{C}(\mathbf{x}_i(1)), ..., \mathcal{C}(\mathbf{x}_i(\mathcal{J}))). \tag{4.45}$$

The final decision $\mathbf{z} \in \mathbb{R}^L$ of this time series is then sorted in decreasing order, e.g. there is a sequence $(\tau_l)_{l=1}^L$ of indices such that $\mathbf{z}_{\tau_1} \geq ... \geq \mathbf{z}_{\tau_L}$. We now assume to know the true class label $\omega \in \Omega$ of the final decision. With $R \in \Delta \cap \{0,1\}^L$ we define a binary unit vector containing a single 1 at the rank of the true class label ω and $L-1$ zeros , e.g.

$$R_l = \begin{cases} 1, & \tau_l = \omega \\ 0, & otherwise \end{cases}, \quad l = 1, ..., L. \tag{4.46}$$

Hereby $R = (1, 0, ..., 0)$ indicates, that the whole time series was classified correctly. The rank distribution $R^o \in \Delta$ (see Eq. 3.5) for the whole data set including the individual time series is calculated by the $\tilde{k}$-fold cross validation procedure (see Section 3.7). Hereby for each of the $\tilde{k}$ cross validation cycles the rank distributions of the time series in the test set (see Eq. 4.46) are averaged. In order to calculate R^o, the center of gravity of the

averaged rank distributions of the individual cross validation tests is computed. The classification accuracy of the feature stream $\mathbf{x}_i$ is then given by the value of R_1^o.

In contrast to the classification accuracy for $L > 2$ this method offers the advantage to estimate the degree of misclassification by considering the ranks of the individual decisions. This is particularly important if multiple classifiers are combined with decision fusion. In this case the misclassifications that appear at lower ranks often enhance the overall performance of the classifier ensemble. But actually, misclassifications appearing at the higher ranks usually decrease the performance of the classifier system.

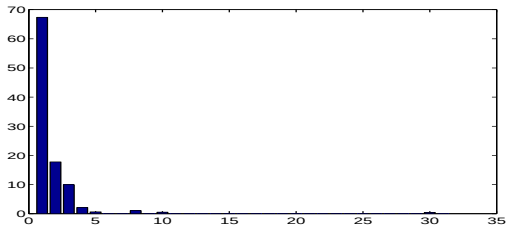
To evaluate the individual features in both hierarchy levels, the feature evaluation was applied in the first level and in the second level of the hierarchy separately (see algorithm HOC). Hereby we calculated the error rates in the first level of the hierarchy, because for $L = 2$ the rank distribution offers no advantages toward simple error rates (see Table 4.1). The best classification result based on a single feature is achieved with the

feature	error in %
filter-bank energies ($\mathbf{B}$)	4.66
pulse distances ($\mathbf{D}$)	18.65
pulse length ($\mathbf{L}$)	32.64

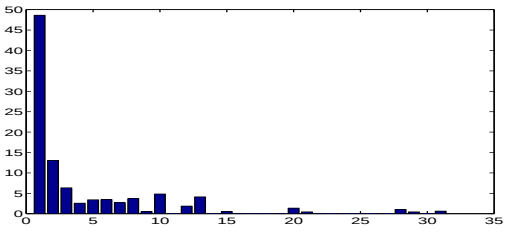
Table 4.1: Error rates of the first level of the hierarchy for each of the individual features. The temporal integration (see Eq. 4.45) was performed with averaging.

filter-bank energies (**B**), proposed in Section 4.2.2.

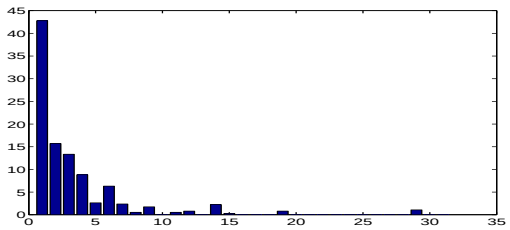
In the second level of the hierarchy we applied the rank distribution for each family and each time scale separately (crickets: see Figure 4.25 and katydids: see Figure A.1 - A.4). Here in each Figure it can be observed that the pulse distances (**D**), usually lead to the best classification results. The only exception are the normalised pulse distances $\bar{\mathbf{D}}$ is TS2 (see Figure A.2) where the pulse distances **D** and $\bar{\mathbf{D}}$ are derivated inside a sliding window covering 21 consecutive pulses to extract 20 pulse distances (see Eq. 4.25 and Eq. 4.26). For this time scale $\bar{\mathbf{D}}$ seems to be more discriminative because the normalised pulse distances ($\bar{\mathbf{D}}$) describe the pattern of the tooth spacings of the *pars stridens* (see Figure 4.2) which contains typically 60 to 90 teeth (see Section 4.1.2). Consequently, the classification performance of the normalised pulse distances ($\bar{\mathbf{D}}$) in TS2 is better than the classification



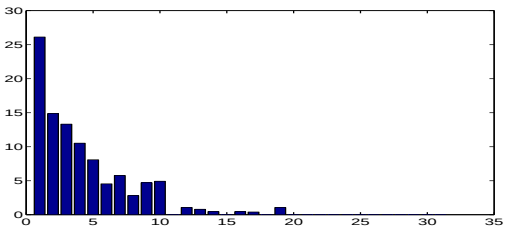
(a) Pulse distances (**D**)



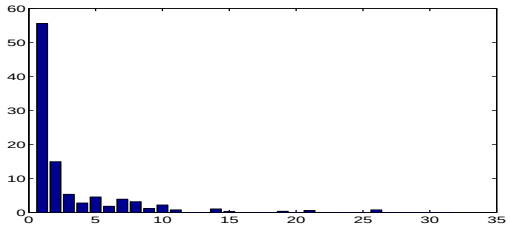
(b) Pulse distances ($\bar{\mathbf{D}}$)



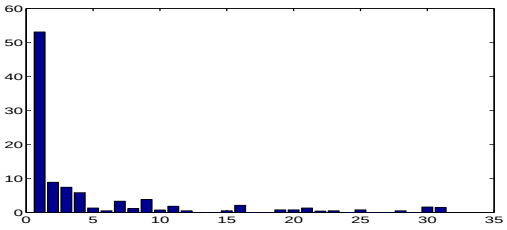
(c) Pulse length (**L**)



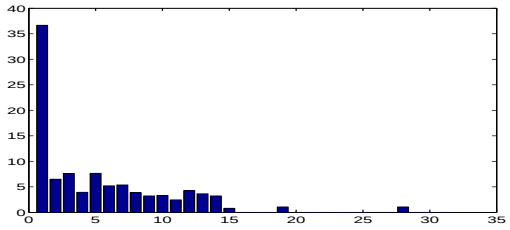
(d) Pulse frequency (**F**)



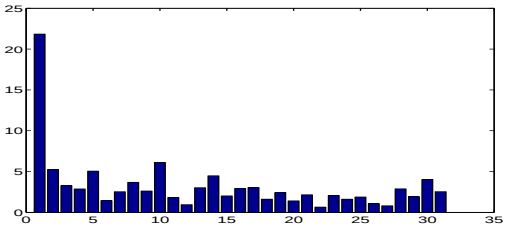
(e) Pulse energy contour (**E**)



(f) Pulse energy contour ($\hat{\mathbf{E}}$)



(g) TES of pulses (**T**)



(h) Maximal amplitude of pulses (**A**)

Figure 4.25: Evaluation of features extracted from 5 consecutive pulses of cricket species. Whereas the first bin of the x-axis shows the classification accuracy, the other bins show the classification rate for the individual ranks. The number of ranks is given by the number of classes.

performances of all evaluated features presented in Figure A.1 - A.4.

On that score, we infer, that a certain feature is not sufficient to separate the vocalisations of individuals from both families (see Table 4.1). As well, a certain feature is not sufficient for the classification to species level (see Figure A.1 - A.4) [Jat95]. One good reason for that is, that the significance of a certain feature is highly dependent of the species itself. For example, the pulse distances (**D**) and the normalised pulse distances ($\bar{\mathbf{D}}$) are generally more discriminative for species producing *quasi-deterministically oscillating time series* (see Figure 4.17) [BD87]. In this case the distances between

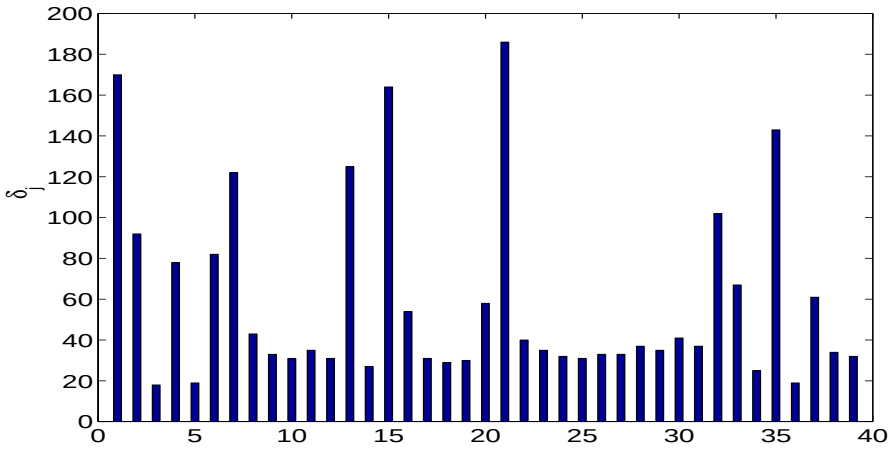


Figure 4.26: Pulse distances of a 2.5 sec. sound signal of the species *Zvenella geniculata*.

two consecutive pulses δ_j (see Eq. 4.23) lead to clusters of feature vectors by using the $D-$tuple coding scheme to produce features of the $\mathbb{R}^D$. On the other hand, some species produce *stochastically oscillating time series*, where the future evolution cannot be determined. One prominent species is the *Zvenella geniculata*, generating sounds in which the pulse distances create straggled prototypes all over the feature space if these distances are coded with the D-tuple coding scheme (see Figure 4.26). Since the pulse distances of this species is highly stochastic, this feature leads to misclassifications.

In our numerical experiments we found that for most species of the *Orthoptera* the oscillation of pulse distances (**D**) is quasi-deterministic, because the condition of strong periodicity $\delta_{j+m} = \delta_j$, where m is the period of the oscillation, is never completely fulfilled. Nevertheless, in behavioural experiments it was observed, that species of the *Orthoptera* are using a set of acoustic features to recognise songs of the own species (see Section 4.1).

4.5 Feature Selection

In contrast to feature evaluation by the rank distribution (see Section 4.4) in this Section combinations of feature components are evaluated by a feature selection algorithm. This allows to take the influence of the correlations among the individual feature components to the classification performance into account. Due to the fact that optimal methods, e.g. *exhaustive search* are not suitable for high-dimensional problems [CC77], many suboptimal methods have been developed. One of the most popular wrapper selection method [Hal99] is the *sequential forward selection* (SFS) algorithm.

In SFS a single classifier is utilised to classify a labelled data set. Then for each feature component a *criterion function* (e.g. the classification accuracy) is calculated by applying the classifier to each feature component of the data set, in order to select the best feature component (the feature component which maximises the criterion function). This step is repeated by consecutively adding a single feature component to the current feature set in such a way, that the criterion function is maximised. Hence SFS is step-optimal, since for each optimisation step the best feature component is added to current feature set [PNK95].

Whereas in SFS one feature component is consecutively selected to increase the criterion function of a single classifier system, we proposed a modified version of the SFS algorithm to determine feature sets for multiple classifiers called SFSM [RBF+03]. In SFSM we suppose a set of classifiers, each classifier is associated with its corresponding feature set, and in each optimisation step a single feature component together with a classifier is selected. This means, that in each optimisation step the selected feature component is added to the feature set of the selected classifier in such a way, that the criterion function is maximised (a discussion and a detailed description of the SFSM algorithm can be found in [RBF+03]).

To demonstrate the classification performance of the selected feature components we depicted the selected feature components and the corresponding classification accuracy dependent of the number of selected feature components for the first (see Figure 4.27) and the second hierarchy level (crickets: Figure 4.28 and katydids: Figure B.1 to Figure B.4) separately. Each Figure shows the classification error of the selected feature components for each step of the feature selection procedure. Different brightness values have been assigned to the individual feature components to associate the feature components to a specific classifier.

In Figure 4.27 the selected feature components and the corresponding classification error by using the SFSM algorithm with three features of the first family level (see hierarchy level 1) is given. Hereby for the filter-

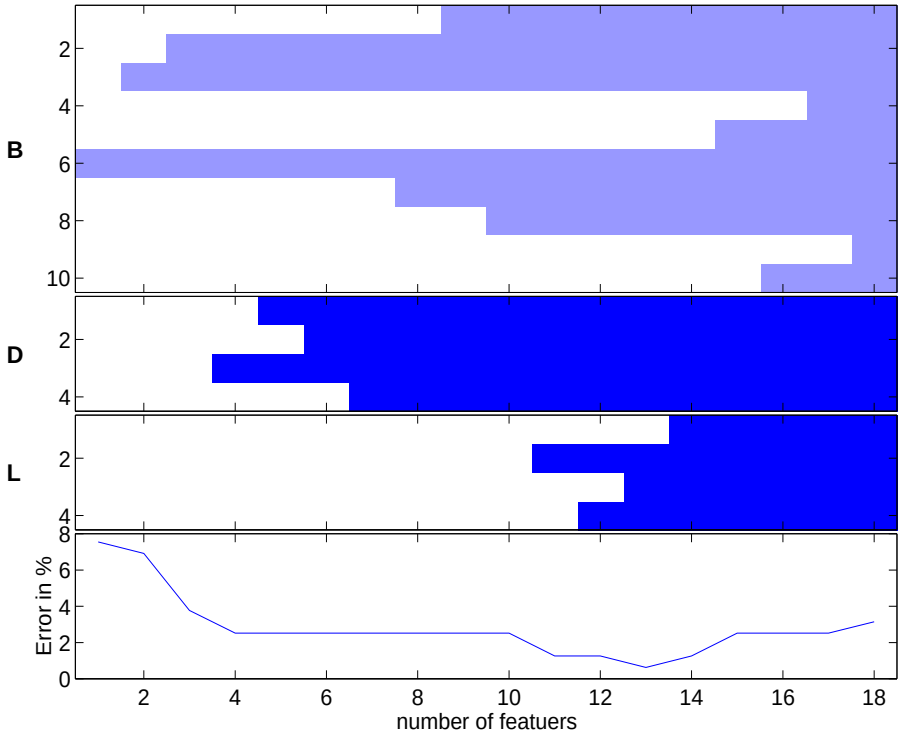


Figure 4.27: Feature selection in the family level with SFSM. The Figure shows the classification error and the selected feature components of the features **B**, **D** and **L**, for each step of the feature selection procedure. Two different brightness values have used, each of them is assigned to a specific classifier. The best classification result during the feature selection procedure have been obtained for 13 automatically selected feature components.

bank energies (**B**), see Section 4.2.2 the temporal resolution is different to the temporal resolution of the local features (the pulse distances (**D**) and the pulse lengths (**L**), see Section 4.2.5) and therefore, the number of feature vectors of the filter-bank energies and the local features is different. Thus, the filter-bank energies (**B**) define a separate time scale. Another time scale is defined by the two local features extracted within a sequence of pulses; the pulse distances (**D**) and the pulse length (**L**). To determine feature components of two time scales, the FCT architecture has been used for each time scale for the selection of feature components and the classification of the temporal sequences of the first level. Hereby the feature vector **B** is firmly associated with the classifier $\mathcal{C}^1$ and the features **D** and **L** are steadily associated with the classifier $\mathcal{C}^2$. The combination of these

two time scales is then accomplished by averaging the class membership estimates of the individual time scales (see algorithm HOC, step (e)).

The most discriminative features are the filter-bank energies (see Section 4.2.2) $\mathbf{B}_6$, $\mathbf{B}_3$ and $\mathbf{B}_2$ with center frequencies of 11.888 Hz, 6555 Hz and 4.777 Hz. Whereas the largest center frequency (11.888 Hz) is mainly in the frequency range of katydid species, the two lower center frequencies are in the frequency range of crickets. Although the filter-bank energies $\mathbf{B}_9$, ..., $\mathbf{B}_{10}$ correspond to center frequencies in the range of 17.222 Hz to 19.000 Hz, these features seem not to be very important to separate species of the two families. Two reasons for this are, that the airbone sound in higher frequency ranges is sensitive against influences discussed in Section 4.1.2, and that different microphone types have been used for the sound recordings (see Section 4.3).

For both families (crickets: see Figure 4.28 and katydids: see Figure B.1 to Figure B.4) the selected feature components and the corresponding classification error have been calculated separately. Three brightness levels are used to assign the individual feature components to three classifiers $\mathcal{C}^1$, $\mathcal{C}^2$ and $\mathcal{C}^3$. These classifiers are used to calculate sequences of local decisions that are combined with the CFT fusion architecture. Whereas for crickets the SFSM algorithm was applied, for katydids we applied the SFSM algorithm by alternating through four time scales TS1, ..., TS4. This means, that for each time scale one feature component together with a single classifier is alternately selected in such a way, that the overall classification error that depends on feature components of all four time scales is minimised. Therefore, the selected feature components (see Figure B.1 to Figure B.4) are assigned to the individual time scales. The evaluation of feature components based on each of the individual time scales is not considered here, because the correlations of the individual features through each of the four time scales are important and have to be taken into account.

Feature components of the pulse distances ($\mathbf{D}$) and the normalised pulse distances ($\bar{\mathbf{D}}$) have been selected for both families at the beginning. For crickets the pulse distance features have been combined with feature components of the pulse length ($\mathbf{L}$) and the pulse frequencies ($\mathbf{F}$) before components of other features[9]. By selecting further feature components from the energy contours ($\mathbf{E}$ and $\hat{\mathbf{E}}$) and the time encoded signals ($\mathbf{T}$), the criterion function was anymore increased (see the decreasing error rate in Figure 4.28).

Similar features have been selected for katydid species. Obviously, the pulse distances ($\mathbf{D}$) seem to be most discriminative for time scale TS1. For

[9] See the six primary selected features in Figure 4.28.

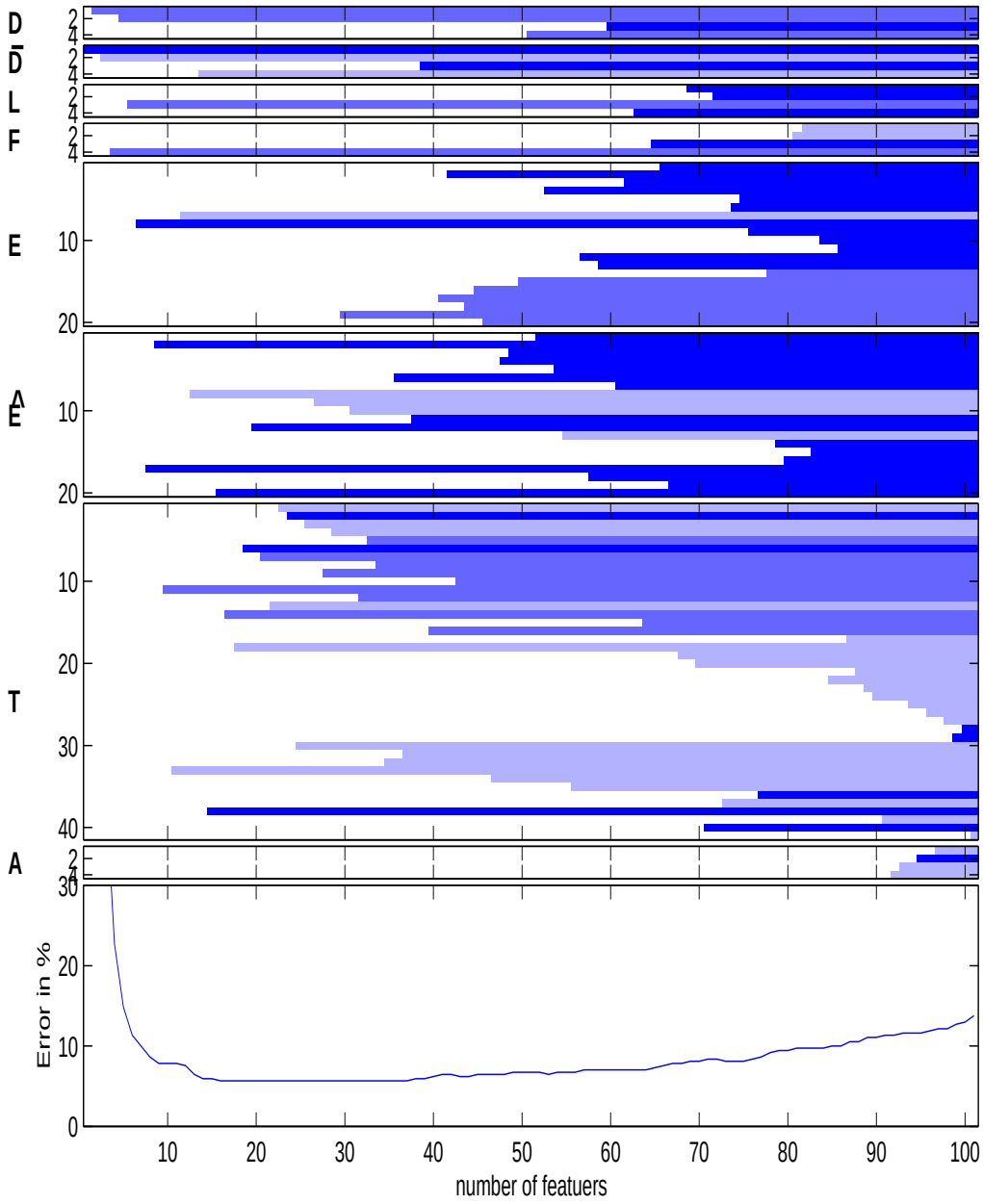


Figure 4.28: Feature selection for crickets to species level with SFSM. The Figure shows the classification error and the selected feature components of the features **D**, $\bar{\mathbf{D}}$, **L**, **E**, $\hat{\mathbf{E}}$, **T** and **A**, for each step of the feature selection procedure. Three different brightness values have used, each of them is assigned to a specific classifier. The best classification results during the feature selection procedure have been obtained for 18 to 37 automatically selected feature components.

TS2 and TS3 the normalised pulse distances ($\bar{\mathbf{D}}$) seem to be more discriminative, since they have been selected before components of other features. In contrast to crickets, for katydids the pulse frequency (**F**) is not as discriminative due to the wide spectra (see Figure 4.5 and Figure B.1 to Figure B.4).

Finally it has to be concluded, that for both data sets the SFSM algorithm has selected components of the features **E**, $\hat{\mathbf{E}}$, **T** and **A** rather late. But then, in several cases, a single feature has been selected consecutively by the same classifier.

4.6 Experiments and Results

Because of the limited data sets (see Section 4.3) we use the $\tilde{k}$-fold cross validation test (see Section 3.7) with $\tilde{k}$ = 12 to test the performance of the classifier system described by algorithm HOC. Five cross validation tests have been made and the classification results of all cross validation tests have been averaged. Hereby the focus is to evaluate the performance of the classifier fusion schemes presented in Chapter 3. Classification results are given for fuzzy-K-nearest neighbour classifiers (see Section 2.1.3). This classifier exhibits a large similarity to artificial neural networks, in particular to competitive neural networks and radial basis function networks (see Section 2.1.1 and [PG90]). Furthermore, it does not need any time-consuming adaption of the parameters.

Single Features / All Features

In order to get an overview of the discrimination power of single features and combinations of all features in Table 4.2 classification results for single features and all features are given for both *Orthoptera* families. For the classification of the individual feature streams we applied the FCT fusion scheme with averaging for the temporal integration (see Eq. 3.17). Additionally, for katydid species the error rates based on the individual features are given for each of the four time scales. The combination of the individual time scales is accomplished by averaging (AVR) and the probabilistic function (PF), see algorithm HOC step (e') and Section 3.4.2.

Not surprisingly, the classification performances of single features **D**, $\bar{\mathbf{D}}$, **L**, **F**, **E**, $\hat{\mathbf{E}}$, **T** and **A** are rather weak (crickets: 36.11 % to 81.08 % and katydids: 37.69 % to 84.44 %). By combining the features with data fusion (FCT architecture) the error rate decreased to 31.46 % for crickets and

family	crickets	katydids					
		TS1	TS2	TS3	TS4	AVR	PF
pulse distances ($\mathbf{D}$)	36.11	52.44	61.19	37.69	55.66	37.02	31.94
pulse distances ($\bar{\mathbf{D}}$)	51.24	68.86	63.88	45.42	65.56	34.33	33.73
pulse length ($\mathbf{L}$)	60.87	81.99	81.69	55.87	71.41	49.65	50.34
pulse frequencies ($\mathbf{F}$)	73.62	83.79	82.79	84.32	84.44	80.60	81.29
energy contours ($\mathbf{E}$)	60.11	78.41	77.91	61.61	63.74	52.44	52.33
energy contours ($\hat{\mathbf{E}}$)	66.70	73.53	73.43	63.02	56.87	50.35	48.06
time encoded signals ($\mathbf{T}$)	77.73	67.76	67.46	73.27	72.02	61.69	60.50
maximal amplitude of pulses ($\mathbf{A}$)	81.08	80.29	80.10	71.96	77.37	66.97	67.07
FCT	31.46	62.79	61.49	45.33	49.50	37.41	34.73
CFT	34.38	56.31	50.65	33.37	40.10	31.34	32.64

Table 4.2: Error rates for crickets and katydids for single features (upper part) and combinations of all features (bottom part). The combination of the individual features was applied by the FCT and the CFT architecture. For katydids the error rates of the individual time scales TS1, ..., TS4 and the error rates of the combination of all time scales are given. The combination of the individual time scales is applied with averaging (AVR) and the probabilistic function (PF).

37.41 % for katydids. The combination of all features with the CFT architecture leads to similar results (crickets: 34.38 % and katydids: 31.34 %) which is not satisfying, since the error rates for the pulse distances ($\mathbf{D}$) alone are not much higher (crickets: 36.11 % and katydids: 37.06 %). But for katydid species the combination of the four time scales with averaging improves the classification performance of the individual features and combinations of all features (see column AVR in Table 4.2). Particularly promising are the classification results if the class membership estimates of all time scales are combined with the probabilistic function (see column PF in Table 4.2).

Manually Selected Features

In the following Section we present classification results for manually selected features. Hereby for species of both families the selected features are biologically motivated [Jat95, Nis99]. For katydid species these features have to be determined for each of the individual time scales. Therefore, we also use the energy contours ($\mathbf{E}$, $\hat{\mathbf{E}}$) and the time encoded signals

(**T**) as classifier input, because these features alone lead to low error rates (see Table 4.2). Three features for each time scale have been selected for both families (see Table 4.3). An overview of the examined features can be found in Section 4.2.

family	time scale	features
crickets	TS1	**D**, **L** and **F**
katydids	TS1	**E**, $\hat{\mathbf{E}}$ and **T**
	TS2	**D**, $\bar{\mathbf{D}}$ and **L**
	TS3	**D**, **L** and **A**
	TS4	**D**, **L** and $\hat{\mathbf{E}}$

Table 4.3: Manually selected features for the classification of crickets and katydids. Whereas for crickets three features have been determined, for katydids in each of the four time scales TS1, ..., TS4, three features have been manually selected.

Classification results are given for three feature set compositions. For example, for the cricket data set these feature set compositions are defined as follows: In feature set composition A: (**D**), (**L**), (**F**), three classifiers $\mathcal{C}^1$, $\mathcal{C}^2$ and $\mathcal{C}^3$ are combined, each of them gets a single feature as input ($\mathcal{C}^1$ on (**D**), $\mathcal{C}^2$ on (**L**) and $\mathcal{C}^3$ on (**F**)). For feature set composition B: (**D**, **L**), (**D**, **F**), (**L**, **F**), pairs of features are used to build the individual classifiers $\mathcal{C}^1$, $\mathcal{C}^2$ and $\mathcal{C}^3$, whereas in feature set composition C: (**D**, **L**, **F**), a single classifier $\mathcal{C}$ is based on all three features (**D**,**L**, **F**). In Table 4.4 the feature compositions

feature set composition	number of classifiers	input features
A	3	$(\mathbf{x}_1)$, $(\mathbf{x}_2)$, $(\mathbf{x}_3)$
B	3	$(\mathbf{x}_1, \mathbf{x}_2)$, $(\mathbf{x}_1, \mathbf{x}_3)$, $(\mathbf{x}_2, \mathbf{x}_3)$
C	1	$(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$

Table 4.4: The three feature set compositions A, B and C for the classifier input feature vectors $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Hereby (a, b) denotes the concatenation of feature vector a and feature vector b.

A, B and C are given for the classifer input features $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$.

Classification rates for the three static combining schemes the FCT, the CFT and the CTF architecture (see Section 3.4) are depicted in Table 4.5 and 4.6. Table 4.5 contains the results for the FCT architecture.

architecture	feature set composition	AVR	PF	VOT
FCT	A/B/C	10.81	9.95	11.89
FCT	A/B/C	25.97	33.53	30.25

Table 4.5: Classification results for the cricket data set (upper part) and the katydid data set (bottom part) for the FCT fusion architecture and different aggregation rules for the temporal integration. Averaging (AVR), probabilistic function (PF) and majority voting (VOT) are the applied aggregation operators. A, B and C are the feature set compositions as defined in Table 4.4.

For the FCT architecture (see Figure 3.4(b)) the results for the three feature set compositions A, B and C are equal, as data fusion is the first step. Average fusion (AVR), probabilistic fusion (PF) and majority voting (VOT) are the fusion mappings used for the temporal integration of local classifier decisions (see Eq. 3.17).

In contrast to the FCT fusion scheme for the CFT and the CTF architecture (see Figure 3.4(a) and Figure 3.4(c)), two aggregation operators are used, one for decision fusion over the feature space and another one for the temporal integration of local decisions (see Table 4.6). Due to the fact that in feature set composition C all feature vectors are concatenated there is only one local feature vector to be locally classified which leads to one decision per time window (see Figure 3.3) to be temporally integrated. Therefore, for this feature composition the FCT fusion architecture is equivalent to the CFT and the CTF architecture if averaging (AVR) is used for fusion over the feature space and for both fusion schemes the same aggregation operator is used for the temporal integration. The classification results in Table 4.5 can be also found in Table 4.6 (see the underlined numbers).

For the described application the classification results of feature set composition A are not as good as the classification results of the two other feature set compositions, because the classifiers trained on the individual features seem to be too weak to build separate classifiers. The best classification performances with static combining paradigms have been achieved for both families with different feature set compositions (crickets: feature set composition C: 9.84 %, katydids: feature set composition B: 23.18 %). Hereby it is obvious, that the class membership estimates of the individual classifiers are important for the combination of the classifiers, as the com-

arch.	feature comp.	$\frac{AVR}{AVR}$	$\frac{AVR}{PF}$	$\frac{AVR}{VOT}$	$\frac{PF}{AVR}$	$\frac{PF}{PF}$	$\frac{PF}{VOT}$	$\frac{VOT}{AVR}$	$\frac{VOT}{PF}$	$\frac{VOT}{VOT}$
CFT	A	**35.35**	35.46	35.57	24.00	24.11	23.24	44.00	44.00	52.54
	B	**12.32**	13.08	13.41	12.22	12.22	12.65	13.08	13.51	14.16
	C	**10.81**	9.95	11.89	10.70	9.84	11.89	11.89	11.89	11.89
CTF	A	35.35	43.46	44.00	20.65	47.68	41.08	74.25	74.09	74.19
	B	12.32	13.95	13.08	12.22	15.89	12.00	58.66	58.44	59.19
	C	10.81	9.95	11.89	10.81	9.95	11.89	55.65	55.22	56.29
CFT	A	**29.15**	31.54	32.84	25.77	29.65	25.97	37.01	41.89	44.08
	B	**23.18**	27.56	27.06	23.38	28.16	26.17	28.26	37.61	31.64
	C	**25.97**	33.53	30.25	25.77	34.03	30.25	30.25	38.11	30.25
CTF	A	29.15	40.20	37.01	26.57	41.79	37.81	57.35	56.36	60.08
	B	23.18	31.44	28.26	24.28	31.74	28.96	50.98	50.08	52.88
	C	25.97	33.53	30.25	25.87	37.21	30.35	54.70	54.70	55.53

Table 4.6: Classification rates for the cricket data set (upper part) and the katydid data set (bottom part) for the CFT and the CTF architecture and different aggregation rules. Here $\frac{a}{b}$ denotes that decision fusion is applied with aggregation operator a, whereas the temporal combination is applied with aggregation operator b. Averaging (AVR), majority voting (VOT) and the probabilistic function (PF) are the applied fusion mappings. A, B and C are the feature set compositions as defined in Table 4.4.

bination with majority voting (see Eq. 3.11) usually performs not as good as the combination with averaging (see Eq. 3.21) and the combination with the probabilistic function (see Eq. 3.28).

To examine the influence of the second training phase in which the decision fusion mapping is trained with different decision template algorithms, the CFT architecture may be considered as static reference architecture, since the classifiers in the decision template approach are trained with the same feature combinations. Furthermore, the CFT architecture with averaging (AVR) for the combination over the feature space is equivalent to the combination with decision templates if for both fusion schemes the same aggregation operator is used for the temporal integration and other particular conditions (see discussion in Section 3.5.2). Therefore, for both families and each feature set composition the bold error rates in Table 4.6 are well comparable to the error rates of the proposed decision template algorithms (see Table 4.7), because for both fusion schemes averaging is used for the temporal integration.

alg.	feature comp.	ρ=0.0	ρ=0.2	ρ=0.4	ρ=0.6	ρ=0.8	ρ=1.0
	A	17.51	17.84	18.05	18.38	18.38	18.27
DT	B	11.46	10.59	10.49	10.49	10.27	10.05
	C	12.32	10.81	10.49	10.49	10.59	10.38
	A	18.16	18.27	18.27	18.49	18.81	19.03
TDT1	B	11.89	11.24	10.81	9.95	9.84	9.95
	C	13.62	11.57	10.92	10.49	10.49	10.70
	A	18.59	17.84	18.16	18.49	18.16	17.95
CDT1	B	10.81	10.05	9.73	9.41	9.51	9.41
	C	15.68	10.38	9.62	9.51	10.05	9.08
	A	22.19	21.29	20.50	20.20	20.30	21.00
DT	B	19.50	17.21	17.01	18.91	19.40	20.10
	C	24.78	21.19	21.49	22.49	23.08	23.78
	A	23.08	22.49	21.99	21.89	22.09	21.69
TDT1	B	21.19	19.40	18.21	18.71	19.40	20.10
	C	24.48	21.69	21.79	21.49	21.99	22.59
	A	22.59	21.69	22.59	22.09	22.39	22.99
CDT1	B	20.60	18.91	17.71	18.01	18.71	18.61
	C	22.09	18.51	19.00	18.61	18.71	20.10

Table 4.7: Classification results for the cricket data set (upper part) and the katydid data set (bottom part) for the proposed decision template algorithms and different overlaps of training and validation sets given by different values of ρ (see Eq. 3.12). The suffixed number defines S, the number of decision templates per class to calculate the virtual decision template. A, B and C are the feature set compositions as defined in Table 4.4.

For this data set we calculated a single DT, 3 TDTs per class and 5 CDTs per class on randomly chosen training and validation sets utilising the Manhattan distance (L_1-norm) as similarity measure (see Eq. 3.33). Additionally, six different training and validation set compositions are determined by the overlap parameter ρ (see Eq. 3.12). In order to deal with several characteristic decisions for whole time series we applied the TDT and the CDT algorithm to deal with the temporal alteration of classifier decisions within time series (see Figure 4.30). The classification results in Table 4.7 show performance improvements by training the decision fusion mapping with decision template algorithms in comparison to the error

rates of the static fusion schemes (see Table 4.5 and Table 4.6). The only exception are the classification results for the cricket data set based on feature set composition C.

In the numerical evaluation of the DT, the TDT and the CDT algorithm we have found, that for this application the TDT fusion scheme did not improve the classification performance of the DT fusion scheme. Furthermore, for feature set composition A, the CDT algorithm did not improved the error rates of the DT algorithm. But for feature set composition B and C, the CDT fusion scheme improved the performance of the classifier system significantly. Here for $\rho = 0.0$ and $\rho = 0.2$ there is no performance improvement in comparison to the DT fusion scheme. However, for ρ

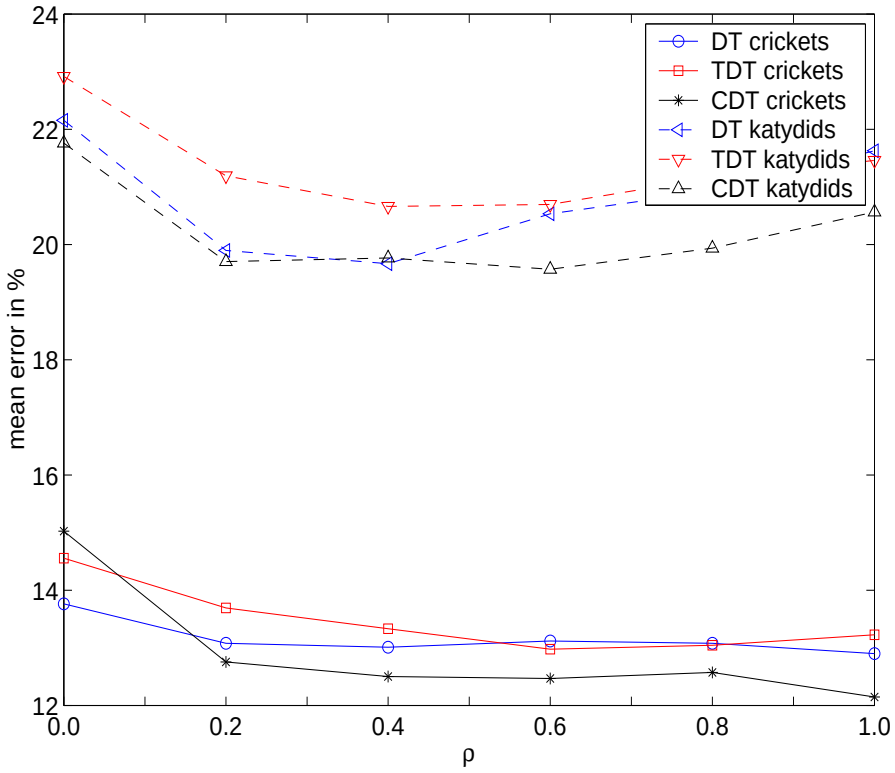


Figure 4.29: Mean classification results for the DT, the TDT and the CDT fusion scheme dependent on different overlaps of training and validation data given by six values of ρ.

in the range of 0.6 to 1.0 the CDT fusion scheme shows performance improvements in many cases. The best results for the cricket data set have been achieved with the CDT algorithm by using feature set composition B (error rate < 9.80 % for ρ in the range of 0.4 and 1.0). For the katydid

data set the lowest error rate (17.01 %) has been achieved with the DT algorithm, feature set composition B and $\rho = 0.4$.

In order to illustrate the classification performance of the proposed decision template algorithms for all three feature set combinations together, we averaged the error rates of the individual feature set combinations (see Table 4.7). The results for the DT, the TDT and the CDT algorithm are graphically represented in Figure 4.29 for both families of the *Orthoptera*. Again six different values of the overlap parameter ρ are considered.

Automatically Selected Features

In this Section the automatically selected features of both hierarchy levels (see Figure 4.27, 4.28 and B.1 to B.4) served as input of the classifiers (crickets: 3 classifiers, katydids: for each time scale 3 classifiers). Whilst for the first hierarchy level the ten best feature components are considered, for the second hierarchy level the fifteen best feature components are used as classifier input. Classification results for the three static fusion schemes the FCT, the CFT and the CTF architecture are given in Table 4.8.

architecture	$\frac{AVR}{AVR}$
FCT	13.51
CFT, CTF	7.24
FCT	14.43
CFT, CTF	16.22

Table 4.8: Classification results for the cricket data set (upper part) and the katydid data set (bottom part) for the FCT, the CFT and the CTF fusion architecture. Decision fusion is applied with averaging (AVR) and the temporal combination as well.

Not surprisingly, the error rates of the CFT and the CTF fusion scheme are very low, because the CFT architecture has been used to combine the classifiers during the feature selection procedure (see Section 4.5). In contrast to the manually selected features (see Table 4.5 and Table 4.6) the classification performance could be improved for both families by using automatically selected features as classifier input. The performance improvement for katydid species is hereby relatively large (feature composition B: 22.64 %, automatically selected features: 14.43 %).

As for manually selected features (see Table 4.7), for automatically selected features the performance could be improved by learning the second fusion layer with decision templates, see Table 4.9. Again we calculated a single DT, 3 TDTs per class and 5 CDTs per class on randomly chosen training and validation sets utilising the L_1-norm as similarity measure. Error rates are given for six different training and validation set compositions determined by the overlap parameter ρ.

algorithm	ρ=0.0	ρ=0.2	ρ=0.4	ρ=0.6	ρ=0.8	ρ=1.0
DT	9.08	7.24	6.70	7.03	7.46	7.35
TDT1	11.24	7.89	7.24	7.03	6.92	7.03
CDT1	9.19	7.14	7.14	6.92	6.27	<u>6.27</u>
DT	17.81	16.52	16.52	15.52	15.02	14.63
TDT1	18.81	17.71	16.82	16.52	16.32	16.32
CDT1	17.61	16.72	15.62	15.72	14.53	<u>13.93</u>

Table 4.9: Classification results for the cricket data set (upper part) and the katydid data set (bottom part) for different decision template algorithms and different overlaps of training and validation sets given by several values of ρ (see Eq. 3.12). The suffixed number defines S, the number of decision templates per class to calculate the virtual decision template.

For this feature set composition the TDT fusion scheme did not improved the DT algorithm. However, the CDT fusion scheme improved the performance of the classifier system. The best classification results for both families have been achieved with the CDT fusion scheme for ρ = 1.0.

In order to present classification results for the whole classifier hierarchy (see algorithm HOC) we give error rates for the first, the second and both hierarchy levels together (see Table 4.10). Hereby the overall decision is wrong if the decision of the first or the decision of the second hierarchy level is wrong. On that score the overall error rate is always less or equal than the sum of the error rates of the first and the second hierarchy level.

The results in Table 4.10 are given for the best parameter setting of the second hierarchy level evaluated in Table 4.9 (see the underlined numbers). At this, the FCT fusion scheme is used in the first, and the CDT approach in the second level of the hierarchy.

hierarchy layer	crickets	katydids	both families
L1	2.70	2.69	2.70
L2	6.27	13.93	10.25
L1$\hat{+}$L2	8.65	16.32	12.64

Table 4.10: Error rates for the *Orthoptera* data set based on automatically selected features. Classification results are given for crickets, katydids and both families by considering the first layer of the hierarchy. Furthermore, the results for hierarchy level 1 (L1), for hierarchy level 2 (L2) and for both hierarchy levels together (L1$\hat{+}$L2) are shown.

4.7 Summary and Conclusion

In Section 4.1 the sound production of the *Orthoptera* is described which reveals significant features for the automated classification. Based on the knowledge of the sound production in Section 4.2, a hierarchical system for the classification of *Orthoptera* species by their sounds is presented. The classifier system consists of two levels and the extraction of relevant features in four time scales. Whereas in the first level the family of the *Orthoptera* is determined, in the second level the classification to the species level is family-specific.

The discrimination power of the individual features is evaluated and discussed in Section 4.4. To evaluate combinations of feature components, in Section 4.5 feature components are automatically selected by the sequential forward selection algorithm for multiple classifiers. For cricket species we have shown, that the feature compositions that are used in traditional bioacoustics (the pulse distances (**D**), the pulse length (**L**) and the pulse frequencies (**F**)) are significant features for the automated classification to species level as well (see Figure 4.28 and Section 4). The same observation we have made for katydid species in which similar features exposed to be discriminative to classify to species level (see Figure B.1 to Figure B.4).

The experimental results of Section 4.6 show performance improvements due to:

1. the extraction of features in several time scales (see Table 4.2),

2. learning the decision fusion mapping with decision templates (see Table 4.7),

3. using the sequential forward selection algorithm to derive feature sets for multiple classifiers (see Table 4.8) and

4. using the CDT algorithm to increase the expression power of the fusion layer (see Table 4.9).

All these modifications bring the MCS a step closer to the real problem of classifying bioacoustic time series.

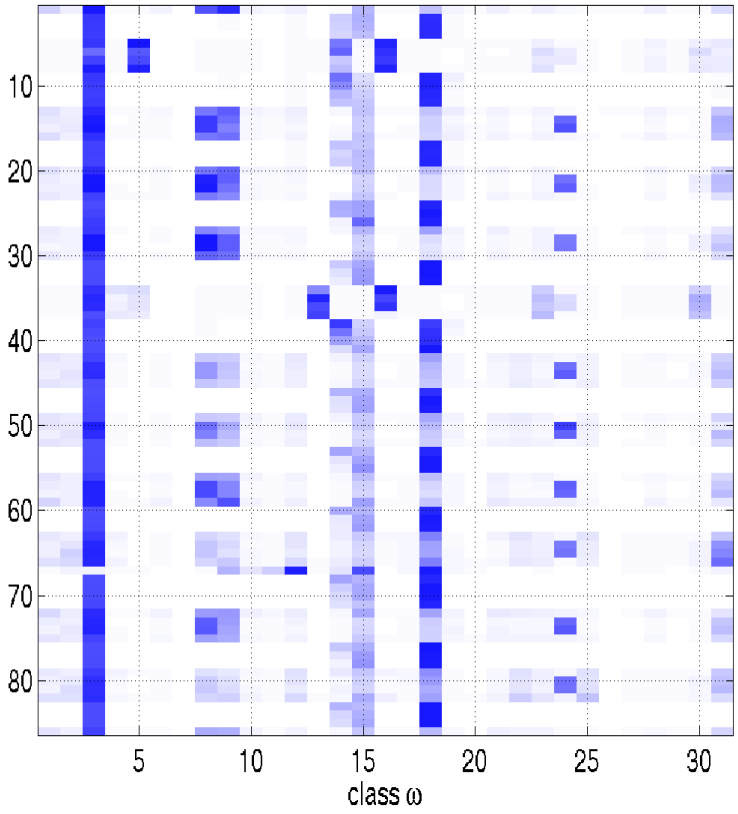


Figure 4.30: The classifier outputs $\mathcal{C}(\mathbf{x}(j)) = (\mathcal{C}_1(\mathbf{x}(j)), ..., \mathcal{C}_{31}(\mathbf{x}(j)))$, $j = 1, ..., 86$ of the species *Aclodes chamocoru* ($\omega = 3$) shows confusions to three other species $\omega = 8$, $\omega = 18$ and $\omega = 24$ which alternate between these species.

In the following we exemplify the performance gain of the CDT approach in the temporal domain by analysing the temporal structure of local classifier decisions that are derivated from bioacoustic time series.

Figure 4.30 for example shows the classifier outputs for one individual of the cricket species *Aclodes chamocoru* ($\omega = 3$) over time. Hereby it can be observed, that the classifier outputs show a variety of alternating misclassifications over the whole time series. Confusions are oscillating quasi-deterministically and appear to the classes $\omega = 8$, $\omega = 18$ and $\omega = 24$. These confusions arise, because features, e.g. pulse distances (**D**), show oscillations over time (see Figure 4.17). Provided that the outputs of the base classifiers show different characteristic patterns, which is a typical behaviour in time series classification, the CDT approach may enhance performance of the classifier system, since a fusion mapping based on several templates per class allows to learn several characteristic classifier decisions over time.

The evaluated classifier system is able to discriminate between 53 different *Orthoptera* species; 31 species of the insect family *Gryllidae* and 22 species of the insect family *Tettigoniidae*. Hereby the classification error in the first level of the hierarchy is 2.70 %. In the second level of the hierarchy we achieved a classification error of 10.25 % by using the CDT approach. Due to the fact that the errors in the first and in the second hierarchy level are highly uncorrelated, the error rate of the overall classifier system (see algorithm HOC) is 12.64 %. In comparison to the species classification systems presented in Section 1.2, the proposed system is able to identify a very large number of species. The results are very promising by considering that the training and test recordings are always recordings from different species and that a large number of recordings are made outdoors.

Chapter 5

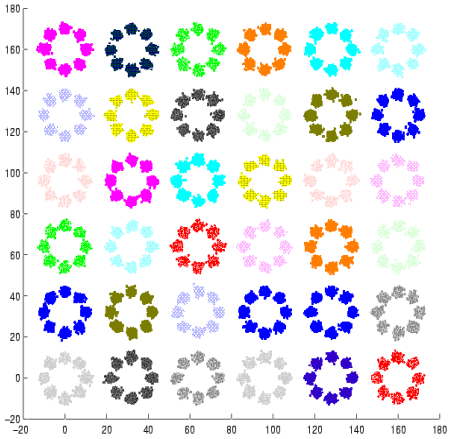
Synthetic Data

In order to evaluate the proposed fusion schemes for the classification of temporal sequences in this Section classification results for a synthetic data set consisting of Gaussian distributions are given. The results of the proposed fusion schemes, particularly the decision template algorithms, are discussed by considering the structure of this data set.

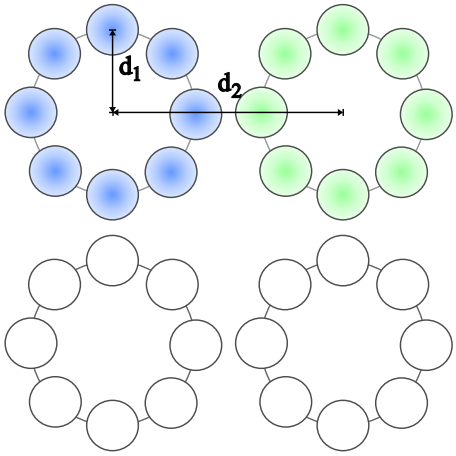
5.1 Data Set Description

The synthetic data set used in the numerical experiments consists of $L = 36$ classes, each class with χ time series, where each time series $(X^{\mu}(j))_{j=1}^{\mathcal{J}}$ is represented by a small cluster of $\mathcal{J} = 20$ feature vectors. Each time series X^{μ} consists of $I = 3$ feature streams, so $X^{\mu}(j) = (\mathbf{x}_1^{\mu}(j), \mathbf{x}_2^{\mu}(j), \mathbf{x}_3^{\mu}(j))$. The feature vectors $\mathbf{x}_i^{\mu}(j) \in \mathbb{R}^2$, $i = 1, 2, 3$ are generated in the following way: The centers of gravity of the 36 classes are placed on a regular 6×6 2D-grid with a feature dependent distance d_2^i in x- and y-direction (see Figure 5.1).

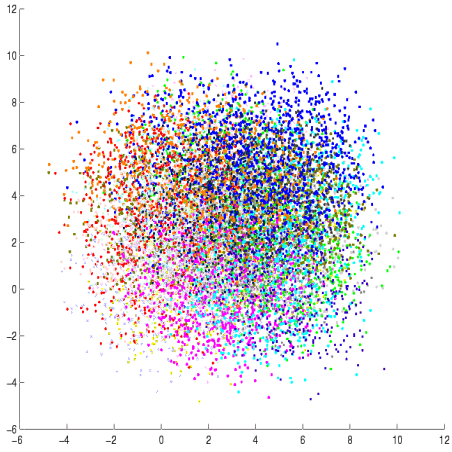
Then the χ centers of gravity of the time series are regularly placed on the sphere with radius d_1 around the center of gravity of the corresponding class. All $\mathcal{J}$ feature vectors $\mathbf{x}_i^{\mu}(j) \in \mathbb{R}^2$ of each feature stream are randomly generated through a $2D$ Gaussian distribution function located in the center of gravity of the μ-th time series with $\sigma^2 = 1.0$. In Figure 5.1 such a data set is shown, class labels are coded through different grey values. Figure 5.1(a) shows a data set with $L = 36$ classes and $\chi = 8$ time series per class. Here the distance parameters d_1 and d_2 (see Figure 5.1(b)) for the cluster generation are chosen in such a way that the time series of all classes are well separated. The data set used in the numerical experiments consists of $\chi = 20$ time series, and has highly overlapping clusters as depicted in Figure 5.1(c).



(a) Data set with $L = 6 \times 6 = 36$ well separated classes and with $\chi = 8$ time series per class, each with $\mathcal{J} = 20$ feature vectors ($d_1 = 10$, $d_2 = 32$)



(b) Data generation based on the distance parameters (d_1, d_2)



(c) Data set used in the experiments with $L = 36$ highly overlapping classes, $\chi = 20$ time series per class, ($d_1 = 2.0$, $d_2^1 = 0.65$, $d_2^2 = 0.77$, $d_2^3 = 0.89$).

Figure 5.1: Synthetic data set consisting of time series of Gaussian distributed feature vectors. The individual time series of a single class are regularly placed on the sphere.

5.2 Experiments and Results

For the experiments with the synthetic data set we used the $\tilde{k}$ fold cross-validation procedure (see Section 3.7) with $\tilde{k}$ = 20 cycles. 19 time series of each class have been used to train the overall classifier ensemble and one time series per class has been used for the classification test. In the numerical evaluation one cross validation test has been made.

The evaluation of single features was achieved by considering the classification results of the individual feature streams $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Local and temporal fusion was accomplished by the FCT fusion architecture (see Figure 3.4(b)) with averaging for the temporal integration. Fuzzy-K-nearest neighbour classifiers are used to calculate local decisions, each of them gets single feature as input.

The classification performance of the individual feature streams $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ of the synthetic data set is rather weak ($\approx$ 96 % error rate). Combining these features with data fusion (= concatenation of feature vectors) leads to a significant improvement of the classification results, e.g. for combinations of two features $(\mathbf{x}_1, \mathbf{x}_2)$, $(\mathbf{x}_1, \mathbf{x}_3)$ and $(\mathbf{x}_2, \mathbf{x}_3)$ the error rates decreased to the range of 65.14 % to 75.56 % and for the combination of all three features $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ the error rate decreased to 25.56 %, see Table 5.1.

feature	error in %
$(\mathbf{x}_1)$	96.94
$(\mathbf{x}_2)$	96.45
$(\mathbf{x}_3)$	96.25
$(\mathbf{x}_1, \mathbf{x}_2)$	75.56
$(\mathbf{x}_1, \mathbf{x}_3)$	68.47
$(\mathbf{x}_2, \mathbf{x}_3)$	65.14
$(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$	25.56

Table 5.1: Error rates for the synthetic data set based on single features $(\mathbf{x}_1)$, $(\mathbf{x}_2)$ and $(\mathbf{x}_3)$, pairs of features $(\mathbf{x}_1, \mathbf{x}_2)$, $(\mathbf{x}_1, \mathbf{x}_3)$, and $(\mathbf{x}_2, \mathbf{x}_3)$, and all three features $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$. Local and temporal fusion is performed by the FCT architecture.

This huge performance improvement is due to the fact that the individual feature streams have been calculated on independent distributions which leads to a high diversity between the individual base classifiers (see discussion in Section 3.2 and [KW01b]).

In the following, results for the two remaining static fusion architec-

tures: the CFT and the CTF architecture proposed in Section 3.4 are presented. Three different feature set compositions (see Table 4.4) are considered to evaluate the classification performance of static and adaptive combining paradigms. This means, that in feature set composition A: $(\mathbf{x}_1), (\mathbf{x}_2), (\mathbf{x}_3)$ three classifiers $\mathcal{C}^1$, $\mathcal{C}^2$ and $\mathcal{C}^3$ are combined, each of them gets single features as input ($\mathcal{C}^1$ on $(\mathbf{x}_1)$, $\mathcal{C}^2$ on $(\mathbf{x}_2)$ and $\mathcal{C}^3$ on $(\mathbf{x}_3)$). For feature set composition B pairs of features $(\mathbf{x}_1, \mathbf{x}_2)$, $(\mathbf{x}_1, \mathbf{x}_3)$, $(\mathbf{x}_2, \mathbf{x}_3)$ are used to build the individual classifiers $\mathcal{C}^1$, $\mathcal{C}^2$ and $\mathcal{C}^3$, whereas in feature set composition C: $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ a single classifier $\mathcal{C}$ is based on all three features $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$.

The lowest error rates with static combining paradigms have been achieved with feature set composition C (error rates in the range of 24.03 to 34.17 %) which are significantly better than the results with feature set composition A and B (see Table 5.2). For the most combinations of aggregation rules the CFT architecture outperformed the CTF fusion scheme. However, for both fusion architectures and all feature set combinations

arch.	feature comp.	$\frac{AVR}{AVR}$	$\frac{AVR}{PF}$	$\frac{AVR}{VOT}$	$\frac{PF}{AVR}$	$\frac{PF}{PF}$	$\frac{PF}{VOT}$	$\frac{VOT}{AVR}$	$\frac{VOT}{PF}$	$\frac{VOT}{VOT}$
CFT	A	**85.56**	86.67	85.69	80.97	80.97	80.14	89.58	89.86	92.08
	B	**41.53**	40.28	47.92	34.72	33.75	42.22	51.81	51.81	64.72
	C	**25.56**	24.03	34.03	25.69	24.03	34.03	34.03	34.03	34.03
CTF	A	85.56	94.72	89.31	82.22	93.19	86.25	96.94	96.67	94.44
	B	41.53	52.64	51.11	38.33	44.31	55.42	62.92	61.81	65.28
	C	25.56	24.03	34.03	25.56	24.03	34.03	25.56	24.03	34.17

Table 5.2: Classification results for the synthetic data set by using the CFT and the CTF architecture and different aggregation rules. Here $\frac{a}{b}$ denotes that decision fusion is applied with aggregation operator a whereas the temporal combination is applied by aggregation operator b. Averaging (AVR), voting (VOT) and the probabilistic function (PF) have been applied as aggregation operators. A, B and C are the feature set compositions as defined in Table 4.4.

the classification performance of simple average fusion ($\frac{AVR}{AVR}$) could be increased by using the probabilistic function (PF) for decision fusion and for the temporal integration as well. But, the combination with majority voting (VOT) has not performed as good as the combination with averaging and the probabilistic function. The best classification results (24.03 % error rate) have been achieved with the probabilistic function for the temporal

combination (see the underlined numbers in Table 5.2).

To examine the influence of the second level training to the classification power of the classifier ensemble in Table 5.3 the error rates for the adaptive fusion paradigms discussed in Section 3.6 are given. We applied the AM, the DT, the NB fusion schemes and the pseudoinverse solution (PI) on 19 randomly chosen training and validation sets. For the DT fusion scheme the Euclidean distance is used as similarity measure (see Eq. 3.33). Additionally, in Table 5.3 the error rates are presented for different training and validation set compositions that are determined by the overlap parameter ρ (see Eq. 3.12).

alg.	feature comp.	ρ=0.0	ρ=0.2	ρ=0.4	ρ=0.6	ρ=0.8	ρ=1.0
AM	A	87.22	85.83	85.28	84.72	84.44	83.61
	B	37.50	36.11	35.83	36.39	35.83	36.39
	C	33.61	31.39	29.17	27.50	25.56	25.00
DT	A	78.06	78.89	79.17	79.17	79.72	80.56
	B	17.22	20.83	22.78	25.56	30.28	32.78
	C	13.33	14.72	18.89	21.11	22.50	26.11
NB	A	89.17	89.17	87.78	88.33	86.67	89.72
	B	56.39	55.83	53.89	50.00	47.78	49.31
	C	33.33	31.39	31.39	32.50	33.61	35.42
PI	A	81.94	83.89	85.00	85.56	86.39	87.50
	B	41.67	42.22	42.78	43.33	45.00	47.22
	C	35.83	32.78	33.06	34.44	35.00	37.22

Table 5.3: Error rates for adaptive fusion paradigms for different compositions of the training and validation sets and different feature set compositions for the base classifiers. Training and validation set compositions have been determined by the parameter ρ. AM denotes the associative memory, DT denotes the decision template, NB denotes the naive Bayes combination and PI denotes the pseudoinverse solution. A, B and C are the feature set compositions as defined in Table 4.4.

Because of the fact that under several requirements discussed in Section 3.5.2 the DT fusion scheme utilising the normalised correlation as similarity measure is equivalent to the CFT architecture the bold error rates in Table 5.2 are well comparable to the error rates of the trainable fusion schemes, since in both approaches averaging is used for the temporal integration.

For the discussion of the classification results in Table 5.3, Section 3.6 reveals important links between the individual fusion schemes. Due to the fact that for this data set the number of time series per class $\omega \in \Omega$ in the validation set $\mathcal{D}^{\mathrm{V}}$ is constant, the linear heteroassociative memory (AM) and the decision template fusion scheme using the normalised correlation as similarity measure (see Eq. 3.34) are equivalent (see Eq. 3.53). Classification results for both fusion schemes are given in Table 5.3, line AM.

alg.	feature comp.	ρ=0.0	ρ=0.2	ρ=0.4	ρ=0.6	ρ=0.8	ρ=1.0
	A	30.56	31.67	34.44	34.72	35.83	36.94
TDT1	B	6.11	6.11	7.50	8.61	10.83	14.17
	C	9.44	10.56	10.83	13.33	14.72	16.11
	A	31.94	33.61	35.28	37.22	39.44	40.83
TDT2	B	6.94	6.11	9.17	10.28	12.22	14.72
	C	10.28	9.44	10.83	12.50	13.89	16.11
	A	36.39	38.06	38.06	37.78	41.39	43.61
TDT3	B	7.22	7.22	9.44	10.56	12.78	15.56
	C	9.72	9.44	11.39	12.78	13.89	16.11
	A	45.83	50.56	48.06	53.06	52.78	55.00
CDT1	B	18.61	20.00	19.44	22.22	23.89	26.11
	C	21.94	18.61	14.44	16.67	18.89	18.33
	A	56.39	57.50	59.72	60.56	60.00	63.06
CDT3	B	20.00	19.17	20.28	20.56	22.78	26.39
	C	16.11	14.72	13.61	16.39	18.33	18.33
	A	65.00	64.44	66.67	66.11	68.89	68.06
CDT2	B	19.17	18.33	20.28	21.94	23.61	26.11
	C	16.67	13.61	14.72	16.67	17.22	18.89

Table 5.4: Error rates for the TDT and the CDT algorithm for different compositions of training and validation sets and different feature set compositions for the base classifiers. Training and validation set compositions have been determined by the parameter ρ. The suffixed number defines S, the number of decision templates per class to calculate the virtual decision template. A, B and C are the feature set compositions as defined in Table 4.4.

For this data set for each feature set combination A, B and C the linear associative memory (AM) shows a quite robust classification performance.

Nevertheless, the DT fusion scheme with the L_2-norm as similarity measure significantly outperformed the AM, the NB and the PI fusion scheme, even for each value of ρ and each feature set combination A, B and C. Unfortunately, the error rates of the naive Bayes combination for this data set are even higher than the error rates of the static combining schemes given in Table 5.3.

The classification results of adaptive fusion schemes with a single matrix operator (see Table 5.3) can be further improved by the TDT and the CDT algorithm (see Table 5.4). The best results (6.11 % error rate) have been achieved with the TDT algorithm, by using feature set composition B, one decision template per class to calculate the virtual decision template, e.g. $S = 1$ and $\rho = 0.0$ where the training set $\mathcal{D}$ and the validation set $\mathcal{D}^{\mathrm{V}}$ are disjoint. In comparison to all other fusion schemes, e.g. CFT, CTF, AM, DT, PI and CDT which revealed the best classification performances for feature set composition C, the TDT algorithm revealed the lowest error rates for feature set composition B.

5.3 Discussion

The basic conclusion of our experiments is, that the multiple decision template approach can improve the performance of the classifier system. In our numerical experiments with the synthetic data set the TDT and the CDT fusion scheme outperformed the standard DT approach significantly. Each of the adaptive fusion schemes is sensitive to training and validation set compositions, varied with the parameter ρ in our experiments. The TDT and the CDT fusion scheme seem to be also sensitive to the parameter S which determines for each class the number of decision templates to select (see algorithm CFTBYMDT). In our experiments with the synthetic data set and the bioacoustic data set [DSP02], the classification performance of TDTs and CDTs for $S = 1, ..., 3$ is robust. But, it has to be considered that for $S = 1$ the reliability of the virtual decision template is rather low, since it is based on a single decision template per class. Increasing S leads to a higher reliability of the virtual decision template, whereas for $S = K$ the TDT and the DT fusion schemes are equivalent.

Most of the classification results in Table 5.1, 5.2, 5.3 and 5.4 are evident by considering the structure of the data set. At first, we discuss the huge performance improvement of TDTs in comparison to DTs. This is due to the fact, that each time series in this data set has 2, 3 or 4 adjacent time series in each of the three feature spaces (the number of the adjacent time series depends on its location, see Figure 5.1(a)). During the training phase

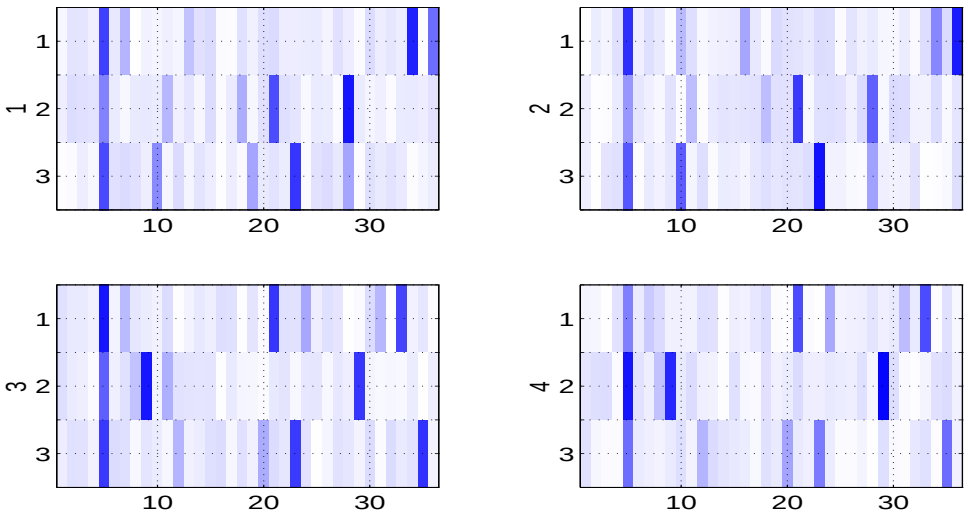


Figure 5.2: Four decision templates of class $\omega = 5$. Each decision template was calculated by the outputs of three classifiers (one classifier per feature stream). Each row represents the sub-template of a single feature.

of the fusion layer this leads to misclassifications to the classes of the adjacent time series. These misclassifications are also reflected in the decision templates. In Figure 5.2 an example of four TDTs of class $\omega = 5$ are given. Two groups of similar DTs (G1 $= \{1, 2\}$ and G2 $= \{3, 4\}$) are depicted. We found, that each group is calculated on validation data of the adjacent time series of class $\omega = 5$ (see Figure 5.3) and that the TDTs contain just confusions to the classes of the adjacent time series. Group G1 contains confusions to $\omega = 34$ and $\omega = 36$ and group G2 contains confusions to $\omega = 21$ and $\omega = 33$ (see line 1 in the decision templates in Figure 5.2). It is obvious that averaging all these DTs leads to a decision template which is quite different to one of the characteristic DTs which may be represented by each of the two groups.

The same effect (confusions typically appear to a small number of adjacent time series) may be observed by considering the classifier outputs over time (see Figure 4.30). Obviously, these confusions negatively influences the classification performance of the naive Bayes fusion scheme, as the calculation of the label matrices (see Eq. 3.50) is based on crisp classifier decisions. Therefore, the validation set has to contain a large number of samples in order to estimate the probabilities of the label matrices. Since, in the classification phase the estimated conditional probabilities are multiplied a single zero in the label matrix can lead to the combined probability estimate zero (see Eq. 3.52).

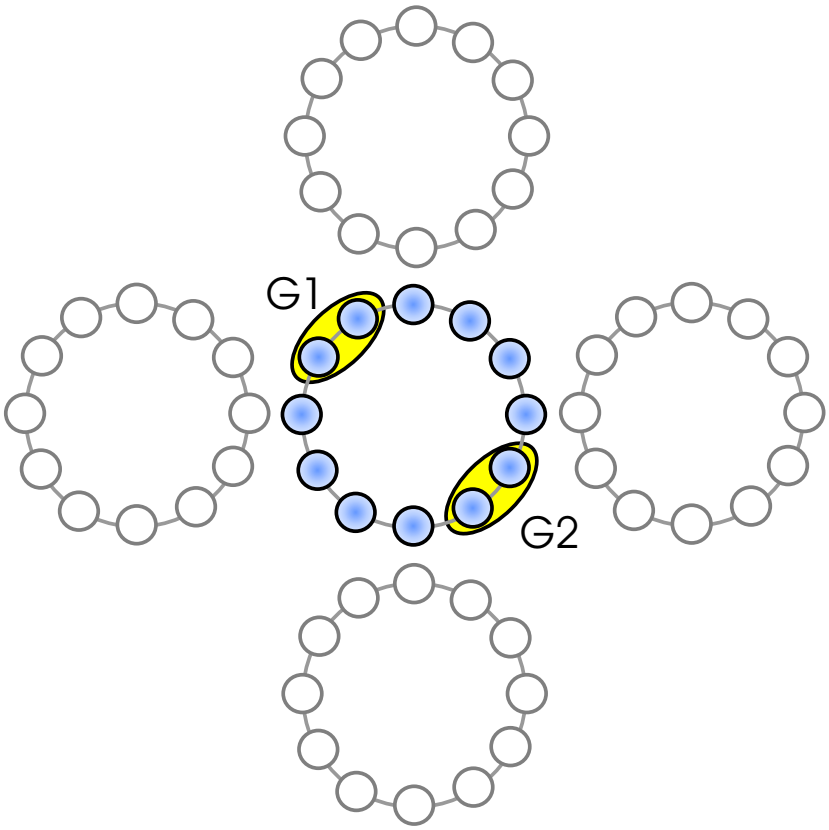


Figure 5.3: Structure of the data set (feature stream $\mathbf{x}_1$).

Finally, we want to remember that the CDT algorithm primarily offers the advantage to learn different characteristic patterns over time. Due to the fact that for the synthetic data set the feature vectors over time are generated by a Gaussian distribution and hence the temporal structure is random, we first assumed no performance improvement of the CDT fusion scheme according to the DT algorithm. Nevertheless, the CDT approach outperformed the DT fusion scheme. This performance gain can be explained by considering the CDT algorithm (see Section 3.5.2) in which the concatenated classifier outputs of validation data $\mathcal{D}_\omega^{\mathrm{V}}$ for each class $\omega \in \Omega$ are clustered separately to determine multiple decision templates per class. As the temporal structure of the time series is random and $\mathcal{D}_\omega^{\mathrm{V}}$ usually contains several time series of class ω, the clustering algorithm determines the centers of gravity of the individual time series. In particular, if the desired number of clusters is set to $|\mathcal{D}_\omega^{\mathrm{V}}|$.

Chapter 6

Summary, Conclusions and Future Research

6.1 General Nature of this Work

This thesis deals with the classification of bioacoustic time series by utilising multiple classifier systems. In the course of solving this classification problem knowledge from traditional bioacoustics reveals significant features which have been derivated from the sound production mechanism of the *Orthoptera*. New ensemble learning methods are justified by considering the temporal structure of the acoustic signals.

This reflects the major subjects of this interdisciplinary work, which mainly includes bioacoustics, pattern recognition and fusion of multiple classifiers. Whereas the knowledge from bioacoustics allows to make realistic assumptions about the data to be classified, the knowledge from pattern recognition allows to use applicative classification procedures. By using feature selection algorithms we additionally evaluated feature set compositions and found, that the most discriminative features are motivated by the sound production mechanism of the *Orthoptera*, and that these feature also have been used in traditional bioacoustics for the classification of cricket species [Nis99].

6.2 A Survey of Major Results

In the following a survey of major results is graduated by the Chapters of this thesis. Chapter 1 "Introduction" is excluded in this context, because it gives a motivation and refers related work.

Chapter 2 "Classification Methods": A review of the utilised classification methods, e.g. radial basis function (RBF) neural networks, nearest neighbour classifiers and learning vector quantisation is given. New methods to initialise the kernel parameters (centers and widths) of RBF networks by classification tree algorithms are proposed [SD00]. In contrast to the meth-

ods developed by KUBAT [Kub98] in which decision tree regions are used to calculate the kernel parameters, our methods also take the data points within a particular region into account in order to calculate the RBF centers and the RBF widths. The proposed algorithms have also been applied to initialise the RBF kernel parameters in three phase learning schemes for RBF networks [SKP01]. To introduce well-established methods for time alignment and the classification of temporal sequences these algorithms are described.

Chapter 3 "Multiple Classifier Systems": This Chapter deals with static and adaptive methods for the combination of multiple classifiers. In particular, these methods have been adapted for the classification of temporal sequences. Three static fusion architectures, the FCT, the CFT and the CTF architecture have been proposed [DSP01a, DSP01b].

By considering an adaptive fusion mapping for the combination over the feature space, we proposed to train the first fusion mapping of the CFT architecture with the decision template algorithm (see algorithm CFT-BYMDT). In contrast to the well known DT algorithm this allows to classify temporal sequences. Nevertheless, the expression power of the DT algorithm is limited, since for each class there is only one decision template. In order to increase the expression power of the fusion layer we proposed two new adaptive fusion schemes, the TDT and the CDT algorithm, to calculate multiple decision templates per class [DSP02, DPS03]. Both methods increased the expression power of the fusion layer by integrating time series specific and temporal information into the fusion layer. Due to the fact that under particular conditions the decision template fusion schemes are equivalent to the CFT architecture, the influence of the adaption of the fusion layer to the performance of the classifier system can be estimated (see discussion in Section 3.5.2).

By assuming the normalised correlation as similarity measure, the DT approach is discussed in the context of well established supervised neural network training schemes, e.g. the pseudoinverse matrix, the linear associative memory and the naive Bayes combination. The main ingredient of the discussed fusion schemes is the confusion matrix that describes the uncertainty of the regarding classifier. We found, that for each fusion scheme the confusion matrix is normalised in a different way. Hereby, fusion with decision templates is equivalent to the combination with the linear heteroassociative memory if the validation set contains for each class the same number of feature vectors. Furthermore, we showed, that for crisp classifiers the naive Bayes combination is very similar to the combination with the pseudoinverse matrix, regarding the coefficients of the

matrix operators.

Chapter 4 "*Orthoptera* Bioacoustics": This Chapter is concerned with the automated classification of *Orthoptera* species by their songs. Due to different sound structures for the two families (crickets and katydids) the classifier system consists of a hierarchy which is derivated from the biological systematics. It consists of two levels (a family level and a species level) and the extraction of relevant features in several time scales.

Major contributions of this Chapter are: (a) a detailed description of the extraction of relevant features, (b) the motivation of these features by taking the sound production mechanism of the *Orthoptera* into account, (c) a signal segmentation algorithm for the detection of pulses in several time scales (see also [DSRP01]), (d) the evaluation of the individual features by the rank distribution, (e) the evaluation of feature set compositions with the sequential forward selection algorithm for multiple classifiers and (f) a software package for the automated feature extraction and classification of *Orthoptera* species.

In comparison to other species identification systems (see Section 1.2) our system differs due to the large number of species which can be recognised. The classification results are very promising by considering that the recordings in the training set and the recordings in the test set are always from different individuals and that a large number of recordings are made outdoors. The results firmly support that the CDT algorithm improves the classification performance if the classifier outputs over time are structured and therefore can be represented through multiple decision templates (see discussion in Section 4.7).

Chapter 5 "Synthetic Data": This Chapter presents a description and results for a synthetic data set consisting of Gaussian distributions with confusions to adjacent classes for the whole time series. The temporal structure of the time series is random. Due to the fact that the expression power of the DT fusion layer is not sufficient to represent the classifier outputs of all time series of a single class, the multiple decision template algorithms outperformed the DT fusion scheme. The behaviour of the TDT and the CDT algorithm is discussed by considering the structure of the synthetic data set.

Chapter 4 and **Chapter 5**: The basic conclusion of our experiments is that adaptive fusion methods can increase the expression power of the fusion layer which may lead to a significant performance improvement of the classifier system. By increasing the number of decision templates, the expression power of the fusion layer was anymore increased. The results in

all our numerical evaluations showed, that the standard decision template algorithm is a very robust fusion scheme which is conform with the results published in [KBD01]. The TDT and the CDT fusion schemes can be powerful techniques, in particular for time series classification. TDTs perform well if confusions for the whole time series alternate between several adjacent classes (see Chapter 5), whereas CDTs improve the performance of the classifier system if confusions alternate between several classes over time which is the case for quasi-deterministically oscillating time series (see discussion in Section 4.4).

All adaptive fusion schemes proposed in this thesis are sensitive to training and validation set compositions, varied with the parameter ρ in our experiments. The multiple decision template fusion schemes seems to be also sensitive to the parameter K which determines the number of decision templates for each class and the parameter S determining the number of selected decision templates for each class (see Algorithm 3.7). For the synthetic data set (see Table 5.4) and the bioacoustic data set (see [DSP02]) the classification results firmly support, that both algorithms are robust to the parameters S and K. A detailed discussion can be found in [DPS03].

As the multiple decision template algorithms are intuitive, one can observe the classifier outputs over time (see Figure 4.30) and the feature space (see Figure 5.3) to find out, if these methods are applicable for a specific application.

6.3 Open Problems and Future Work

Not surprisingly, many open problems arise from our research. Whereas the open problems in the field of bioacoustics depend on requirements to the classifier system, the open problems in the field of classifier fusion are methodically.

One of the next goals in the DORSA project is to classify and monitor the tropical biodiversity of a particular region by analysing the spectrum of insect sounds in this region. In order to recognise the *Orthoptera* species of a particular region, the classifier system has to be adapted to be used in fully automatic environmental monitoring applications. For this task, a pre-processing step, separating the audio sources should be integrated to classify simultaneously singing individuals [GTCW96], e.g. using independent component analysis [HKO01]. These individuals may be from the same species [1] or from different species.

[1] Crickets and katydids tend to call in choruses with hundreds of individuals of the

Although the classifier system has been successfully tested for a data set of 53 species, the number of species should be increased. In particular, species which are reliable biodiversity indicators should be included to the data set. This allows the utilisation of the classifier system to estimate tropical biodiversity. To deal with a very large number of species, the classifier system has to be expanded. In this context further biological investigations together with further research in the field of classifier fusion, neural networks and automated feature selection is a promising interdisciplinary approach. Solutions to deal with such a large number of species may be to expand the hierarchy or to find further discriminative features. Here it seems important to me to apply automated feature selection together with further biological research.

Many of such feature selection algorithms have been studied in single classifier approaches, however for multiple classifier systems only a few methods are known to determine reliable feature sets for the individual base classifiers [GB03, KJ00, RBF+03]. In particular, for adaptive fusion schemes, where the fusion layer has to be calculated. Here, the overall classifier architecture (base classifiers and fusion layer) has to be calculated for each feature selection step which is very time consuming.

But, the task of feature selection and evaluation can be displaced to the fusion layer. Simple combining schemes could be extended to take the significance of the individual features into account. For example, the decision template approach could be extended by weighting coefficients. If, for example, one weighting coefficient for each row of the decision template is used, each weighting coefficient determines the influence of the regarding feature to the overall classifier system. These weighting coefficients may be trained in the second training step together with the fusion layer, or separately in a third training step.

---

same species.

Kapitel 7

Zusammenfassung

Die Artbestimmung von Tieren anhand ihrer Lautäußerungen ist eine elementare Herausforderung, um die Bioakustik einzelner Arten zu erforschen. Viele vorausgehende Untersuchungen basieren auf der manuellen Auswertung von Signalparametern, die durch den Experten aus Spektren, Sonogrammen und Amplitudenverläufen gewonnen worden sind. Darauf basierend wurden in den letzten zehn Jahren Systeme entwickelt, um die akustischen Signale der Tiere automatisch auszuwerten und zu klassifizieren.

In der vorliegenden Arbeit wird ein Gesangserkennungssystem zur Klassifikation bioakustischer Zeitreihen vorgestellt, welches exemplarisch zur Klassifikation von *Orthopteren* (Grillen und Laubheuschrecken) eingesetzt wird. Das Erkennungssystem ist in C/C++ implementiert und kann daher auf einem tragbaren Computer angewendet werden, um auf der Basis eines einzelnen Gesanges die Familie und die Art des Individuums automatisch zu bestimmen. Das System baut auf einem Mehrklassifikatorsystem auf, da sich in vielen Anwendungen gezeigt hat, dass die Performanz von Einzelklassifikatoren durch die Verwendung mehrerer Klassifikatoren und deren Fusion verbessert werden kann.

Diskriminative Merkmale für die Familien- und Artbestimmung lassen sich für Orthopteren aus dem Aufbau der artspezifischen Stridulationsorgane und aus dem physikalischen Prozess, durch den die Signale erzeugt werden, ableiten. Zusätzlich werden jedoch Merkmalsselektionsverfahren wie *Sequentielle Vorwärts Selektion* eingesetzt, um geeignete Merkmalskombinationen für die Einzelklassifikatoren zu bestimmen. Dieser interdisziplinäre Ansatz erlaubt es einerseits realistische Annahmen über die akustischen Signale zu machen und andererseits geeignete Klassifikationsmethoden zu finden.

Der methodische Schwerpunkt der Arbeit besteht in der Integration lokaler Klassifikatorausgaben aus verschiedenen Merkmalsräumen und deren temporaler Fusion. *Statische Fusionsmethoden*, bei denen die Klassifikatorausgaben durch eine statische Abbildung kombiniert werden, und *adaptive Fusionsmethoden*, bei denen die Fusionsabbildung in einer zusätz-

lichen Trainingsphase adaptiert wird, werden vorgestellt und diskutiert. Üblicherweise wird für die zusätzliche Trainingsphase eine Menge von Merkmalsvektoren (Validierungsdatensatz) klassifiziert, um die Fusionsabbildung anhand der Klassifikatorausgaben zu berechnen.

Der *Decision Template Ansatz* ist ein intuitives, adaptives Fusionsverfahren, welches zur Klassifikation von statischen Objekten anhand mehrerer Klassifikatorentausgaben herangezogen wird. Ein Decision Template repräsentiert hierbei die gemittelten Klassifikatorausgaben für alle Merkmalsvektoren einer Klasse aus dem Validierungsdatensatz. Die Fusionsabbildung wird unter Verwendung einer Verwechslungsmatrix, die in der zusätzlichen Trainingsphase berechnet wird, adaptiert. Da einige überwachte neuronale Lernverfahren wie *linearer Assoziativspeicher*, *Naive Bayes Fusion* und *Pseudoinverse Matrix* bezüglich der Klassifikatorfusion auf der gleichen Idee basieren, werden in dieser Arbeit Ähnlichkeiten zu diesen Methoden diskutiert.

In einigen empirischen Studien hat sich gezeigt, dass gerade adaptive Fusionsverfahren, insbesondere die sogenannten Decision Templates robuste und performante Methoden zur Kombination mehrerer Klassifikatoren darstellen. Bei der Klassifikation von Zeitreihen weisen diese jedoch Schwächen auf, da im Verlauf der Erkennung einer einzelnen Zeitreihe typischerweise Verwechslungen mit mehreren Klassen auftreten, die durch Standard Decision Template Methoden nicht hinreichend repräsentiert werden können.

Um die Variation der Klassifikatorausgaben besser zu repräsentieren, werden mehrere Decision Templates pro Klasse benötigt. Ausgehend von dieser Idee werden zwei neue Methoden vorgestellt, um mehrere Templates pro Klasse zu berechnen.

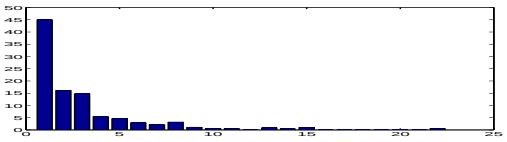
Beim *Temporal Decision Template* (TDT) Ansatz wird für jede Zeitreihe im Validierungsdatensatz genau ein Decision Template pro Klasse berechnet, um die Ausdruckskraft der Fusionsabbildung zu erhöhen. Deshalb führt der TDT Ansatz bei Klassifikationsproblemen, bei denen die Klassifikatorausgaben der einzelnen Zeitreihen unterschiedliche Muster aufweisen, zu einer Verbesserung der Performanz.

Demgegenüber werden beim *Clustered Decision Template* (CDT) Ansatz die lokalen Klassifikatorausgaben geclustert, um repräsentative Decision Templates für eine Zeitreihe zu bestimmen. Lernen von strukturierten temporalen Verwechslungen innerhalb einer Zeitreihe wird durch diesen Algorithmus unterstützt.

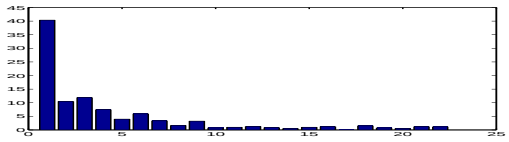
Die Performanz der vorgestellten Methoden wird in dieser Arbeit unter der Verwendung realer und synthetischer Daten diskutiert.

Appendix A

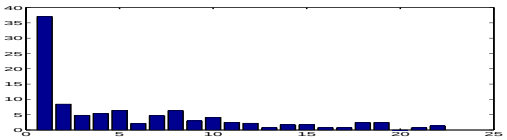
Feature Evaluation



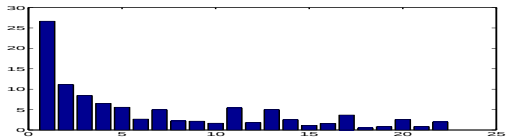
(a) Impulse distances (**D**)



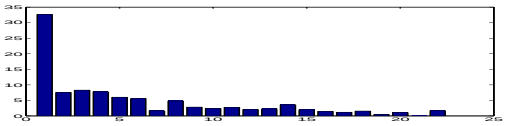
(b) Impulse distances ($\bar{\mathbf{D}}$)



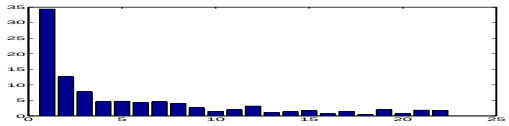
(c) Impulse length (**L**)



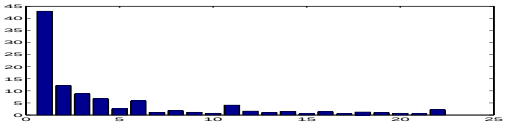
(d) Impulse frequency (**F**)



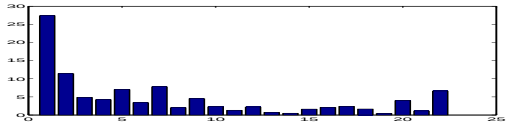
(e) Impulse energy contour (**E**)



(f) Impulse energy contour ($\hat{\mathbf{E}}$)

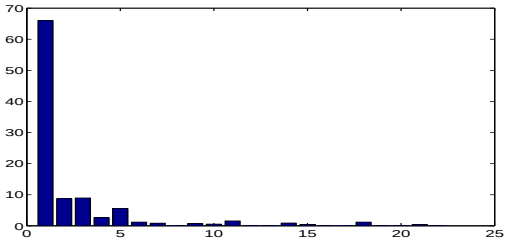


(g) TES of impulses (**T**)

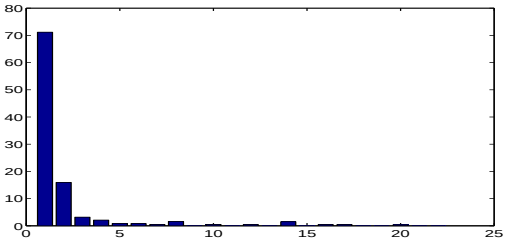


(h) Maximal amplitude of impulses (**A**)

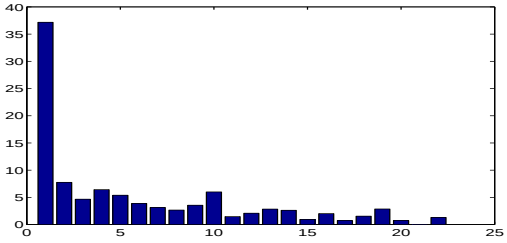
Figure A.1: Evaluation of features extracted from 5 consecutive impulses of katydid species (TS1). Whereas the first bin of the x-axis shows the classification accuracy, the other bins show the classification rate for the individual ranks. The number of ranks is given by the number of classes. For more details about the feature evaluation see Section 4.4.



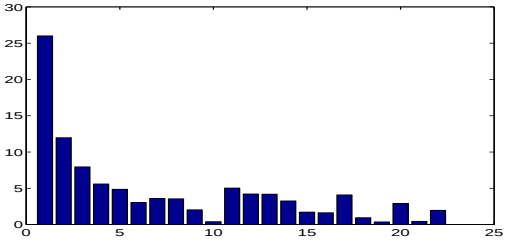
(a) Impulse distances (**D**)



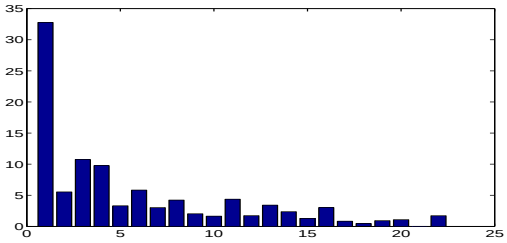
(b) Impulse distances ($\bar{\mathbf{D}}$)



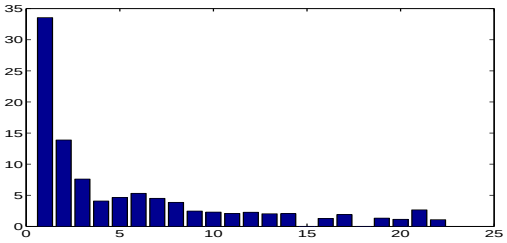
(c) Impulse length (**L**)



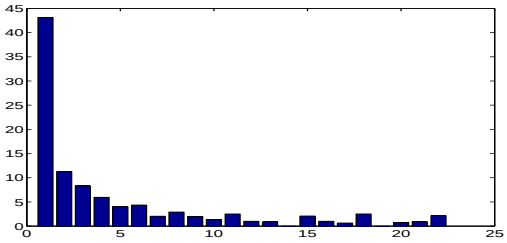
(d) Impulse frequency (**F**)



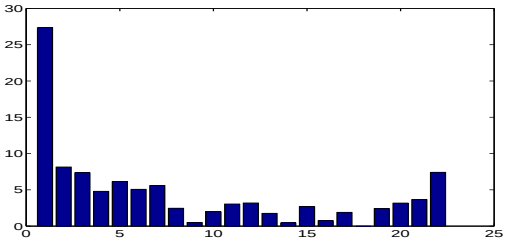
(e) Impulse energy contour (**E**)



(f) Impulse energy contour ($\hat{\mathbf{E}}$)

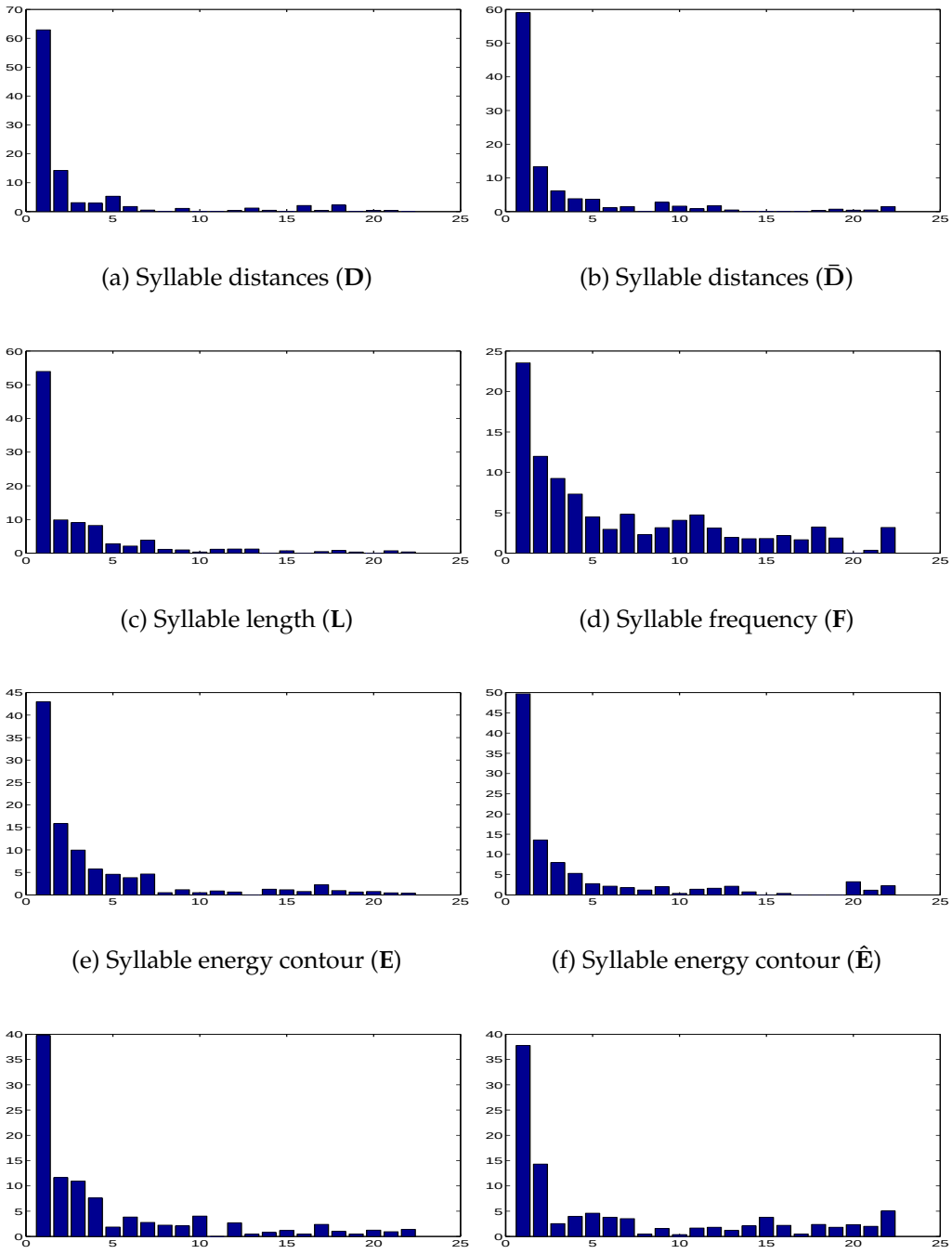


(g) TES of impulses (**T**)



(h) Maximal amplitude of impulses (**A**)

Figure A.2: Evaluation of features extracted from 21 consecutive impulses of katydid species (TS2). Whereas the first bin of the x-axis shows the classification accuracy, the other bins show the classification rate for the individual ranks. The number of ranks is given by the number of classes. For more details about the feature evaluation see Section 4.4.



(a) Syllable distances (**D**)

(b) Syllable distances ($\bar{\mathbf{D}}$)

(c) Syllable length (**L**)

(d) Syllable frequency (**F**)

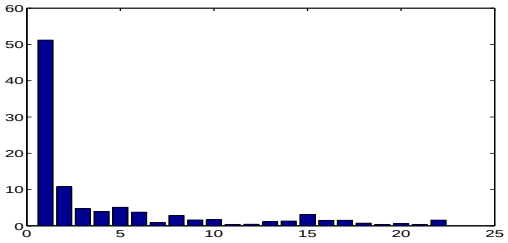
(e) Syllable energy contour (**E**)

(f) Syllable energy contour ($\hat{\mathbf{E}}$)

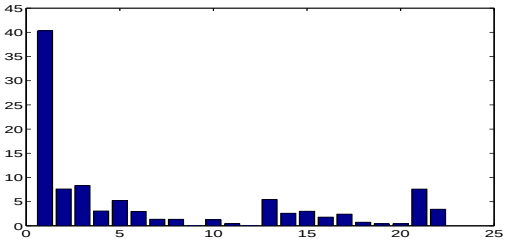
(g) TES of syllables (**T**)

(h) Maximal amplitude of syllables (**A**)

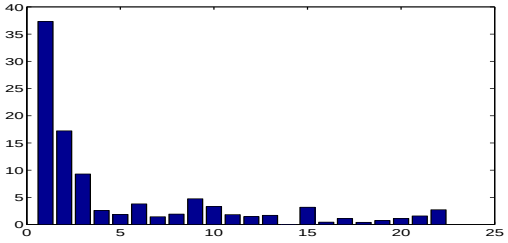
Figure A.3: Evaluation of features extracted from 5 consecutive syllables of katydid species (TS3). Whereas the first bin of the x-axis shows the classification accuracy, the other bins show the classification rate for the individual ranks. The number of ranks is given by the number of classes. For more details about the feature evaluation see Section 4.4.



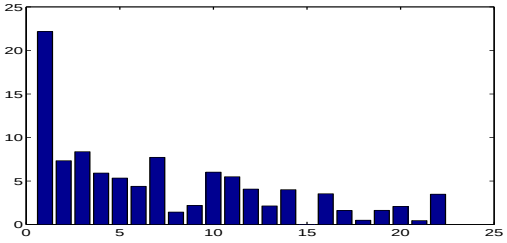
(a) Syllable distances (**D**)



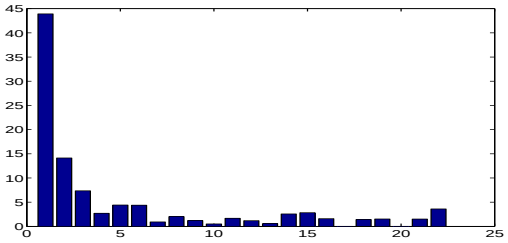
(b) Syllable distances ($\bar{\mathbf{D}}$)



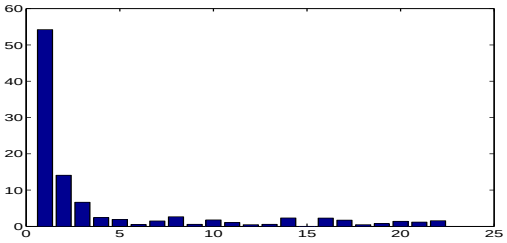
(c) Syllable length (**L**)



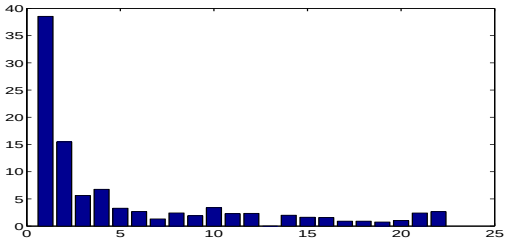
(d) Syllable frequency (**F**)



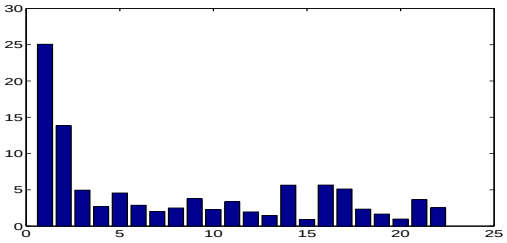
(e) Syllable energy contour (**E**)



(f) Syllable energy contour ($\hat{\mathbf{E}}$)



(g) TES of syllables (**T**)



(h) Maximal amplitude of syllables (**A**)

Figure A.4: Evaluation of features extracted from 3 consecutive syllables of katydid species (TS4). Whereas the first bin of the x-axis shows the classification accuracy, the other bins show the classification rate for the individual ranks. The number of ranks is given by the number of classes. For more details about the feature evaluation see Section 4.4.

Appendix B

Feature Selection

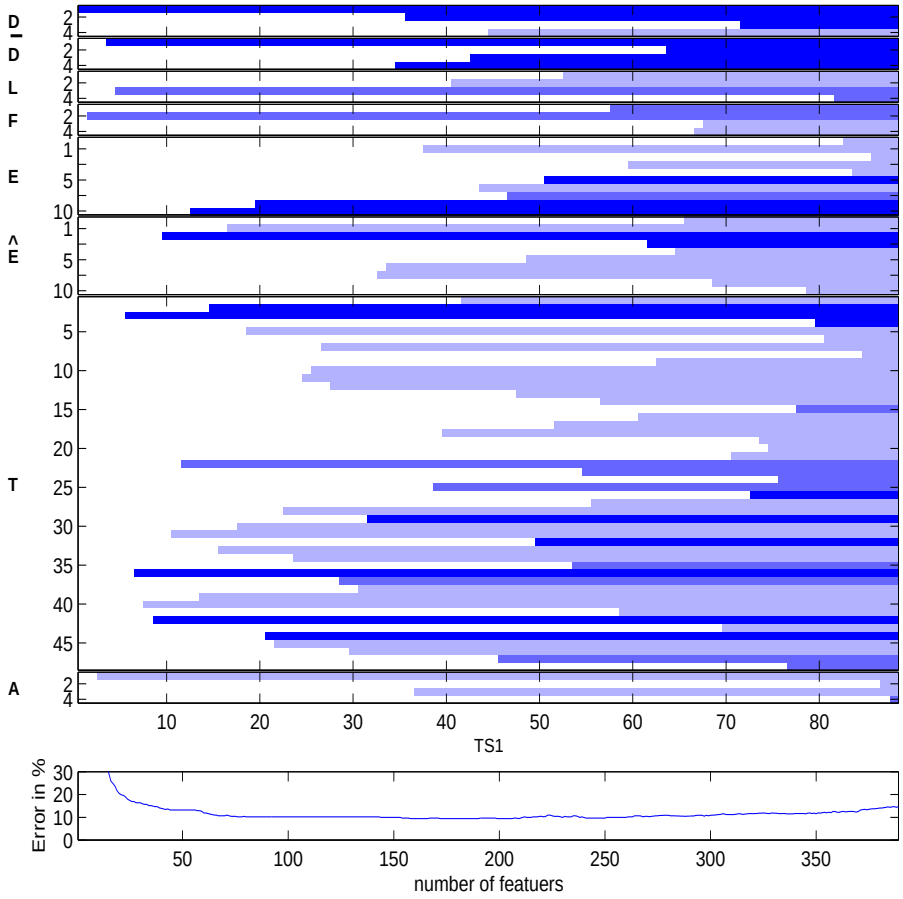


Figure B.1: Feature selection for katydids to species level with SFSM. The Figure shows automatically selected feature components of time scale 1 for each step of the feature selection procedure. Furthermore, the classification error for the combination of four time scales is given for each step of the feature selection procedure. Three different brightness values have used, each of them is assigned to a specific classifier. For more details see Section 4.5.

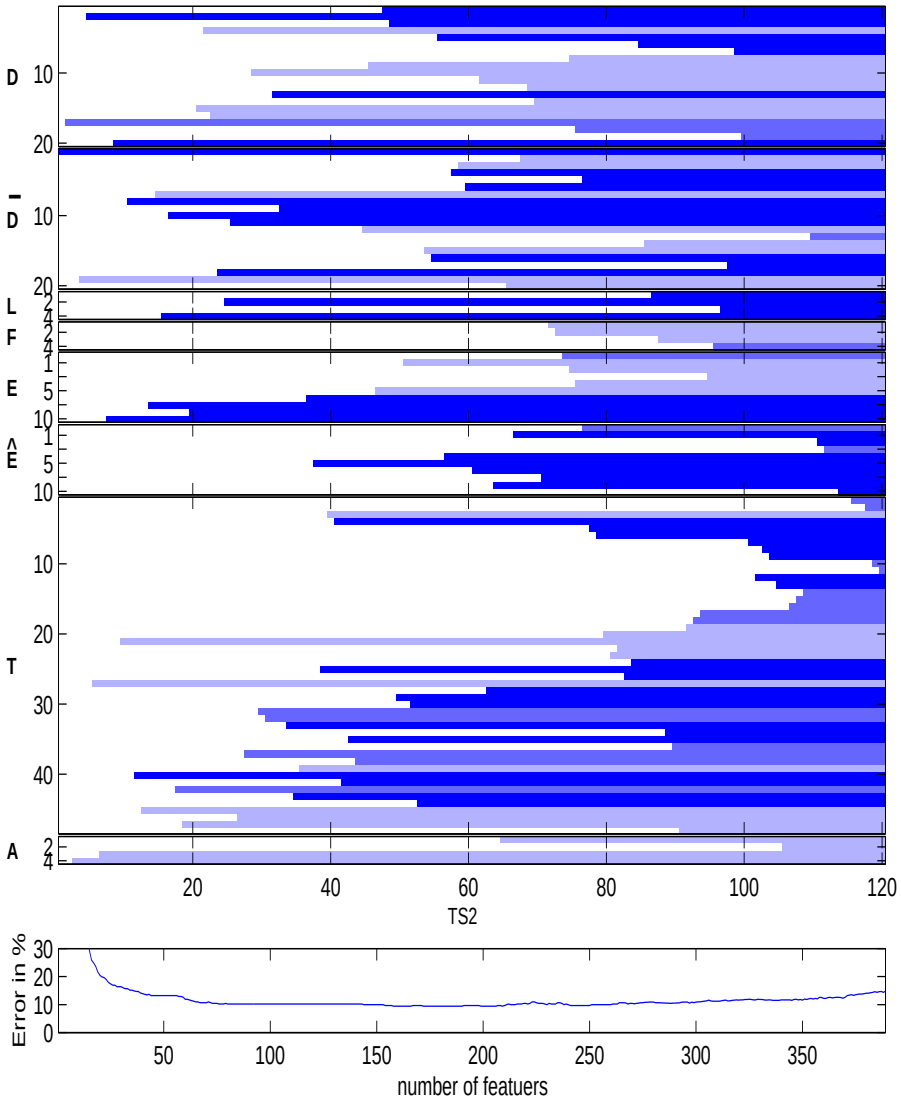


Figure B.2: Feature selection for katydids to species level with SFSM. The Figure shows automatically selected feature components of time scale 2 for each step of the feature selection procedure. Furthermore, the classification error for the combination of four time scales is given for each step of the feature selection procedure. Three different brightness values have used, each of them is assigned to a specific classifier. For more details see Section 4.5.

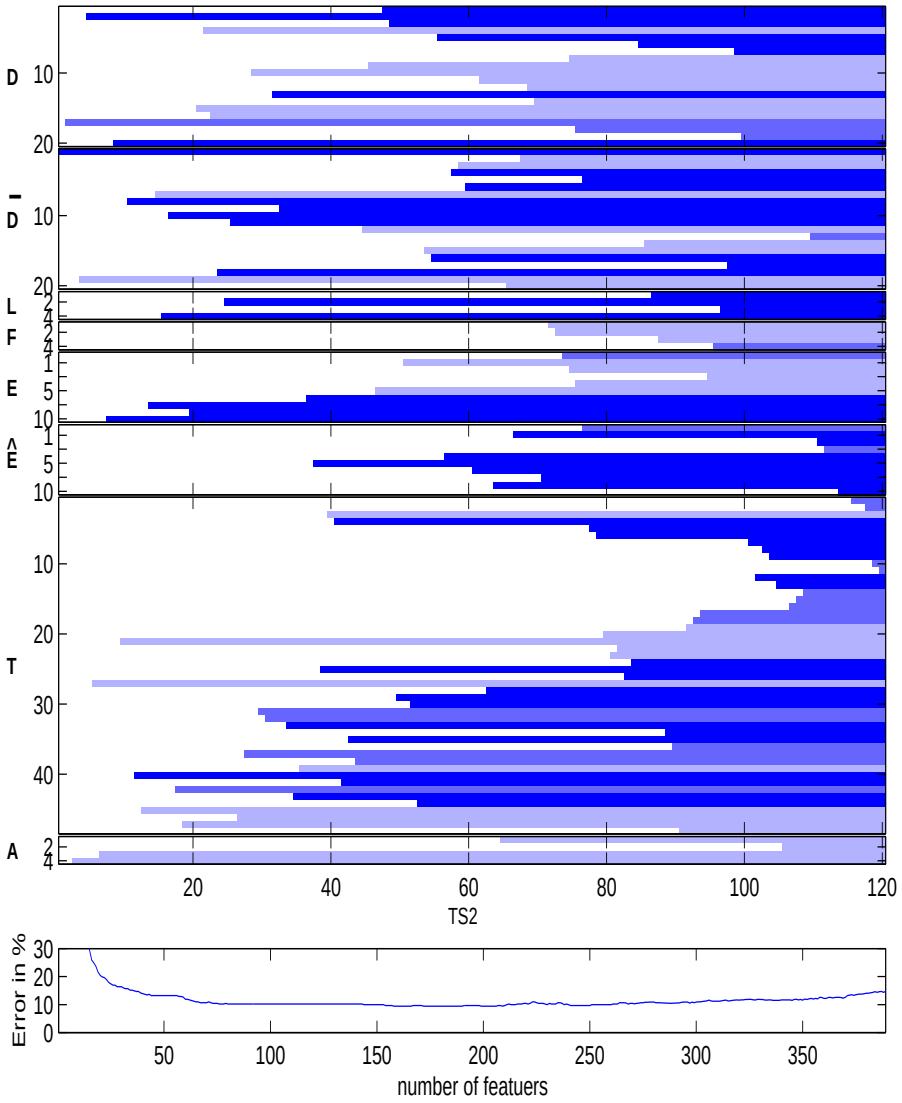


Figure B.3: Feature selection for katydids to species level with SFSM. The Figure shows automatically selected feature components of time scale 3 for each step of the feature selection procedure. Furthermore, the classification error for the combination of four time scales is given for each step of the feature selection procedure. Three different brightness values have used, each of them is assigned to a specific classifier. For more details see Section 4.5.

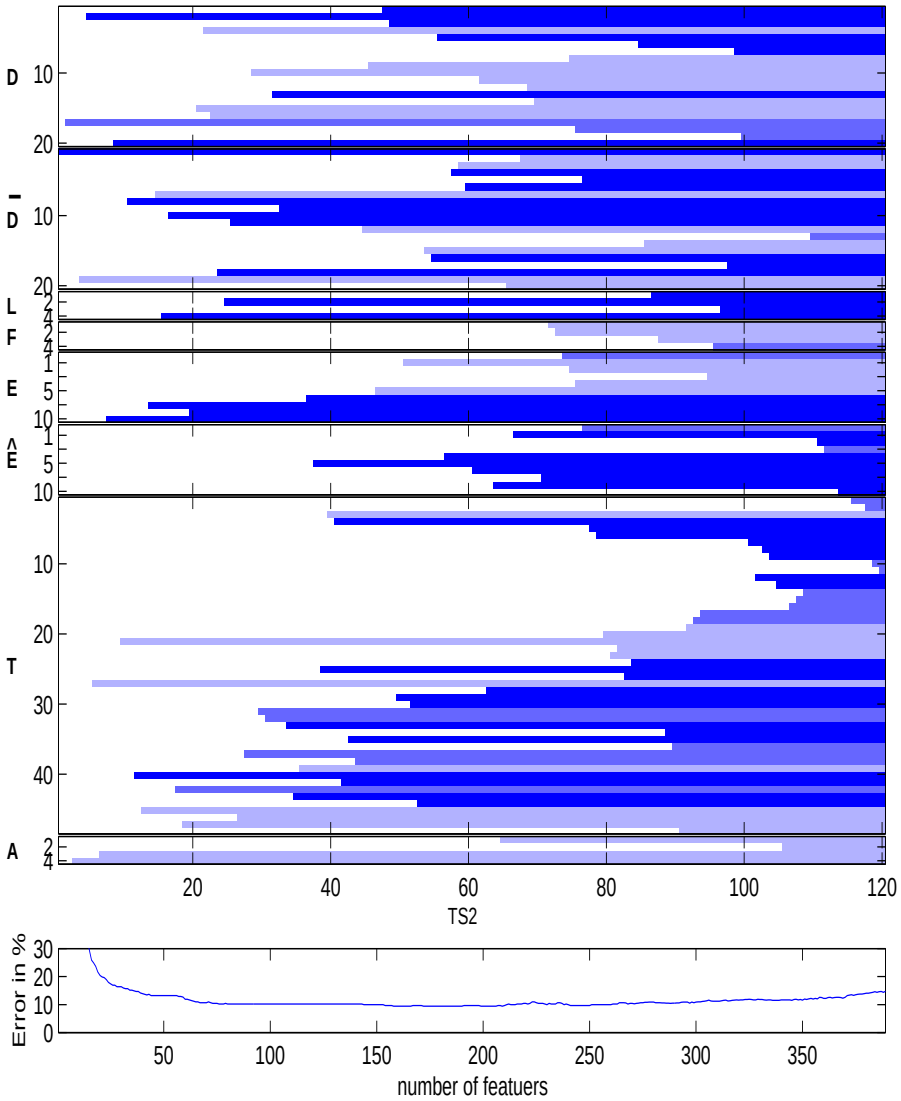


Figure B.4: Feature selection for katydids to species level with SFSM. The Figure shows automatically selected feature components of time scale 4 for each step of the feature selection procedure. Furthermore, the classification error for the combination of four time scales is given for each step of the feature selection procedure. Three different brightness values have used, each of them is assigned to a specific classifier. For more details see Section 4.5.

Appendix C

Parameters

Algorithm	*crickets / katydids*
Pulse segmentation	U = 0.454 ms (20 samples)
	V = 7.256 ms (320 samples)
	W = 22.67 ms (1000 samples)
	ψ = 2.267 ms (100 samples)
	α = 2.00
	γ = 0.15
	ξ = 42
	ζ = 0.15
	κ_{min} = 2
	κ_{max} = 0
Filter-bank analysis	V = 500 ms (22050 samples)
	β = 1
	f_1 = 3000 Hz
	f_D = 19000 Hz
	b = 2000 Hz
	D = 10
	J = 4
Pulses	Λ = 5
Pulse distances	D = 4
	$p_{20}^{\mathbf{D}}$ = 0.03
Pulse length	D = 5
	$p_{20}^{\mathbf{L}}$ = 0.01

Table C.1: The parameters for pulse segmentation (upper part, Section 4.2.4) and feature extraction (lower part, Section 4.2.5) from *Orthoptera* sounds. The parameter Λ determines the number of pulses inside a single sliding window (see Figure 4.16).

Algorithm	*crickets*
Filtering	f_{max} = 7300, f_σ = 2830
Pulse segmentation	U = 2.7 ms (80 samples)
	V = 29.02 ms (1280 samples)
	W = 22.67 ms (1000 samples)
	ψ = 2.267 ms (100 samples)
	α = 2.30
	γ = 0.15
	ξ = 62
	ζ = 2.00
	κ_{min} = 2
	κ_{max} = 0
Pulses	Λ = 5
Pulse distances	D = 4
	$p_{20}^{\mathbf{D}}$ = 0.00
Pulse length	D = 4
	$p_{20}^{\mathbf{L}}$ = 0.00
Frequency contour	V = 1.36 ms
	α = 0.5
	D = 5
	G = 1
	ϕ = 5
	σ = 0
Pulse frequency	D' = 4
Energy contour 1	V = 1.36 ms
	α = 0.5
	D = 20
	G = 1
Energy contour 2	V = 1.36 ms
	α = 0.1
	β = 1.2
	D = 20
	G = 1
Time encoded signals	$\mathbf{r}_1 = (0, 1, ..., 4)$
	$\mathbf{r}_2 = (\frac{0}{20}, \frac{1}{20}, ..., \frac{12}{20})$
	G = 4

Table C.2: The parameters for pulse segmentation (upper part, Section 4.2.4) and feature extraction (lower part, Section 4.2.5) from cricket sounds. The parameter Λ determines the number of pulses inside a single sliding window (see Figure 4.16).

Algorithm	*katydids*			
Time scale	TS1	TS2	TS3	TS4
Filtering	f_{max} = 15000, f_σ = 5660			
P. segm.	U = 0.227 ms	U = 0.227 ms	U = 3.17 ms	U = 9.07 ms
	V = 36.28 ms	V = 36.28 ms	V = 50.79 ms	V = 145.1 ms
	W = 22.7 ms	W = 22.7 ms	W = 136.0 ms	W = 136.0 ms
	ψ = 2.267 ms	ψ = 2.267 ms	ψ = 2.267 ms	ψ = 2.267 ms
	α = 2.00	α = 2.00	α = 2.15	α = 1.90
	γ = 0.15	γ = 0.15	γ = 0.05	γ = 0.03
	ξ = 42	ξ = 42	ξ = 62	ξ = 200
	ζ = 0.15	ζ = 0.15	ζ = 0.12	ζ = 0.17
	κ_{min} = 2	κ_{min} = 2	κ_{min} = 2	κ_{min} = 2
	κ_{max} = 0	κ_{max} = 0	κ_{max} = 0	κ_{max} = 0
Pulses	Λ = 5	Λ = 21	Λ = 5	Λ = 4
P. distances	D = 4	D = 20	D = 4	D = 3
	$p_{20}^{\mathbf{D}}$ = 0.01	$p_{20}^{\mathbf{D}}$ = 0.01	$p_{20}^{\mathbf{D}}$ = 0.03	$p_{20}^{\mathbf{D}}$ = 0.03
P. length	D = 4	D = 4	D = 4	D = 3
	$p_{20}^{\mathbf{L}}$ = 0.01	$p_{20}^{\mathbf{L}}$ = 0.01	$p_{20}^{\mathbf{L}}$ = 0.03	$p_{20}^{\mathbf{L}}$ = 0.03
Frequency c.	V = 1.36 ms	V = 1.36 ms	V = 1.36 ms	V = 1.36 ms
	α = 0.5	α = 0.5	α = 0.5	α = 0.5
	D = 5	D = 5	D = 5	D = 5
	G = 1	G = 1	G = 1	G = 1
	ϕ = 5	ϕ = 5	ϕ = 5	ϕ = 5
	σ = 0	σ = 0	σ = 0	σ = 0
P. frequency	D' = 4	D' = 4	D' = 4	D' = 3
Energy c. 1	V = 0.45 ms	V = 0.45 ms	V = 3.17 ms	V = 9.07 ms
	α = 0.5	α = 0.5	α = 0.5	α = 0.5
	D = 10	D = 10	D = 10	D = 15
	G = 1	G = 1	G = 1	G = 1
Energy c. 2	V = 0.45 ms	V = 0.45 ms	V = 3.17 ms	V = 9.07 ms
	α = 0.1	α = 0.1	α = 0.1	α = 0.1
	β = 0.8	β = 0.8	β = 0.8	β = 0.8
	D = 10	D = 10	D = 10	D = 15
	G = 1	G = 1	G = 1	G = 1
TES	$\mathbf{r}_1 = (k)_{k=0}^{4}$	$\mathbf{r}_1 = (k)_{k=0}^{4}$	$\mathbf{r}_1 = (k)_{k=0}^{4}$	$\mathbf{r}_1 = (k)_{k=0}^{4}$
	$\mathbf{r}_2 = (\frac{k}{20})_{k=0}^{12}$	$\mathbf{r}_2 = (\frac{k}{20})_{k=0}^{12}$	$\mathbf{r}_2 = (\frac{k}{20})_{k=0}^{12}$	$\mathbf{r}_2 = (\frac{k}{20})_{k=0}^{12}$
	G = 4	G = 4	G = 4	G = 3

Table C.3: The parameters for pulse segmentation (upper part, Section 4.2.4) and feature extraction (lower part, Section 4.2.5) from katydid sounds. The parameter Λ determines the number of pulses inside a single sliding window (see Figure 4.16).

Appendix D

Species of the Dataset

nr.	species	subfamily	rec.
1.	Conocephalus discolor	Conocephalinae	12
2.	Barbitistes ocskayi	Phaneropterinae	7
3.	Ephippiger ephippiger	Bradyporinae	12
4.	Ephippiger perforatus	Bradyporinae	6
5.	Eupholidoptera chabrieri	Tettigoniinae	6
6.	Isophya kraussii	Phaneropterinae	8
7.	Isophya modestior	Phaneropterinae	8
8.	Metrioptera bicolor	Tettigoniinae	11
9.	Metrioptera brachyptera	Tettigoniinae	10
10.	Pholidoptera fallax	Tettigoniinae	10
11.	Pholidoptera macedonica	Tettigoniinae	10
12.	Platycleis affinis	Tettigoniinae	9
13.	Platycleis albopunctata	Tettigoniinae	12
14.	Platycleis intermedia	Tettigoniinae	9
15.	Poecilimon artedentatus	Phaneropterinae	6
16.	Poecilimon chopardi	Phaneropterinae	9
17.	Poecilimon nobilis	Phaneropterinae	10
18.	Poecilimon thessalicus	Phaneropterinae	12
19.	Pterolepis germanica	Tettigoniinae	5
20.	Sepiana sepium	Tettigoniinae	6
21.	Tettigonia cantans	Tettigoniinae	7
22.	Tettigonia viridissima	Tettigoniinae	12

Table D.1: The katydid species of the data set including the subfamily and the number of recordings per species. All recordings have been made in Europe.

nr.	species	subfamily	rec.	locality
1.	Anaxipha tachephona	Trigonidiinae	12	EC
2.	Acheta domestica	Gryllinae	4	EU
3.	Aclodes chamocoru	Phalangopsinae	12	EC
4.	Cryptacla clandestina	Phalangopsinae	7	EC
5.	eneopsp1	Eneopterinae	3	EC
6.	eneopsp2	Eneopterinae	5	EC
7.	Gryllotalpa ...	Gryllotalpidae	5	IN/TH
8.	Gryllus bimaculatus	Gryllinae	4	EU
9.	Gryllus campestris	Gryllinae	8	EU
10.	Gymnogryllus angustus	Gymnogryllini	3	IN
11.	Homoeoxipha lycoides	Trigoniini	4	TH
12.	Loxoblemmus parabolicus	Modicogryllini	7	IN/MA
13.	Mogoplistinae DS1	Mogoplistinae	4	TH
14.	nemosp1	Gryllinae	6	EC
15.	Oecanthus pellucens	Oecanthinae	9	EU
16.	Paraclodes cryptos	Phalangopsinae	6	EC
17.	Pteronemobius heydeni c.	Nemobiinae	4	EU
18.	Teleogryllus mitratus	Gryllinae	6	TH
19.	Teleogryllus oceanicus	Gryllinae	4	EU/TH
20.	triglp1	Trigonidiinae	3	EC
21.	trigsp03 (high pulse rate)	Trigonidiinae	4	EC
22.	trigsp03 (low pulse rate)	Trigonidiinae	4	EC
23.	Trigsp04	Trigonidiinae	8	EC
24.	Trigsp06	Trigonidiinae	9	EC
25.	Trigsp09	Trigonidiinae	4	EC
26.	Trigsp11	Trigonidiinae	4	EC
27.	Trigsp15	Trigonidiinae	9	EC
28.	Trigsp16	Trigonidiinae	12	EC
29.	Trigsp17	Trigonidiinae	4	EC
30.	Velarifictorus aspersus	Modicogryllinae	6	IN/TH
31.	Xenogryllus transversus	Eneopterinae	6	TH

Table D.2: The cricket species of the data set including the subfamily, the number of recordings per species and the locality (EU Europe, EC Ecuador, IN Indonesia, MA Malaysia and TH Thailand).

Bibliography

[BC96] H. C. Bennet-Clark. Songs and the physics of sound production. in cricket behaviour and neurobiology. Technical report, Ithaca, London, 1996.

[BD87] P. J. Browell and R. A. Davis. *Time series: Theory and Methods*. Springer, New York, 1987.

[Bel57] R. Bellmann. *Dynamic Programming*. Princeton University Press, 1957.

[Bis95] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[BJ92] R. Beale and T. Jackson. *Neuronal Computing*. IOP Publishing Inc., University of York, 1992.

[BL88] D.S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.

[BPSW70] L. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occuring in the statistical analysis of probabilistic functions of markov chains. *Ann. Math Statist.*, 41:164–171, 1970.

[Bra95] R. Brause. *Neuronale Netze*. Teubner, Frankfurt/Main, 1995.

[Bre96] L. Breiman. Bagging predictors. *Machine Learning*, 26(2):123–140, 1996.

[Bro63] W. B. Broughton. Method in bio-acoustic terminology. *Acoustic behaviour of animals*, pages 3–24, 1963.

[BS01] J. Bigun and F. Smeraldi. *Audio- and Video-Based Biometric Person Authentication*. Springer, 2001.

[CC77] T. M. Cover and J. M. Van Campenhout. On the possible orderings in the measurement selection problem. *IEEE Transactions on Systems Man and Cybernetics*, SMC-7:657–661, 1977.

[CC00] D. Chen and X. Cheng. Majority vote based on weak classifiers. In *Proceedings of the 5th International Conference on Signal Processing*, pages 1560–1562, 2000.

[CCG91] S. Chen, C. F. N. Cowan, and P. M. Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2(2):302–309, 1991.

[CF97] C. Carlsson and R. Fuller. Owa operators for decision support. In *Proceedings of EUFIT'97 Conference*, 1997.

[CFS98] E. D. Chesmore, O. P. Femminella, and M. D. Swarbrick. Automated analysis of insect sounds using time-encoded signals - a new method for species identification. *CAB International*, pages 273–287, 1998.

[CK95] S-B. Cho and J. H. Kim. Combining multiple neural networks by fuzzy integral for robust classification. *IEEE Transactions on Systems Man and Cybernetics*, 25(2):380–384, 1995.

[CKHP02] S. Chu, E. Keogh, D. Hart, and M. Pazzani. Iterative deepening dynamic time warping for time series. In *Proceedings of the Second SIAM International Conference on Data Mining*, 2002.

[CS96] M. J. Creaney-Stockton. *Isolated Word Recognition Using Reduced Connectivity Neural Networks With Non-Linear Time Alignment Methods*. PhD thesis, University of Newcastle-Upon-Tyne, 1996.

[CXC96] K. Chen, D. Xie, and H. Chi. Speaker identification using time-delay hmes. *International Journal of Neural Systems*, 7(1):29–43, 1996.

[Dau01] J. Daugman. Personal identification in real-time by wavelet analysis of iris patterns. In Yuan Yan Tang, M. Victor Wickerhauser, Pong Chi Yuen, and Chun Hung Li, editors, *WAA*, page 1. Springer, 2001.

[DG95] M. Dambach and A. Gras. Bioacoustics of a miniature cricket, cycloptiloides canariensis (orthoptera:gryllidae:mogoplistinae). *Journal of Experimental Biology*, 198:721–728, 1995.

[DG01] J. J. R. Diez and C. J. A. González. Learning classification rbf networks by boosting. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 43–52. Springer, 2001.

[DH73] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, 1973.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, 2000.

[DPRSnt] C. Dietrich, G. Palm, K. Riede, and F. Schwenker. Classification of bioacoustic time series based on the combination of global and local decisions. *Pattern Recognition*, in print.

[DPS03] C. Dietrich, G. Palm, and F. Schwenker. Decision templates for the classification of time series. *International Journal of Information Fusion*, 4:101–109, 2003.

[DSP01a] C. Dietrich, F. Schwenker, and G. Palm. Classification of time series utilizing temporal and decision fusion. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 378–387. Springer, 2001.

[DSP01b] C. Dietrich, F. Schwenker, and G. Palm. Fusion architectures for the classification of time series. In G. Duffner, H. Bischof, and K. Hornik, editors, *Artificial Neural Networks - ICANN 2001*, pages 749–755. Springer, 2001.

[DSP02] C. Dietrich, F. Schwenker, and G. Palm. Decision templates for the classification of bioacoustic time series. In H. Bourland et. al., editor, *Proceedings of IEEE Workshop on Neural Networks and Signal Processing*, pages 159–168, 2002.

[DSP03a] C. Dietrich, F. Schwenker, and G. Palm. Classification of bioacoustic time series by training a fusion layer with decision templates. In Marco Gori and Simone Marinai, editors, *Proceedings of the IAPR-TC3*, pages 15–18. University of Florence, 2003.

[DSP03b] C. Dietrich, F. Schwenker, and G. Palm. Multiple classifier systems for the recognition of orthoptera songs. In B. Michaelis and G. Krell, editors, *Proceedings of the DAGM*, pages 474–481. Springer, 2003.

[DSRP01] C. Dietrich, F. Schwenker, K. Riede, and G. Palm. Classification of bioacoustic time series utilizing pulse detection, time and frequency features and data fusion. *www.informatik.uni-ulm.de/pw/berichte* Ulmer Informatik-Berichte 2001-04, University of Ulm, 2001.

[DT00] R. P. W. Duin and D. M. J. Tax. Experiments with classifier combining rules. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 16–29. Springer, 2000.

[Dui90] M. Duijm. On some characteristics in ephippiger (orthoptera, tettigoniidae) and their geographic variation. *Netherl. J. Zool.*, 40:428–453, 1990.

[EE94] M. R. Evans and J. A. Evans. A computer-based technique for the quantitative analysis of animal sounds. *International Journal of Animal Sound Recording*, 5:281–290, 1994.

[EK85] C. J. H. Elliott and U. T. Koch. The clockwork cricket. *Naturwissenschaften*, 72:150–153, 1985.

[Elm90] J. L. Elman. Finding structure in time. *CognitiveScience*, 14:179–211, 1990.

[End01] S. Enderle. *Probabilistic Map Building and Self-Localization for Autonomous Mobile Robots*. PhD thesis, University of Ulm, 2001.

[EP99] N. Euliano and J. Principe. A spatio-temporal memory based on soms with activity diffusion. In *Workshop on Self-Organizing Maps*, pages 253–266. Helsinki, Finland, 1999.

[Ewi89] A. W. Ewig. Arthropod bioacoustics - neurobiology and behaviour. Technical report, Edinburgh University Press, 1989.

[FHT98] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Technical Report Technical Report, Dept. of Statistics, Stanford University, 1998.

[Fre95] Y. Freud. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[Fuk75] K. Fukushima. Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics*, 20:121–136, 1975.

[Fuk90] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, New York, 1990.

[GB03] S. Günter and H. Bunke. Fast feature selection in an hmm-based multiple classifier system for handwriting recognition. In B. Michaelis and G. Krell, editors, *Proceedings of the DAGM*, pages 289–296. Springer, 2003.

[GBD92] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4:1–58, 1992.

[GD00] C. J. A. González and J. J. R. Diez. Boosting intervall-based time series classifiers. *Inteligencia Artificial*, 11:2–11, 2000.

[Geu02] P. Geurts. *Contributions to Decision Tree Induction*. PhD thesis, Université de Liége, 2002.

[GGF99] H. G. Göckler, A. Groth, and C. M. Flatten. Efficient multirate digital filters based on fractional polyphase decomposition for subnyquist processing. In *Proceedings of the ECCTD*, pages 409–412. Stresa, 1999.

[GH02] H. C. Gerhardt and F. Huber. *Acoustic Communication in Insects and Anurans*. University of Chicago Press, Chicago, 2002.

[Gla96] M. I. Glavinovic. Comparison of parzen density and frequency histogram as estimators of probability density functions. *European Journal of Physiology*, 433(1):174–179, 1996.

[GR00] G. Giacinto and F. Roli. Dynamic classifier selection. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 177–189. Springer, 2000.

[GS91] S. Geva and J. Sitte. Adaptive nearest neighbour pattern classification. *IEEE Transactions on Neural Networks*, 2(2):318–322, 1991.

[GTCW96] G. Grigg, A. Taylor, H. Mc Callum, and G. Watson. Monitoring frog communities: An application of machine learning. In *Proceedings of Eighth Innovative Applications of Artificial Intelligence Conference, Portland Oregon*, pages 1564–1569. AAAI Press, 1996.

[GVB+92] I. Guyon, V. Vapnik, B. Boser, L. Bottou, and S.A. Solla. Capacity control in linear classifiers for pattern recognition. In *11th International Conference on Pattern Recognition*, pages 385–388, 1992.

[HAK01] D. J. Hand, N. M. Adams, and M. G. Kelly. Multiple classifier systems based on interpretable linear classifiers. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 136–147. Springer, 2001.

[Hal99] Mark A. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, University of Waikato, 1999.

[Ham77] R. W. Hamming. *Digital Filters*. Prentice Hall, Englewood Cliffs, N.J., 1977.

[Hau94] A. Hausmann. *Statistische Klassifizierung in hochdimensionalen Merkmalsräumen - Ein paralleler lokaler Klassifikator*. PhD thesis, University of Ulm, 1994.

[Hay94] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Macmillan College Publishing Company, 1994.

[Hel88] K. G. Heller. *Bioakustik der Europäischen Laubheuschrecken. Ökologie in Forschung und Anwendung.* Margraf, Weikersheim, 1988.

[Her02] R. Herbrich. *Learning Kernel Classifiers: Theory and Algorithms*. MIT Press, London, England, 2002.

[HHK86] J. Holbeche, R. D. Hughes, and R. A. King. Time encoded speech (tes) descriptors as a symbol feature set for voice recognition. In *IEEE Int. Conf. Speech Input/Output: Techniques and Applications*, 1986.

[HHM99] D. S. Houser, D. A. Helweg, and P. W. Moore. Classification of dolphin echolocation clicks by energy and frequency distributions. *Journal of the Acoustical Society of America*, 106(3):1579–1585, 1999.

[HHS92] T. K. Ho, J. J. Hull, and S. N. Srihari. On multiple classifier systems for pattern recognition. In *Proceedings of the 11th International Conference on Pattern Recognition*, pages 84–87, 1992.

[HKO01] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, New York, 2001.

[HKP91] J A. Hertz, A Krogh, and R G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[HML89] F. Huber, T. E. Moore, and W. Loher. *Cricket Behavior and Neurobiology*. Cornell University Press, Ithaca, NY, 1989.

[HMS66] E. B. Hunt, J. Marin, and P. J. Stone. *Experiments in Induction*. Academic Press, 1966.

[Ho98] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[Ho01] T. K. Ho. Data complexity analysis for classifier combination. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 53–63. Springer, 2001.

[HS90] L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:993–1001, 1990.

[HS95] Y. S. Huang and C. Y. Suen. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:90–93, 1995.

[HTS00] Jaakko Hollmen, Volker Tresp, and Olli Simula. A learning vector quantization algorithm for probabilistic models. In *Proceedings of EUSIPCO - European Signal Processing Conference*, pages 721–724, 2000.

[III98] E. Mercado III. *Humpback Whale Bioacoustics From Form to Function*. PhD thesis, University of Hawaii, 1998.

[IK98] S. Ingrisch and G. Koehler. *Die Heuschrecken Mitteleuropas*. Westarp Wissenschaften, Magdeburg (in German), 1998.

[Ing97] S. Ingrisch. Taxonomy, stridulation and development of Podoscirtinae from Thailand. *Senckenbergiana biologica*, 77:47–75, 1997.

[IRD01] S. Ingrisch, K. Riede, and C. Dietrich. Dorsa - german orthoptera collections. In *BIOLOG. German programme on biodiversity and global change*, pages 200–201, 2001.

[Jóź83] A. Jóźwik. A learning scheme for a fuzzy k-nn rule. *Pattern Recognition Letters*, 1:287–289, 1983.

[Jat95] M. Jatho. *Untersuchungen zur Schallproduktion und zum phonotaktischen Verhalten von Laubheuschrecken Orthoptera: Tettigoniidae*. PhD thesis, University of Marburg in Germany, 1995. ISBN 3-89588-292-5.

[Jay96] E. T. Jaynes. *Probability Theory: The Logic of Science*. Washington University, St. Louis, MO 63130, U.S.A., 1996.

[JJNH91] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[Jor86] M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eight Ann. Conf. of the Cognitive Science Society*, pages 531–546, 1986.

[JSSK94] M. Jatho, J. Schul, O. Stiedl, and K. Kalmring. Specific differences in sound production and pattern recognition in tettigoniids. *Behavioural Processes*, 31:293–300, 1994.

[JWK92] M. Jatho, S. Weidemann, and D. Kretzen. Species-specific sound production in three ephippigerine bushcrickets. *Behavioural Processes*, 26:31–42, 1992.

[Kad02] M. W. Kadous. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series*. PhD thesis, University of New South Wales, 2002.

[KB98a] L. I. Kuncheva and J. C. Bezdek. An integrated framework for generalized nearest prototype classifier design. *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, 6(5):437–457, 1998.

[KB98b] L. I. Kuncheva and J. C. Bezdek. An integrated framework for generalized nearest prototype classifier design. *Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(5):437–457, 1998.

[KBD01] L. I. Kuncheva, J. C. Bezdek, and R. P. W. Duin. Decision templates for multiple classifier fusion. *Pattern Recognition*, 34(2):299–314, 2001.

[KBS98] L. I. Kuncheva, J. C. Bezdek, and M. A. Sutton. On combining multiple classifiers by fuzzy templates. *North American Fuzzy Information Procecsing Society*, pages 193–197, 1998.

[KESK87] U. T. Koch, C. J. H. Elliott, K-H. Schäffner, and H-U. Kleindienst. The mechanics of the stridulation of the cricket gryllus campestris. *Journal of Comparative Physiology*, 162:213–223, 1987.

[KG78] R. A. King and W. Gosling. Time-encoded speech. *Electron. Lett.*, 14(15):456–457, 1978.

[KHDM98] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[KHK+96] T. Kohonen, J. Hynninen, J. Kangas, J. Laaksonen, and K. Tork ko la. Lvq-pak: The learning vector quantization program package. Technical Report A30, Helsinky University of Technology, 1996.

[KJ00] L. I. Kuncheva and L. C. Jain. Designing classifier fusion systems by genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4(4):327–336, 2000.

[KKK90] K. Kalmring, K. Keuper, and W. Kaiser. Aspects of acoustic and vibratory communication in seven european bushcrickets. In W. J. Bailey and D. C. F. Rentz, editors, *The Tettigoniidae: Biology, Systematics and Evolution*, pages 191–216. Springer-Verlag, Berlin, 1990.

[KKZ95] L. I. Kuncheva, R. K. Kounchev, and R. Z. Zlatev. Aggregation of multiple classification decisions via fuzzy templates. In *Proc. III-d European Congress on Intelligent Techniques and Soft Computing*, pages 1470–1474, 1995.

[KM98] J. A. Kogan and D. Margoliash. Automated recognition of bird song elements from continous recordings using dynamic time warping and hidden markov models: A comparative study. *Journal of the Acoustical Society of America*, 103(4):2185–2196, 1998.

[KM02] J. Kittler and K. Messer. Fusion of multiple experts in multimodal biometric personal identity verification systems. In

H. Bourland et. al., editor, *Proceedings of IEE Workshop on Neural Networks and Signal Processing*, pages 3–12, 2002.

[Koh90] T. Kohonen. Improved versions of learning vector quantization. In *Proceedings of the International Joint Conference on Neural Networks*, pages I 545–550. San Diego, 1990.

[Koh95] T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, 1995.

[KP98] R. A. King and T. C. Phipps. Shannon, tespar and approximation strategies. In *9th International Conference on Signal Processing Applications and Technology, Toronto*, 1998.

[Kub98] M. Kubat. Decision trees can initialize radial-basis-function networks. *IEEE Transactions on Neural Networks*, 9:813–821, 1998.

[Kun98] L. I. Kuncheva. On combining multiple classifiers. In *Proceedings of the 7th International Conference on Information Processing and Management Uncertainty, Paris, France*, pages 1890–1891, 1998.

[Kun00a] L. I. Kuncheva. Clustering and selection model for classifier combination. In *4th International Conference on Knowledge-Based Intelligent Engineering Systems*, 2000.

[Kun00b] L. I. Kuncheva. *Fuzzy Classifier Design*. Physica Verlag, Heidelberg New York, 2000.

[Kun01] L. I. Kuncheva. Using measures of similarity and inclusion for multiple classifier fusion by decision templates. *Fuzzy Sets and Systems*, 122(3):401–407, 2001.

[KW01a] L. I. Kuncheva and C. J. Whitaker. Feature subsets for classifier combination: An enumerative experiment. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 228–237. Springer, 2001.

[KW01b] L. I. Kuncheva and C. J. Whitaker. Ten measures of diversity in classifier ensembles: Limits for two classifiers. In *IEEE Workshop on Intelligent Sensor Processing*, pages 1–10. Birmingham, 2001.

[KWSD00] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. W. Duin. Is independence good for combining classifiers? In *Proc. International Conference on Pattern Recognition*, pages 168–171, 2000.

[LW79] R. E. Love and T. J. Walker. Systematics and acoustic behavior of scaly crickets (orthoptera: Gryllidae: *Mogoplistinae*) of eastern united states. *Trans. Am. Entomol. Soc.*, 105:1–66, 1979.

[MD89] J. Moody and C. J. Darken. Fast larning in networks of locally-tuned processing units. *Neural Computation*, 1:284–294, 1989.

[Mel00] P. Poirazi B. W. Mel. Choice and value flexibility jointly contribute to the capacity of a subsampled quadratic classifier. *Neural Computation*, 12(5):1189–1205, 2000.

[MH88] J. A. Marshall and E. C. M. Haes. *Grasshoppers and Allied Insects of Great Britain and Ireland*. Harley Books, UK, 1988.

[MK98] E. Mercado and A. Kuh. Classification of humpback whale vocalizations using self-organizing neural networks. In *International Joint Conference on Neural Networks*, pages 1584–1589, 1998.

[MLR01] P. J. Moreno, B. Logan, and B. Raj. A boosting approach for confidence scoring. Technical Report CRL 2001/08, Cambridge Research Laboratory, 2001.

[MMH02] A.C. Mason, G. K. Morris, and R. R. Hoy. Tuning in by turning off. *Journal of Comparative Physiology*, 418:831–833, 2002.

[MMR98a] S. O. Murray, E. Mercado, and H. L. Roitblat. Characterizing the graded structure of false killer whale *(pseudorca crassidens)* vocalizations. *Journal of the Acoustical Society of America*, 104(3):1679–1682, 1998.

[MMR98b] S. O. Murray, E. Mercado, and H. L. Roitblat. The neural network classification of false killer whale *(pseudorca crassidens)* vocalizations. *Journal of the Acoustical Society of America*, 104(6):3626–3633, 1998.

[Moz93] M. C. Mozer. Neural net architectures for temporal sequence processing. In A. S. Weigend and N. A. Gershenfeld, editors, *Time series prediction: Forecasting the future and understanding the past*, volume 15, pages 243–264. Addison Wesley, Reading, MA, 1993.

[MP43] W.S. McCulloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[MPS+01] C. K. Machens, P. Prinz, M. B. Stemmler, B. Ronacher, and A. V. M. Herz. Discrimination of behaviorally relevant signals by auditory receptor neurons. *Neurocomputing*, 38-40:263–268, 2001.

[MSC91] D. P. Morgan, C. L. Scofield, and L. N. Cooper. *Neural Networks and Speech Processing*. Kluwer Academic Publishers, Boston, Dordrecht, London, 1991.

[MSP+01] C. K. Machens, M. B. Stemmler, P. Prinz, R. Krahe, B. Ronacher, and A. V. M. Herz. Representation of acoustic communication signals by insect auditory receptor neurons. *Neuroscience*, 21(9):3215–3227, 2001.

[Nis99] F. Nischk. *Die Grillengesellschaften zweier neotropischer Waldökosysteme in Ecuador*. PhD thesis, University of Köln, Germany, 1999. Memorandum UCB/ERL–M89/29 (in German).

[NW01] M. Naguib and R. H. Wiley. Estimating the distance to a source of sound: Mechanisms and adaptations for long-range communication. *Animal Behaviour*, 62:825–837, 2001.

[Odo97] R. Odorico. Learning vector quantization with training count (LVQTC). *Neural Networks*, 10(6):1083–1088, 1997.

[Orr98] M. J. L. Orr. An em algorithm for regularised rbf networks. In *International Conference on Neural Networks and Brain*. China, 1998.

[Orr99] M. J. L. Orr. Combining Regression Trees and Radial Basis Function Networks. Technical report, Institute for Adaptive and Neuronal Computation, Devision Of Informatiks, Edinburgh University, 1999.

[OT01] N. C. Oza and K. Tumer. Input decimation ensembles: Decorrelation through dimensionality reduction. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 238–247. Springer, 2001.

[Ott92] D. Otte. Evolution of cricket songs. *Journal of Orthopteran Research*, I:25–44, 1992.

[Pal80] G. Palm. On associative memory. *Biological Cybernetics*, 36:19–31, 1980.

[Par62] E. Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[Par01a] J. R. Parker. Rank and response combination from confusion matrix data. *Journal of Information Fusion*, 2(2):113–120, 2001.

[Par01b] S. Parsons. Identification of new zealand bats (chalinolobus tuberculatus and mystacina tuberculata) in flight from analysis of echolocation calls by artificial neural networks. *Journal of Zoology*, 253:447–456, 2001.

[Pat96] D. W. Patterson. *Künstliche neuronale Netze*. Springer, Prentice Hall, 1996.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, San Francisco, 1988.

[PG90] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[PJ00] S. Parsons and G. Jones. Acoustic identification of twelve species of echolocating bat by discriminant function analysis and artificial neural networks. *Journal of Experimental Biology*, 203:2641–2656, 2000.

[PNK95] P. Pudil, J. Novoviĉová, and J. Kitler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125, 1995.

[Pre00] K. N. Prestwich. The control of carrier frequency in cricket calls: A refutation of the subalar-tegminal resonance/auditory feedback model. *Journal of Experimental Biology*, 203:585–596, 2000.

[Pre01] D. Prescher. *EM-basierte maschinelle Lernverfahren für natürliche Sprachen*. PhD thesis, University of Stuttgart, 2001.

[PS93] J. Park and I. W. Sandberg. Approximation and Radial Basis Function Networks. *Neural Computation*, 5:305–316, 1993.

[PW81] K. N. Prestwich and T. J. Walker. Energetics of singing crickets: Effect of temperature in three trilling species (orthoptera: Gryllidae). *Journal of Comparative Physiology*, 143:199–212, 1981.

[Qui93] J.R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.

[RB86] H. Römer and W. J. Bailey. Insect hearing in the field ii. male spacing behaviour and correlated acoustic cues in the bushcricket mygalopsismarki. *Journal of Comparative Physiology*, 159:627–638, 1986.

[RBF+03] M. Raab, C. Beck, F. Förster, C. Dietrich, and F. Schwenker. Verfahren zur merkmalsauswahl bei mehrklassifikatorsystemen. Technical report, Universität Ulm and DORSA, 2003. *www.dorsa.de/references.html*.

[RD94] K. Riede and K. Duffner. Systematik und biogeographie nordwestamazonischer orthopteren. *Andrias*, 13:151–160, 1994.

[RID01] K. Riede, S. Ingrisch, and C. Dietrich. Integration of orthoptera collection data within a virtual museum: the german orthoptera collections database. In *Metaleptea Special Meeting*, 2001.

[Rie93] K. Riede. Monitoring biodiversity: Analysis of amazonian rainforest sounds. *Ambio, A Journal of the Human Environment*, 22(8):546–548, 1993.

[Rie98] K. Riede. Acoustic monitoring of orthoptera and its potential for conversation. *Journal of Insect Conservation*, 2:217–223, 1998.

[RJ86] L. R. Rabiner and B. H. Juang. An introduction to hidden markov models. *IEEE Magazine on Accoustics, Speech and Signal Processing*, 3(1):4–16, 1986.

[RJ93] N. Rabiner and L. Y. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, New Jersey, 1993.

[RKH90] K. Riede, G. Kämper, and I. Höfler. Tympana, auditory thresholds, and projection of tympanal nerves in singind and silent grasshoppers. *Zoomorphology*, 109:223–230, 1990.

[RMS91] H. Ritter, T. Martinetz, and K. Schulten. *Neuronale Netze*. Addison-Wesley, 2nd edition, 1991.

[RS75] L. R. Rabiner and S. E. Sambur. An algorithm for determining the endpoints of isolated utterances. *Bell Syst. Technical Journal*, 54(2):297–315, 1975.

[Rus88] G. Ruske. *Automatische Spracherkennung*. Oldenbourg Verlag, München, 1988.

[SBK+99] M. D. Szymanski, D. E. Bain, K. Kiehl, S. Pennington, S. Wong, and K. R. Henry. Killer whale *(orcinus orca)* hearing: Auditory brainstem response and behavioral audiograms. *Journal of the Acoustical Society of America*, 106(2):1134–1141, 1999.

[SC78] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoustics, Speech, and Signal Proc.*, 26:43–49, 1978.

[Sch92] R. Schalkoff. *Pattern Recognition - Statistical, Structural and Neural Approaches*. John Wiley & Sons, New York, 1992.

[Sch94] K. Schildberger. The auditory pathway of crickets: Adaptions for intraspecific communications. In K. Schildberger and N. Elsner, editors, *Neuronal Basis of Bihavioral Adaptions*. Fischer, 1994.

[SD00] F. Schwenker and C. Dietrich. Initialization of radial basis function networks using classification trees. *Neural Network World*, 10:473–482, 2000.

[SDKP03] F. Schwenker, C. Dietrich, H. A. Kestler, and G. Palm. Radial basis function neural networks and temporal fusion for the classification of bioacoustic time series. *Neurocomputing*, 51:265–275, 2003.

[SDP04] F. Schwenker, C. Dietrich, and G. Palm. Classification of bioacoustic time series by training a decision template fusion mapping. In Michel Verleysen, editor, *European Symposium on Artificial Neural Networks 2004*, pages 555–560. d-side, Belgium, 2004.

[SH97] L. Schrader and K. Hammerschmidt. Computer-aided analysis of acoustic parameters in animal vocalisations: A multiparametric approach. *Bioacoustics*, 7:247–265, 1997.

[Sin98] S. Singh. 2D spiral pattern recognition with possibilistic measures. *Pattern Recognition Letters*, 19(2):141–147, 1998.

[Sin00] S. Singh. Multiple forecasting using local approximation. *Pattern Recognition*, 34(2):443–455, 2000.

[SK02] C. A. Shipp and L. I. Kuncheva. Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion*, 3(2):135–148, 2002.

[Ska97] D. B. Skalak. *Prototype Selection for Composite Nearest Neighbour Classifiers*. PhD thesis, University of Massachusetts, 1997.

[SKP01] F. Schwenker, H. A. Kestler, and G. Palm. Three phase learning for radial basis function networks. *Neural Networks*, 14:439–458, 2001.

[SKPH94] F. Schwenker, H. A. Kestler, G. Palm, and M. Höher. Similarities of LVQ and RBF learning - A survey of learning rules and the application to the classification of signals from high-resolution electrocardiography. In *Proc. IEEE International Conference on Systems, Man and Cybernetics*, pages 646–651, 1994.

[Sku01] M. Skurichina. *Stabilizing Weak Classifiers*. PhD thesis, University of Delft, 2001.

[SMLH02] H. Stöltig, T. E. Moore, and R. Lakes-Harlan. Substrate vibrations during acoustic signalling in the cicada *okanagana rimosa*. *Journal of Insect Science*, 2, 2002. *www.insectscience.org*.

[SP99] F. T. Sommer and G. Palm. Improved bidirectional retrieval of sparse patterns stored by hebbian learning. *Neural Networks*, 12:281–297, 1999.

[SS99] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.

[SSB+97] B. Schölkopf, K.-K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, 45(11):2758–2765, 1997.

[ST95] E. G. Schukat-Talamazzini. *Automatische Spracherkennung: Statistische Verfahren der Musteranalyse*. Vieweg Verlag, Braunschweig, Wiesbaden, 1995.

[Stu99] A. Stumpner. Comparison of morphology and physiology of two plurisegmental sound-activated interneurons in a bush-cricket. *Journal of Comparative Physiology*, 185:199–205, 1999.

[Tay95a] A. Taylor. Bird flight call discrimination using machine learning. *Journal of the Acoustical Society of America*, 97(5):3369–3371, 1995.

[Tay95b] A. Taylor. Recognising biological sounds using machine learning. In *Proceedings of Eighth Australian Joint Conference on Artificial Intelligence*, 1995.

[TB96] S. Thrun and A. Brücken. Integrating grid-based and topological maps for mobile robot navigation. *National Conference on Artificial Intelligence Portland, Oregon*, 1996.

[TD01] D. M. J. Tax and R. P. W. Duin. Combining one-class classifiers. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 299–308. Springer, 2001.

[TDvB97] D. M. J. Tax, R. P. W. Duin, and M. van Breukelen. Comparison between product and mean classifier combination rules. In *Proceedings of the Workshop on Statistical Pattern Recognition*, 1997.

[The89] C. W. Therrien. *Decision Estimation and Classification*. John Wiley & Sons, New York, 1989.

[TKP97] B. Talle, G. Krone, and G. Palm. Comparison of neural architectures for sensorfusion. In *Proceedings of the International Conference of Acoustics Speech and Signal Processing (ICASSP)*, pages 3273–3276. Munich, Germany, 1997.

[TS95] Janet M. Twomey and Alice E. Smith. Committee networks by resampling. *Intelligent Engineering System Through Artificial Neural Networks*, 5:153–158, 1995.

[TvBDK00] D. M. J. Tax, M. van Brueckle, R. P. W. Duin, and J. Kittler. Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition*, 33:1475–1485, 2000.

[TW99] B. Talle and A. Wichert. Audio-visual sensorfusion with neural architectures. In *Proceedings of Audio-Visual Speech Processing (AVSP)*. Santa Cruz, 1999.

[VC98] P. Verlinde and G. Chollet. Combining vocal and visual cues in an identity verification system using k-nn based classifiers. In *Proceedings of the IEEE Workshop on Multi-Media Signal Processing*. Los Angeles, CA, USA, 1998.

[VHH98] P. Vary, U. Heute, and W. Hess. *Digitale Sprachsignalverarbeitung*. B. G. Teubner, Stuttgart, 1998.

[Vit67] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Information Theory*, 13:260–269, 1967.

[Wal73] T. J. Walker. Systematics and acoustic behavior of u.s. and caribbean short-tailed crickets (orthoptera: Gryllidae: *Anurogryllus*). *Ann. Entomol. Soc. Am.*, 66(6):1269–1277, 1973.

[Wal74] T. J. Walker. Effects of temperature, humidity, and age on stridulatory rates in *atlanticus* spp. (orthoptera: Tettigoniidae: *Decticinae*). *Ann. Entomol. Soc. Am.*, 68(3):607–611, 1974.

[WAP99] S. Wermter, G. Arevian, and C. Panchev. Recurrent neural network learning for text routing. In *Proceedings of ICANN-99, 9th International Conference on Artificial Neural Networks*, pages 898–903, Edinburgh, UK, 1999. Institution of Electrical Engineers, London, UK.

[WC75] T. J. Walker and T. C. Carlysle. Stridulatory file teeth in crickets: Taxonomic and acoustic implications (orthoptera: Gryllidae). *Journal of Insect Morphol. and Embryol.*, 4(2):151–158, 1975.

[WC96] B. A. Whitehead and T. D. Choate. Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction. *IEEE Transactions on Neural Networks*, 7(4):869–880, 1996.

[WE95] J. Wiles and J. L. Elman. Learning to count without a counter: A case study of dynamics and activation landscapes in recurrent networks. In *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, pages 482–487. Cambridge, MA: MIT Press, 1995.

[Web96] B. Webb. A crieket robot. *Scientific American*, 275(6):62–67, 1996.

[Wer92] K.-D. Wernecke. A coupling procedure for discrimination of mixed data. *Biometrics*, 48:497–506, 1992.

[WH00] B. Webb and R. R. Harrison. Integrating sensorimotor systems in a robot model for cricket behaviour. In *SPIE Vol. 4196: Sensor Fusion and Decentralized Control in Robotic Systems III*, pages 113–124. SPIE, 2000.

[WHH+89] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang. Phoneme recognition using time-delay neural networks. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 37(3):328–339, 1989.

[WJP00] W. Wang, P. Jones, and D. Partridge. Diversity between neural networks and decision trees for building multiple classifier systems. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, pages 240–249. Springer, 2000.

[WKB97] K. Woods, W. P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:405–410, 1997.

[WM96] D. Wolpert and W. Macready. Combining stacking with bagging to improve a learning algorithm. Technical report, Santa Fe Institute, 1996.

[Wol99] D. Wolf. *Signaltheorie: Modelle und Strukturen*. Springer Verlag, Berlin, 1999.

[Woo90] P. C. Woodland. Isolated word speech recogniton based on connectionist techniques. *Br. Telecom. Technol. Journal*, 8(2):61–66, 1990.

[WR78] R. H. Wiley and D. G. Richards. Physical constraints on acoustic communication in the atmosphere: implications for the evolution of animal vocalizations. *Behav. Ecol. Sociobiol.*, 3:69–94, 1978.

[WS00] B. Webb and T. Scutt. A simple latency-dependent spiking-neuron model of cricket phonotaxis. *Biological Cybernetics*, 82(3):247–269, 2000.

[XKS92] Lei Xu, Adam Krzyzak, and Ching Y. Suen. Methods of combining multiple classifiers and their applications in handwritten character recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 22(3):418–435, 1992.

[YA99] M.-H. Yang and N. Ahuja. Recognizing hand gestures using motion trajectories. In *Proc. of IEEE CS Conference on Computer Vision and Pattern Recognition*, pages 466–472, 1999.