A Pattern Recognition Approach to Electromyography Data

Ivan Stefanov Mitzev

Follow this and additional works at: https://scholarsjunction.msstate.edu/td

Recommended Citation
https://scholarsjunction.msstate.edu/td/222

This Graduate Thesis is brought to you for free and open access by the Theses and Dissertations at Scholars Junction. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Scholars Junction. For more information, please contact scholcomm@msstate.libanswers.com.
A PATTERN RECOGNITION APPROACH TO ELECTROMYOGRAPHY DATA

By

Ivan Stefanov Mitzev

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Electrical Engineering
in the Department of Electrical Engineering

Mississippi State, Mississippi
August 2010
Copyright 2010

By

Ivan Stefanov Mitzev
A PATTERN RECOGNITION APPROACH TO ELECTROMYOGRAPHY DATA

By

Ivan Stefanov Mitzev

Approved:

Nicolas H. Younan  
Professor and Department Head,  
James Worth Bagley Chair  
Department of Electrical and  
Computer Engineering  
Mississippi State University  
(Director of Thesis)

Jenny Q. Du  
Associate Professor  
Department of Electrical and Computer  
Engineering  
Mississippi State University  
(Committee Member)

Pan Li  
Assistant Professor  
Department of Electrical and Computer  
Engineering  
Mississippi State University  
(Committee Member)

Jim Fowler  
Professor and Graduate Program Director  
Department of Electrical and Computer  
Engineering  
Mississippi State University

Sarah Rajala  
Dean  
Bagley College of Engineering  
Mississippi State University
EMG classification is widely used in electric control of mechanically developed prosthesis, robots development, clinical application etc. It has been evaluated for years, but the main goal of this research is to develop an easy to implement and fast to execute pattern recognition method for classifying signals used for human gait analysis. This method is based on adding two new temporal features (form factor and standard deviation) for EMG signal recognition and using them along with several popular features (area under the curve, wavelength function-pathway and zero crossing rate) to come up with a low complexity suitable feature extraction. Results are presented for EMG data and a comparison with existing methods is made to validate the applicability of the foregoing method. It is shown that the best combination in terms of accuracy and time performance is given by spectral and temporal extraction features along with neural network recognition (NN) algorithm.
ACKNOWLEDGEMENTS

I would like first to thank Dr. Nicolas H. Younan for sharing his experience and knowledge during several courses I have taken during my study, as well as for his guidance through difficulties of signal processing and for his support for this project. I would like also to thank my committee members- Dr. Jenny Du and Dr. Pan Li.

Special thanks to Myosotic, LLC (Kirkland, WA) for giving me the opportunity to use their development environment and data analysis system. Also, special thanks go to Motion Lab Systems, Inc (Baton Rouge, LA) for posting materials over the Internet for free educational use.

Finally, I would like to thank to the Mississippi State University (MSU) staff for accepting me into the distance learning graduate program and giving me the opportunity to achieve the best of my potential.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGEMENTS</th>
<th>ii</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
</tbody>
</table>

## CHAPTER

### I. INTRODUCTION .................................................................10

- Introduction to the problem .................................................10
- Thesis statement and contribution .......................................11
- Thesis organization ............................................................12

### II. EMG SIGNALS AND PROPERTIES........................................13

- EMG data – general overview .............................................13
  - Organization of the muscle and action potential .................13
  - Schematic model for MUAP ..............................................14
  - Structural and functional factors influencing EMG signals .....15
- EMG signals acquisition ....................................................16
  - Data acquisition sets- EMG sensors, amplifiers .................16
  - Sampling frequency ....................................................19
  - Filtering of sampled sequences .....................................19
  - Analog to Digital Convertors (ADC) consideration ............20
- EMG analysis ....................................................................21
  - Characterization of non-stationary signals and dynamic systems ....21
  - Surface EMG spectrum and signal modeling ....................23

### III. BACKGROUND AND RELATED RESEARCH..........................29

- Pattern recognition of EMG signal using evolutionary algorithm 29
- EMG pattern classification using spectral estimation and neural network .................................................................30
- Classification of electromyography signals based on directed transfer function ..............................................................32
- A heuristic fuzzy logic approach to EMG pattern recognition ........33
Classification of the surface EMG signals using RQA based representation..........................................................................................................................35
Feature extraction of surface EMG signals based on wavelet coefficient entropy ..........................................................................................................................37

IV. RECOGNITION METHODS ..........................................................................................................................40
Probabilistic Models and Statistical Decisions ..................................................................................................40
   Likelihood function and statistical decision ..................................................................................................40
   Bayes classifier for normal patterns .............................................................................................................41
   Nonparametric techniques: $k_n$- nearest neighbor estimation ....................................................................42
   Linear discriminant functions: Support vector machines (SVM) ..............................................................45
   Neural networks (NN) ..................................................................................................................................46
Practical evaluation of the two classification methods ..................................................................................51
   Distance functions and Mahalanobis distance classification method ......................................................51
   Chi-square test ..............................................................................................................................................52
   Neural networks (NN) recognition method ..................................................................................................54

V. FEATURES EXTRACTION METHODS ........................................................................................................59
Temporal feature extraction methods ............................................................................................................59
   Overview of temporal feature extraction methods .......................................................................................59
   Newly proposed feature extraction techniques ............................................................................................61
Spectral feature extraction methods ..............................................................................................................63
   Overview of spectral features methods ........................................................................................................63
   Parametric spectral estimation ....................................................................................................................63
   AR model order selection ............................................................................................................................64
   Pros and cons of Yule-Walker model ...........................................................................................................66

VI. RESULTS AND DISCUSSION .....................................................................................................................69
Research goal and results workability .............................................................................................................69
   Data collection environment .........................................................................................................................69
   Algorithm implementation ............................................................................................................................70
   Actual results ................................................................................................................................................77
   Cross-validation analysis and kappa coefficient ..........................................................................................79
   Discussion ....................................................................................................................................................91

VII. CONCLUSION ..........................................................................................................................................93
Summary .........................................................................................................................................................93
Limitations and obstacles .................................................................................................................................94
Future development ........................................................................................................................................94
REFERENCES ..................................................................................................................95

APPENDIX

A  MATLAB PROGRAM FOR INVESTIGATING THE OPTIMAL AR
MODEL, ACCOUNTING DIFFERENCES BETWEEN PSDS
OF THE ORIGINAL AND THE MODELED SIGNAL.......................97

B  MATLAB CODE FOR INVESTIGATING THE NEURAL
NETWORK PARAMETERS...............................................................100

C  MATLAB CODE FOR INVESTIGATING THE AKAIKE
CRITERION ........................................................................................105

D  MATLAB CODE FOR INVESTIGATING THE STABILITY OF
THE MODEL.....................................................................................108

E  C# CODE FOR CALCULATING THE MAHALANOBIS
DISTANCE BETWEEN THE FIVE FEATURES POINT
AND THE GROUP OF DEFINED POINTS .................................111
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Averaged AR model orders p, which gives closest to the original spectral characteristic for adequate signal</td>
</tr>
<tr>
<td>2</td>
<td>XOR functionality</td>
</tr>
<tr>
<td>3</td>
<td>Chi-square statistics</td>
</tr>
<tr>
<td>4</td>
<td>Learning rate vs. recognition error</td>
</tr>
<tr>
<td>5</td>
<td>Investigating the desired error optimal value in terms of accuracy and performance time</td>
</tr>
<tr>
<td>6</td>
<td>A Extracted feature for EMG1</td>
</tr>
<tr>
<td>7</td>
<td>Extracted feature for EMG4</td>
</tr>
<tr>
<td>8</td>
<td>Error rates from recognition, using two factors (x1 = Zero Crosses, x2 = STD) and Mahalanobis distance recognition algorithm</td>
</tr>
<tr>
<td>9</td>
<td>Error rates from recognition, using three factors (x1 = Zero Crosses, x2 = STD, and x3 = Pathway function)</td>
</tr>
<tr>
<td>10</td>
<td>Time series feature extraction representation for EMG1</td>
</tr>
<tr>
<td>11</td>
<td>Error dependency on window length of signal’s segment</td>
</tr>
<tr>
<td>12</td>
<td>Temporal feature extraction method results</td>
</tr>
<tr>
<td>13</td>
<td>Spectral feature extraction method results</td>
</tr>
<tr>
<td>14</td>
<td>Landis and Koch interpretation of kappa coefficients</td>
</tr>
<tr>
<td>15</td>
<td>Ten mixed features, NN recognition method</td>
</tr>
<tr>
<td>16</td>
<td>Five temporal features, NN recognition method</td>
</tr>
<tr>
<td>17</td>
<td>Five spectral features, NN recognition method</td>
</tr>
<tr>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>18</td>
<td>Seven (5 spectral + 2 temporal) features, NN recognition method .................84</td>
</tr>
<tr>
<td>19</td>
<td>Seven (5 spectral + 2 temporal) features, Mahalanobis distance recognition method .................................................................84</td>
</tr>
<tr>
<td>20</td>
<td>Five (5 spectral) features, Mahalanobis distance recognition method .............85</td>
</tr>
<tr>
<td>21</td>
<td>Five (5 temporal) features, Mahalanobis distance recognition method ............86</td>
</tr>
<tr>
<td>22</td>
<td>Kappa coefficients ..................................................................................87</td>
</tr>
<tr>
<td>23</td>
<td>Overall accuracy [%] ...............................................................................87</td>
</tr>
<tr>
<td>24</td>
<td>Average accuracy and standard deviation of the measured accuracy (5 Spectral features) ..............................................................88</td>
</tr>
<tr>
<td>25</td>
<td>Average accuracy and standard deviation of the measured accuracy (5 Temporal features) ...............................................................89</td>
</tr>
<tr>
<td>26</td>
<td>Average accuracy and standard deviation of the measured accuracy 7 mixed features (five spectral + two temporal) ..........................89</td>
</tr>
<tr>
<td>27</td>
<td>Average accuracy and standard deviation of the measured accuracy 10 mixed features (Five spectral + five temporal) .......................90</td>
</tr>
<tr>
<td>28</td>
<td>Euclidian distance between the parameters (Mean error; STD of error; Time for execution) point and (0;0;0) point ...........................91</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Schematic and real action potential (Courtesy Wikipedia) ................................14</td>
</tr>
<tr>
<td>2</td>
<td>Common drive of motor units in regulation of muscle force (Courtesy of De Luca and Erim Z).................................................................15</td>
</tr>
<tr>
<td>3</td>
<td>Real EMG signals visualization (Courtesy Myosotic, LLC). Presented EMG sequences are set from the same muscle scanned during similar condition.....................................................................................17</td>
</tr>
<tr>
<td>4</td>
<td>Single and double EMG sensors (Courtesy of DelSys, Inc) ........................................18</td>
</tr>
<tr>
<td>5</td>
<td>NI 9205 data acquisition board (Courtesy of National Instruments, Inc) ..............21</td>
</tr>
<tr>
<td>6</td>
<td>Power spectrum of overall EMG signal and averaged power spectrum (Courtesy Myosotic, LLC). .................................................................................................23</td>
</tr>
<tr>
<td>7</td>
<td>Y axes- MSE between original EMG signal’s PSD and PSD of its AR model; X axes- order of AR model, varying from 5 to 30 for EMG1 through EMG9 ........................................................................26</td>
</tr>
<tr>
<td>8</td>
<td>Spectral density representation of the original signals (in red) vs. AR model spectral representations (in blue) for order $p = 15$, for EMG1 through EMG9. Random sequence from investigated sets has been chosen to represent adequate signal. ........................................28</td>
</tr>
<tr>
<td>9</td>
<td>First combination of GA and NN (Courtesy [6]) ........................................................30</td>
</tr>
<tr>
<td>10</td>
<td>LVQ network architecture (Courtesy of [7].)........................................................31</td>
</tr>
<tr>
<td>11</td>
<td>Fuzzy logic method accuracy compared to different strategies for EMG signal recognition (Courtesy [10]) ............................................................35</td>
</tr>
<tr>
<td>12</td>
<td>Perceptron configuration (Courtesy of Sacha Barber, <a href="http://www.codeproject.com">www.codeproject.com</a>) ............................................................47</td>
</tr>
</tbody>
</table>
13 Backpropagation neural network structure. Connections from only one neuron per layer are shown; the rest of the neurons has similarly situated connection .................................................................54

14 Accuracy vs. AR model order representation for different number of neurons in the hidden layer (From left to right; top to bottom: number of hidden layer neurons = 10; 15; 20; 25) ........................................56

15 Akaike criterion vs. AR model order representation (p vary from 5 to 15) ........................................................................................................................................66

16 Roots of the polynomial constructed by AR model coefficients lie into the unit circle for the entire test set ..................................................................................................................68

17 EMG1 signal- nine trials. ..........................................................................................70

18 Features distribution for EMG1, EMG2 and EMG3 training set of signals ..................................................................................................................................................71

19 Features distribution for EMG1 through EMG9 training set of signals ..........72

20 Average accuracy [%] (10 mixed features not applicable with Mahalanobis distance) .................................................................................................................................90

21 STD of accuracy [%] (10 mixed features not applicable with Mahalanobis distance) .................................................................91
CHAPTER I
INTRODUCTION

Introduction to the problem

EMG classification is widely used in electric control of mechanically developed prosthesis, robotics, clinical applications, etc. It has been evaluated for years, and a variety of algorithms have been developed to serve different goals. The focus of this research is to develop an easy to implement and fast to execute algorithm for classifying signals used for human gait analysis. Laboratories investigating human gait collect all variety of EMG signals in binary format and store them onto storage devices. As different people work on the same acquisition system but store data in different folders, it is difficult to organize all of the scanned information. Most of the time, researches have to remember where the information is stored, i.e., which file contains the appropriate information needed. In general, there is documentation available on how such information has been collected, but it is a tedious process to keep an eye on all the documentation surrounding the acquiring process.

Another obstacle for organization is the variety of used file formats, such as C3D, CAMARC, TXT, and CSV, among others. In this research, we try to find a solution for this problem by investigating different classification algorithms, which could be used as a base for developing an EMG signal browser software. Such browser would help to browse the whole storage device and find signals/files with similar to certain (defined earlier) types of EMG signals. Once needed types of signals are found and displayed to
the user, he/she can decide which of found signals need to be further processed. That same mechanism could be used for searching certain signals for compression and backup. Then, the user will have a fast way to backup all needed signals in just two steps, automatic search and backup activation.

To accomplish this, the method used in this study is based on adding two new temporal features (form factor and standard deviation) for EMG signal recognition and using them along with several popular features (area under the curve, wavelength function-pathway and zero crossing rate) to come up with a low complexity suitable feature extraction. An objective method for performance estimation is proposed based on mean error, standard deviation of the error, and time for executing of the search algorithm. Results are presented for EMG data and comparison with existing methods is made to validate the applicability of the foregoing method. It is shown that the best combination in terms of accuracy and time performance is given by spectral extraction features along with a neural network recognition (NN) algorithm.

**Thesis statement and contribution**

The goal of this study is to find the most appropriate algorithm for organizing acquired data, specifically for EMG and force plate signals related to human gait. It investigates accuracy, robustness, and performance of several popular algorithms and proposes which one would best achieve the research’s goal. Adding two new temporal features (form factor and standard deviation) for EMG signal recognition and using them along with several features (area under the curve, wavelength function-pathway and zero crossing rate) already used in the practice, produces a low complexity feature extraction algorithm suitable for many application in the field.
Thesis organization

Chapters in this thesis are organized intuitively explaining step by step the problem of recognition, specifically recognition of EMG signals. Chapter two covers the basics of electromyography from a physiological point of view, as well as from engineering point of view. It introduces the reader to EMG signals specific processing. Chapter three begins discussing the problem, which this research claims to investigate by introducing previous and recent research to the theme. It represents five ideas as considered relevant and innovative to the project. Chapter four demonstrates recognition algorithms emphasizing on the Mahalanobis distance and neural network based algorithms and their theoretical basis. Chapter five demonstrates feature extraction methods, dividing them onto temporal and spectral. Chapter six covers the results obtained and discusses which of the proposed methods is most appropriate for solving the problem of signal searching. Chapter seven provides a conclusion of this research based on the results obtained from this investigation and introduces the readers to future work in this area. Finally, all developed and used source codes (MATLAB and C# codes) are presented in an Appendix.
CHAPTER II

EMG SIGNALS AND PROPERTIES

EMG data – general overview

**Organization of the muscle and action potential**

There are three basic types of muscles: skeletal, cardiac, and smooth. Electrical impulses stimulate contraction in skeletal muscles. Cardiac and smooth muscles are stimulated by non-voluntary impulses from internal pacemaker cells, which contract autonomously. The brain produces voluntary and involuntary impulses, which control nerve impulses that further stimulate muscles. Electromyography is interested in electrical signals produced to activate the muscle or so called action potentials. An action potential is the electrical discharge propagated along the cell membranes. Action potentials are used mostly for transmitting information between neurons and between neurons and muscles. Action potential has several phases as shown on Figure 1.
Schematic model for MUAP

Every motor neuron innervates several muscles fibers. This set defines the so-called motor unit. Intensity of contraction of a certain muscle depends on the number of motor units participating in that contraction. Once the nerve impulse reaches the “terminal button”, the end of the nerve most closely to the muscle fiber triggers a chemical reaction, which causes influx of ions within the muscles cells, thus creating potential and causing the muscle to contract. That electrical activity is called motor unit action potential (MUAP). An EMG signal typically is the result of multiple MUAPs interaction. The number of muscle fibers, along with motor unit combination and muscle fiber type, defines the shape of the electromyogram.
Normal MUAPs are biphasic or triphasic. Their duration is 3-15 ms with an amplitude of 100-300 µV and frequency in the range of 6-30/s [1]. Weak volitional effort causes the motor units to fire at about 5-15 pps (pulses per second). Grouping MUAPs has been observed as fatigue develops, leading to decreased high-frequency content and increased amplitude [2]. Figure 2 shows the composition of the surface EMG signal from single MUAP potentials.

![Motor Unit Action Potential MUAPT](image)

**Figure 2** Common drive of motor units in regulation of muscle force (Courtesy of De Luca and Erim Z).

**Structural and functional factors influencing EMG signals**

It is important to consider motor unit patterns because during electromyography data acquisition it is possible that an electrode will be inserted or placed on the surface closer to some muscles fibers than others. If the motor units are spread randomly into the observed muscle then scanning a small area of that muscle will give a representative
estimation of the EMG activity of the muscle [3]. On the other hand, every motor unit is composed of muscle fibers from the same type. Thus, acquiring information from a single unit may give an inaccurate observation of the whole muscle. Another factor influencing the EMG activity is the fiber size, which differs among muscles. Two types of fibers, slow-twitched and fast-twitched, could be distinguished. Slow-twitch fibers have relatively small diameters comparing with fast twitch fibers, and they define muscles with higher force outputs. In general, we should consider, during EMG acquisition, that the conduction velocity is proportional to the fiber diameter.

Skeletal muscles contain twitch fibers as they produce twitch response to nerve stimulus and also generate propagated action potential. Large muscles have hundreds of fibers per motor unit. EMG signals recorded using surface electrodes are complex signals, including interference patterns of several MUAP trains, and are difficult to analyze [4].

**EMG signals acquisition**

**Data acquisition sets- EMG sensors, amplifiers**

The data acquisition set for EMG signals contain EMG sensors, pre- amplifiers, analog-to-digital converters, and low level software for collecting data. EMG signals have amplitudes in the range of 50µV to 1mV without amplification. Typical EMG signals are shown in Figure 3. The EMG activity is captured via two types of sensors: invasive and non-invasive. Invasive methods use needle electrodes inserted into the muscle and are mostly used to investigate very specific parts of the muscle, even at the motor unit level. More popular sensors are sensors used for surface electromyography. Sensors could be single or double differential detectors (Figure 4). For a better contact
between the skin and attached electrodes, a specific gel should be applied. That gel decreases the moving artifacts as they occur during motion.

Figure 3  Real EMG signals visualization (Courtesy Myosotic, LLC). Presented EMG sequences are set from the same muscle scanned during similar condition.
The next part of the acquisition set is pre-amplifiers. Good pre-amplifiers have one or more of the following features: high input impedance, fixed gain, and low/high-pass filters. Gain could vary, but standard values are x20, x300, and x1000. Some pre-amplifiers also include protection against static electricity damages and/or radio-frequency interference. An example of a typical pre-amplifier is the Z03 EMG Preamplifier (courtesy of Motion Lab Systems, Inc). The following are some of the electrical characteristics of the Z03 EMG Preamplifier:

- Gain at 1 kHz x300 ±1% r
- Input Impedance > 100,000,000 ohms.
- Noise < 1.2uV RMS
• CMRR > 100 dB at 65Hz.
• Input Protection - RFI and ESD protected.
• Signal Bandwidth - 15Hz to 2,000Hz (-3dB).
• Power supply range ±5 Volts to ±15 Volts
• Power consumption 2.4mA per supply throughout voltage range.

**Sampling frequency**

The sampling frequency plays a significant role in the accuracy of reproducing the actual EMG signal from the scanned one. The sampling frequency in practice is defined by the Nyquist theorem and depends on the highest frequency of the investigated signal. Different EMG signals have different high-frequencies. For example, a typical EMG signal scanned with surface electrodes has a maximum frequency in the interval of 500-1000Hz, but an EMG signal scanned with needle electrode could reach a high frequency above 10,000Hz. Thus, a good scanning frequency for most surface EMG signals could be in the range of 1000-2000 Hz.

**Filtering of sampled sequences**

Filtering is an integral part of the EMG data acquisition set. Several kinds of filters are used to remove the noise from the real signal. Usually, pieces of the signal above 1000 Hz are considered electronic noise. Although there are cases when the signal’s frequency band may spread up to 10,000Hz, it is not usual to work with such signals. In almost every electronic set, there is an electrical power influence (50/60Hz) which has to be removed from the real signal. Also, there is noise from the so-called moving artifacts, sensor movement, skin friction, and others, which also impact the resulting signal and have to be removed as noise. All the described noises are suppressed
usually with digital filters during the analysis part of the data acquisition process.

Sometimes, there are hardware implemented filters embedded into the data acquisition set. One of the most popular of such filters is a low-pass filter used for anti-aliasing processing.

**Analog to Digital Convertors (ADC) consideration**

When digitizing EMG signals, several components should be considered: pre-amplifier’s gain, overall input noise to the system, and the voltage range of pre-amplifying the EMG signal. It is the researcher’s obligation to regulate the environmental noise, but it is not possible to regulate noise created from the measurement equipment itself. A good example of the intrinsic low level noise occurs with the DelSys, Inc equipment, which “has succeeded in obtaining an extremely low system noise of 5 µV (r.t.i.) per channel, measured by connecting the EMG electrode inputs to the reference potential. This means that if the output of a channel is recorded with no EMG signal, a baseline noise with an average amplitude of 5 µV(r.t.i.) will be observed.” Another important question to be considered is the ADC resolution. Practical results show that a 16-bit resolution for ADC is a good choice, although in some cases a 24-bit resolution ADC is required. In many cases, the number of supported channels for digital conversion is also an important characteristic. There are different systems on the market, but 2, 4, 16, and 32 channels are most popular for EMG data acquisition. In many cases, researchers prefer to use more advanced ADC that are capable of scanning more channels, but in such case an aggregate sampling frequency should be considered knowing that the sampling frequency per channel is equal to the aggregate sampling frequency divided by the number of channels. A relatively good choice for EMG acquisition is the NI data
acquisition board from National Instruments 9205 (Figure 5). With its 32 channels, 250 kS/s, and 16-bit resolution ADC, it covers most of the needs for EMG data acquisition.

Figure 5  NI 9205 data acquisition board (Courtesy of National Instruments, Inc)

**EMG analysis**

**Characterization of non-stationary signals and dynamic systems**

When speaking of EMG signals, we first have to characterize this type of signals from a statistical point of view. One of the main characteristics of electromyography signals is that it is non-stationary. Stationary signals have statistical properties which are invariant in time, meaning that the first order statistic (mean) is independent of time, but the second order statistic (variance) depends only on the time lag (m-n) as shown below:

\[
E \{ x (m) \} = E \{ x (n) \} = \mu
\]

(1)

\[
R (m,n) = E \{ x (m) x (n) \} = r (n - m) = r (i)
\]

(2)
where $i = n - m$ is the correlation lag; $E\{ \}$ is signal estimation; $R(\cdot)$ is correlation function; EMG signals could be treated as statistical time series. Such series have random behavior, and future values of that series could not be predicted. To make the standard EMG signal ready for further processing, some simplifications should be made.

One method of processing time-series non-stationary signals such as EMG signals is to divide the signals into small chunks (windows) and assume that those chunks could be associated with stationary signals. To avoid side-lobe effects in many cases, these chunks of data are windowed, i.e., some window function such as Hanning, Hamming, Blackman or others is applied to smooth out the signal. For better use, signal windows could be overlapped. Overlapping is used to maximize the use of data and yields smaller variance between the data windows. Data analyses of EMG signals, such as mean, variance, and Power-Spectral-Density (PSD), are often computed over a moving window, where The PSD estimate is given by:

$$\hat{P}_{xx}(k) = \frac{1}{PU} \sum_{\rho=1}^{\rho} |X_{\rho}(k)|^2,$$

(3)

Where $U = \sum_{n=0}^{n-1} w^2(n)$ is time domain window energy

A simple example (Figure 6) shows the difference between the overall signal power spectrum and the averaged power spectrum, obtained with a moving window method. That concrete example uses the following moving window parameters: window length of 300 samples, windows overlap of 60%, and a Hanning window type. In practice, overlapping should be in the interval of 50-60% for optimal results. The calculation based on statistical properties should be considered when temporal feature extraction is conducted.
Surface EMG spectrum and signal modeling

In general, the EMG spectrum is influenced by two factors: MUAP morphology and MU trains. As already mentioned, EMG signals are not stationary signals, and a better way to estimate their spectra is to separate the signal into smaller parts and average the spectra of those parts.

An EMG signal is susceptible to AR representation, as AR representation is used for “peaky” signals with some degree of success. In general, between 2 to 7 coefficients are enough to represent a signal which has a so-called a bell-shape spectrum [5]. A simple AR model to be used is the linear prediction modeling, described by:

\[ x(n) = - \sum_{k=1}^{p} a_k x(n-k) + e(n) \]  

(4)
where

\[ x(k) \] are samples of the modeled signal
\[ a_k \] are the AR model coefficients
\[ e(n) \] represents a white noise sequence
\[ p \] is the model order

This model can be easily identified as a filter with input \( e(n) \) and output \( x(n) \) with a transfer function given by:

\[
H(z) = \frac{X(z)}{E(z)} = \frac{1}{1 + \sum_{k=1}^{p} a_k z^{-k}}
\]  

(5)

Also, the spectrum of the modeled EMG, \( x(n) \), can be represented as:

\[
\hat{S}_x(w) = |X(w)|^2 = \frac{1}{1 + \sum_{k=1}^{p} a_k e^{-jwk}}
\]  

(6)

The above equation is only valid when \( e(n) \) truly represents white noise. An important question is what order \( p \) best fits the AR model for a certain set of investigated data. The AR model could be different for different sets of muscles. This current research considers muscles from 9 groups that have participated in human gait as specified later in this thesis. Another important question is what order \( p \) best represents the investigated 9 groups of muscles in such a way that the AR coefficients are not only unique for every muscle, but also help in distinguishing these muscles from the rest of the collected information in the archive.

Some randomly chosen set of every EMG sequence has been analyzed. The selected signal is divided onto smaller parts, and a Hanning function has been applied in this analysis with a window length of 0.37 sec. and 65% overlapping. After the AR model
coefficients are obtained for each sequence, they are averaged to obtain the so-called representative coefficients for all of the muscles. That is done for different values of the order- p. The white noise sequence is filtered with the resulting AR model coefficients, and spectral estimation of the resulting signal is obtained. Then, the averaged spectrum is compared with the original signal’s spectrum to determine the optimum order p, which most closely resembles the PSD of the original signal. To find the closest representation, the mean-square error between the modeled and the original signal spectrums is found. The order which gives the smallest value of that MSE will be considered as the best representative for an adequate muscle. Results from this approach are shown in Figure 7 and the resulting best order fit for all the 9 groups is tabulated in Table 1.
Figure 7  Y axes- MSE between original EMG signal’s PSD and PSD of its AR model; X axes- order of AR model, varying from 5 to 30 for EMG1 through EMG9

Table 1  Averaged AR model orders p, which gives closest to the original spectral characteristic for adequate signal

<table>
<thead>
<tr>
<th>Signal</th>
<th>Averaged AR model order</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>19</td>
</tr>
<tr>
<td>EMG2</td>
<td>17</td>
</tr>
<tr>
<td>EMG3</td>
<td>23</td>
</tr>
<tr>
<td>EMG4</td>
<td>26</td>
</tr>
<tr>
<td>EMG5</td>
<td>15</td>
</tr>
<tr>
<td>EMG6</td>
<td>20</td>
</tr>
<tr>
<td>EMG7</td>
<td>20</td>
</tr>
<tr>
<td>EMG8</td>
<td>22</td>
</tr>
<tr>
<td>EMG9</td>
<td>23</td>
</tr>
</tbody>
</table>
As seen from Table 1, the best results for AR modeling of the presented signals vary from 15 to 26 and depend on the characteristic of the signal itself. The goal of this study is to find a reliable yet fast solution for signal recognition, based on AR modeling and/or temporal feature extraction. For this purpose, a smaller order $p$ will lead to fast calculation and results will be obtained faster, especially during the search process of the archive (hard disk).

According to Table 1, the smallest order $p$ for which the AR representation is still reliable is $p = 15$. To validate this, we modeled a random signal with AR coefficients from the investigated signals for order $p = 15$ with coefficients of adequate muscles signal. The results are illustrated in Figure 8. This figure shows that such a representation produces spectral characteristics close to the original signal, although the representation error is not uniform for all the signals. From this point, we could assume that an order $p = 15$ is representative and from now on, we will investigate how decreasing that order will influence the recognition process. The goal again is to find a less erroneous AR model representation for the smallest possible order $p$. 

27
Figure 8  Spectral density representation of the original signals (in red) vs. AR model spectral representations (in blue) for order $p = 15$, for EMG1 through EMG9. Random sequence from investigated sets has been chosen to represent adequate signal.
 CHAPTER III
BACKGROUND AND RELATED RESEARCH

Pattern recognition of EMG signals is extensively used for prosthesis control, robotic development, and diagnosis in medicine. For this, many algorithms have been developed and still have value. Both basic and newly developed concepts are presented in this chapter, and used in this study as reference.

**Pattern recognition of EMG signal using evolutionary algorithm**

This method utilizes Neural Network (NN) and Generic Algorithm (GA) for recognition and combines them at some point. Principal component analysis (PCA) is used to extract features, along with a feature selection method, to reduce features to the most representative [6]. The first step of the algorithm is to use GA and NN to extract similar frequency bands from all the signals. The second step is to obtain a function that describes individual changes into the signal’s spectrum with the help of another NN.

This method investigates wrist motion and focuses on several motions like top, bottom, left, right, neutral, and inward/outward rotation. The sampling frequency for the scanning data is 10.48 kHz with a signal duration of 2.048 sec. A windowing function (Hanning) for every 256 samples is applied, and then FFT is used to find an averaged spectrum for every signal. Every frequency band is separated into sub-bands, and every sub-band has its mark, “0” or “1”, depending on whether or not a band is dominantly present in the whole signal band. Such decision is made by a GA algorithm. These “1” and “0” are input into a back-propagation NN with 15 hidden layer nodes and seven
output nodes related to the investigated motions, as shown in Figure 9. The next step is to create a function describing the frequency spectrum of the signals based on the results of the first step. Another set of GA and NN is applied at this step. The average recognition accuracy reported for this method is as follows: NN- 73.9%, FGANN- 77.5%, and SGANN- 79.9%. The calculation cost gained from frequent use of GA and NN is still high, and such algorithm is appropriate for recognition in prosthesis control. However, it is not appropriate for use for thousands of signal recognition for search purposes, as this study aims, because of its ineffective calculation cost.

Figure 9 First combination of GA and NN (Courtesy [6])

**EMG pattern classification using spectral estimation and neural network**

This method also investigates wrist movements and suggests a recognition method for further pattern recognition based on spectral estimation of the signal and neural network (NN). In detail, spectral estimation is performed via the Yule-Walker equations, but for recognition, a learning vector Quantization (LVQ) method is used. The
Yule-Walker method for obtaining the AR model coefficients is evaluated with the Levison-Durbin algorithm. The obtained coefficients are used as inputs for the LVQ network (Figure 10) for further recognition.

![LVQ network architecture](image)

**Figure 10** LVQ network architecture (Courtesy of [7].)

The competitive layer learns to classify the input vectors. The linear layer transforms the competitive layer’s classes into target classifications defined by the user [7]. Six hand gestures are proposed for recognition [7]. LVQ has ten input neurons and six output neurons. The learning rate for LVQ is specified as 0.9. The method reports fifty measurements for each hand for all investigated gestures, of which 30 sets are for network training and 20 for testing. The recognition accuracy, as reported, varies from 64 to 94% with an average of 78%.

This method is used in the current study as a base for further research. The AR model used in [7] has order $p = 10$ (as 10 neurons are specified for the LVQ input layer), but in our study, we investigate finding an optimum order $p$ (as shown in Chapter 1), for which the difference between the original signal’s PSD and the modeled one is minimum. As indicated in Chapter 1, $p = 15$ is a suitable value to start with. From now on, that order
p will be decreased to improve the calculation cost. The minimum value of p, for which
the accuracy is still decent, will be used for the actual implementation.

The LVQ network, on the other hand, is still complex and will slow down the
calculation process. That is why we investigate using two methods for recognition: The
Mahalanobis distance and Neural Network (NN)-based methods. Both methods are
simple to implement and fast to execute. The accuracy of the results, as shown later on, is
in the range of this proposed method, but recognition is fast enough to check all present
signals in an archive (on the user’s hard disk) and show the results in an observable time.

Classification of electromyography signals based on directed transfer function

This method investigates extension and flexion of the elbow. Signals from biceps
and triceps during flexion and extension of the elbow are represented with a Multivariate
Autoregressive (MVAR) model [8]. This method proposes that a Direct Transfer
Function (DFT) be used as the classifier for elbow positions. Further processing and
classification are done using support vector machine (SVM).

This method suggests describing every k-channel EMG system with an AR
model as:

$$x(t) = - \sum_{i=1}^{p} A(i)x(t-i) + n(t)$$  \hspace{1cm} (7)

Where p is the AR model order;

$$n(t) = [n_1(t), n_2(t), ..., n_k(t)]^T$$ is a vector of multivariate uncorrelated white noise
process

with covariance matrix $C \in \mathbb{R}^{k \times k}$

$A(i)$ is kxk matrices of model parameters

It is easy to show that the transfer function can be represented by:
where H(f) is a \( k \times k \) transfer function matrix. The interrelation between the scanned EMG channel is present at H(f), therefore DTF(f) can be easily defined as:

\[
DTF_{ij}^2(f) = \frac{|H_{ij}(f)|^2}{\sum_{m=1}^{n} |H_{im}(f)|^2}
\]

where DTF\(_{ij}\) represents the connection between channel i and channel j of the system and DTF (1; 2) and DTF (2; 1) and especially their maximum values represent the distinguishing features [8]. Ten sets of collected data are used to train the classifier and another ten sets to test the classifier. SVM is then used for classification [9].

In general, this method provides high accuracy, but it is applicable for small number of EMG channels. The SVM classification method is relatively slow in terms of calculation performance. Also, this method shows good results for only two types of muscles which are known to be unrelated for movements they are investigated for. For a bigger number of channels and possible overlap of features, this method may not be that effective.

**A heuristic fuzzy logic approach to EMG pattern recognition**

This method uses a heuristic fuzzy logic for pattern recognition for multifunctional prosthesis control. The mean and standard deviation are used for membership function construction [10]. Recognition based on fuzzy clustering uses the idea of grouping the sampled data in such a way that it “minimizes the variance of the data in the same cluster, but maximizes the variance of the data in different clusters”[10].
This algorithm was tested with different muscles and movement, but it shows best results for finger flexion, ulnar deviation, wrist extension, and wrist flexion recognition [10].

“The rms of the myoelectric signal is an accepted maximum likelihood estimator of EMG amplitude, and has been suggested as the choice method of data reduction for EMG signal processing because it provides physiologically significant information of the average power of the muscle” [11].

Once the membership functions and rules have been established, some small amount of calculation is used to produce results. Rules and membership functions are defined when the system is initialized. During its operational state, the method fuzzifies each input and check that the input against the set of rules and de-fuzzifies outputs for which these rules are true [10]. Figure 11 shows the success rate of fuzzy-logic based recognition system (black bar) compared to different recognition systems. The results shown from this fuzzy logic method are promising as the method is executed for short time and the accuracy is still very high. Although, many of the tests are performed on upper limbs, this method potentially can be applied for muscles of the lower limbs, investigated in this study. Another possible obstacle could be that the fuzzy logic algorithm requires extensive training with very similar signals and some variety of signals (as present on the researchers’ hard drive), which can lead to decreasing of accuracy.
Classification of the surface EMG signals using RQA based representation

This method emphasizes feature extraction and is based on Recursive Quantification Analysis (RQA). Two channels are used to collect raw EMG information, and on the output, eight motions are recognized. Ten RQA features are calculated for every channel. The extracted features are then used as input to a back propagation (BP)
neural network for further classification. According to RQ theory, every set \{x_1(t), x_2(t), \ldots, x_n(t)\} could form a trajectory in m phase space defined by:

\[
X_i = \left\{ x(i), x(i + \tau), \ldots, x(i + (m - 1) \tau) \right\}
\] (10)

where \(m\) is an adequate dimension and \(\tau\) is the time delay. The trajectory is described by:

\[
R_{ij}^{\text{dist}} = \Theta \left( r - ||X_i - X_j|| \right)
\] (11)

where \(X_j \in \mathbb{R}^m; i, j = 1 \ldots N\); 

\(N\) is the number of considered states; \(r\) is fixed radius;

\(||\cdot||\) is the norm; \(\Theta (\cdot)\) is the Heaviside function.

When the distance between \(X_j\) and \(X_i\) falls in radius \(r\), then \(R_{ij} = 1\); otherwise, \(R_{ij} = 0\). An \(N\times N\) structure is then drawn, which gives much information about the investigated sequence \(x(t)\). Ten parameters are calculated from the \(R_{ij}\) map as follows:

Recurrence rate:

\[
RR = \frac{1}{N^2} \sum_{i,j=1}^{N} R_{ij}
\]

Determinism: where \(P(1)\) is a histogram of the length \(l\) of the diagonal lines

\[
DET = \frac{\sum_{i,j=1}^{N} IP(l)}{\sum_{i,j=1}^{N} R_{ij}}
\]

Averaged diagonal length:

\[
L = \frac{\sum_{i=1}^{N} IP(l)}{\sum_{i=1}^{N} P(l)}
\]

Longest diagonal line: \(L_{\text{max}} = \max \{ l_i \mid l_i = 1 \ldots N_i \}\), where \(N_i\) is the number of diagonals.
Entropy of the diagonal line length:

$$\text{ENTR} = \sum_{l = l_{\text{min}}}^{N} p(l) \ln(p(l)); p(l) = \frac{1}{\sum_{l}} p(l)$$

Laminarity: $LAM = \frac{\sum_{v = v_{\text{min}}}^{N} vP(v)}{\sum_{v = v_{\text{min}}}^{N} vP(V)}$, where $P(v)$ is the histogram of the lengths- $v$ of the vertical lines.

Trapping time:

$$TT = \frac{\sum_{v = v_{\text{min}}}^{N} vP(v)}{\sum_{v = v_{\text{min}}}^{N} P(v)}$$

Maximum vertical line length:

$$V_{\text{max}} = \max \{v \mid i = 1 \ldots N_v \}; N_v = \sum_{P(v)} P(v)$$

Mean recurrent type of first type: $T_1$

Mean recurrent time of second type: $T_2$

All of these ten features are fed to a BP network with an intention to recognize eight motions: finger flexion, finger tension, forearm pronation, forearm supination, radial flexion of wrist, ulnar flexion of wrist, wrist flexion, and wrist extension. As reported, the recognition rate for four, six, and eight motions reaches averages of 95.2%, 81.4%, and 70.3%, respectively. As shown here, this method provides high accuracy in recognizing a variety of complex motions with a relatively large amount of combinations. Although the reported results are accurate enough, such a method would not be suitable for the current study needs as the calculation time for all ten features is very high, which slows down the entire search process.

**Feature extraction of surface EMG signals based on wavelet coefficient entropy**

This project investigates two popular EMG patterns: forearm supination (FS) and forearm pronation (FP). The method divides EMG signals into 16 frequency sub-bands
and the wavelet coefficient entropy (WCE) between any two bands is calculated [12]. Once the EMG signal is decomposed onto 16 sub-bands with wavelet packet coefficients noted as:

\[ D_n = \{ d_n(k), k = 1, 2, \ldots K \} \]

where \( k \) symbolizes time and the coefficients are normalized, then the coefficient matrix looks like the following:

\[
D = \begin{bmatrix}
D_1 \\
D_2 \\
\vdots \\
D_j \\
\end{bmatrix} = \begin{bmatrix}
d_1(1) & d_1(2) & \cdots & d_1(K) \\
d_2(1) & d_2(2) & \cdots & d_2(K) \\
\vdots & \vdots & \ddots & \vdots \\
d_j(1) & d_j(2) & \cdots & d_j(K) \\
\end{bmatrix}, 
\]

(12)

Previous research shows that the two frequency bands (seven and eight) are more definitive for identifying FP and FS surface EMG signals than other FBs. Thus, in this method, WCEs in the 7\textsuperscript{th} FB and the 8\textsuperscript{th} FB are set as WCE features [12]. Note that WCE is defined as:

\[
WCE(n) = - \sum_{m=1}^{M} p_n(m) \ln \left( p_n(m) \right),
\]

(13)

where \( p_n(m) \) is the probability of the \( m \)-\textsuperscript{th} region for \( D_n(k) \)

where the probability for \((a_{m-1} a_m)\) to have \( N \) coefficients of \( D_n(k) \) is equal to \( p_n(m) = \frac{N}{M} \)

Using the previously calculated WCE(n) and the Bayesian decision rule for two classes, the current method aims to recognize FP from FS with relatively high accuracy. Although the accuracy in this case depends on the number of sampling points of the EMG sign, the resulting error steadily leans toward zero after 500 samples signal, which is a very high accuracy. This method also assumes that EMG FS and FP represent a Gaussian distribution process, and thus, the simplified Bayes rule could be used. Although the method uses only two sub-bands from the 16 sub-band multitude, it still could be
categorized as a slow method because of the high amount of calculations used during the WPT process. Although the Bayes classification method is suitable for the two classes distinguishing case, it is difficult to apply to a case that involves more than two classes, which this current study requires.
CHAPTER IV
RECOGNITION METHODS

Probabilistic Models and Statistical Decisions

Likelihood function and statistical decision

The unconditional probability of a certain event (or class), \( \omega_i \) \( (i = 1, 2, \ldots, N) \), known also as priori probability, \( P(\omega_i) \), reflects the prior knowledge of how likely an event is to occur. Usually, when two events are investigated and it is equally likely for either events to occur, the unconditional probability \( P(\omega_i) \) is equal to 0.5. Analogically, we can define a posteriori probability as some pattern \( x \) observed, coming from \( \omega_i \), and usually it is represented as \( P(\omega_i|x) \).

Classification, in general, is always done with some error. If a certain method indicates that some class is \( \omega_i \) instead of \( \omega_j \), then it is said that the classifier produces a loss \( L_{ij} \). For more than two classes’ classification, we can define the conditional risk for assigning \( x \) to \( \omega_i \) as follows:

\[
R_j(x) = \sum_{i=1}^{M} L_{ij} P(\omega_i|x)
\]  

(14)

The classifier then calculates all possible conditional risks for a certain \( x \) and identifies \( x \) as \( \omega_m \), for which \( R_m(x) \) is the smallest of all calculated risks. This classifier is called the Bayes classifier. From Bayes formula,

\[
P(\omega_i|x) = \frac{P(\omega_i)P(x|\omega_i)}{P(x)}
\]  

(15)
we can write

\[ R_j(x) = \left( \frac{1}{p(x)} \right) \sum_{i=1}^{M} L_{ij} p(x|\omega_i) P(\omega_i) \]  

(16)

From the above formula, we can synthesize a simple rule for a two-class classification problem:

\[ \frac{p(x|\omega_i)}{p(x|\omega_j)} > 1 \]  

\( x \in \omega_i \) if \( \frac{p(x|\omega_i)}{p(x|\omega_j)} > 1 \), as for \( m \) any pattern classifications we assume \( L_{ij} = 1 \), for \( i \neq j \)

For more classes, we can define a decision function as:

\[ d_i(x) = p(x|\omega_i) P(\omega_i), \quad i = 1, 2, \ldots N \]  

(17)

where \( N \) is the total number of compared classes. Then, the classification rule becomes:

\[ x \in \omega_i \text{ if } d_i(x) > d_j(x) \text{ for all } i \neq j \]  

(18)

Simplifying this decision function produces:

\[ d_i(x) = P(\omega_i|x) \text{ for } i = 1, 2, \ldots N \]  

(19)

To calculate \( p(x|\omega_i) \), it is common to assume a Gaussian distribution and estimate its mean and variance using the training set [4].

**Bayes classifier for normal patterns**

Many processes in nature can be described with a Gaussian PDF. The same is valid for EMG signals as well. The Gaussian distribution function for a general case for one event is written as:

\[ p(x) = \frac{1}{\sqrt{2\pi} \sigma} \exp \left[ -\frac{1}{2} \left( \frac{x - m}{\sigma} \right)^2 \right] \]  

(20)
where \( m \) is the mean of the distribution, and \( \sigma \) is the variance. When more signals are used, the distribution function becomes more complex:

\[
p(x|\omega) = \frac{1}{(2\pi)^{\frac{n}{2}}} \exp \left[ -\frac{1}{2} (x - m)^T C_i^{-1} (x - m_i) \right]
\]  
(21)

where \( m_i \) is the mean, and \( C_i \) is the covariance of the i-th sequence. Using the previous definition of the decision function for \( d_i(x) \) and simplifying that function by taking the logarithm of it, we obtain:

\[
d_i(x) = \ln P(\omega_i) - \frac{1}{2} \ln |C_i| - \frac{1}{2} [(x - m_i)^T C_i^{-1} (x - m_i)]
\]  
(22)

The above formula gives the theoretical definition of multiclass classification.

All of these methods are well-defined and give very precise results in terms of correct error estimation. One of the main obstacles of using such methods in practice is the vast amount of calculations that slow down the classification process. There are some more practically-oriented methods, such as component analysis methods, linear discriminant functions, neural networks, and non-parametric techniques like k-nearest neighbor, among others. Some of these methods are described below.

**Nonparametric techniques: k\(_n\)- nearest neighbor estimation**

To estimate the PDF- \( p(x) \), we find a value \( x \), ‘surround’ that the \( x \) value with a cell, and increase the boundaries of that cell until it captures the \( k_n \) nearest-neighbors of \( x \). Smaller cells produce a high density of \( x \), and largely grown cells produce a low density of \( x \). Therefore, we can write:

\[
p_n(x) = \frac{k_n}{V_n},
\]  
(23)
where $V_n$ is the cell’s volume, $k_n$ is an initially defined number to be reached, and $n$ is the total number of counted values of $x$. For practical use, we could define:

$$k_n = \sqrt{n} \tag{24}$$

Then, for $V_n$, we obtain:

$$V_n = \frac{1}{\sqrt{n} \, p(x)} \tag{25}$$

Choosing different values for $k$ brings different results for $p(x)$, and what exact value of $k$ should be used is determined by the concrete case.

It is interesting to note that although $p_n(x)$ is continuous, its slope is not. Furthermore, the points of discontinuity are rarely the same as the prototype points [13]. It is also good to have an estimation of the posteriori probability of the $k_n$- method. We used a volume $V$ and captured $k$ samples of some class $\omega_i$. As we have already estimated,

$$p_n(x, \omega_i) = \frac{k_i}{V} \tag{26}$$

Therefore, the estimation of the posteriori $P(\omega_i|x)$ will be:

$$P_n(\omega_i|x) = \frac{p_n(x, \omega_i)}{\sum_{j=1}^{c} p_n(x, \omega_j)} = \frac{k_i}{k} \tag{27}$$

For smaller cells, this approach is very useful in terms of performance and processor time consumption. It is difficult to estimate how big the window can be and still remain close to the real estimation function. Another method, called the Parzen-window method, uses $k_n$-nearest neighbor, but the difference is that in the Parzen method, $V_n$ depends on $n$. Usually,
As an example, we can test how the k-nearest neighbor method behaves for the two-class case. For two classes, we can choose a test point \( x \), define \( k = 5 \) (often used in practice), and grow the spherical region until it encloses \( k \) samples. Then, we can count the enclosed samples from the \( \omega_1 \) class, and depending on if they are more (you need to clarify here--more than what?) or not, then \( x \) is classified as \( \omega_1 \) or \( \omega_2 \), respectively. The method seems simple, and because of its low computational complexity, it is a popular method among classification methods. Because the computational complexity is very important for comparing classification methods, below is an estimation of the method’s computational cost. In a case with \( n \) training samples in \( d \)-dimensional space and when the closest point to \( x \) is searched, the calculation of the Euclidian distance between any two points is of \( O(d) \) complexity, and for \( d \) dimensional space, it becomes \( O(nd) \).

Three techniques are most commonly used for reducing the computational complexity: partial distance calculation, editing stored prototypes, and prestructuring [13]. The partial distance includes some reduced number of dimension calculation \( r < d \). For prestructuring, a search tree is created, where all the points are selectively connected. During the estimation process, only distances to the “root” point and to several other points linked to that root are calculated until the point finds its place into the tree. In this way, not all of the presented points are included in the calculation. The third optimization method, editing, removes samples that are considered “useless” from the search. Usually, this is done by removing the training points from the search points, which are from the same category as the investigated points. The complexity in such cases reduces to:

\[
O\left(d^{1-n\left[\frac{d}{2}\right]}/\ln n\right)
\]
This is an example of reducing the complexity, while the accuracy remains almost the same. Although this method is good from a performance point of view, it has some negatives, including the fact that all the training points must be known in advance.

**Linear discriminant functions: Support vector machines (SVM)**

Support vectors are patterns equally close to a hyperplane. Support vector machines (SVM) are trained by choosing the worst-classified pattern [13].

Searching for a worst classifying pattern is a complex task, and it becomes more complex with adding new patterns. Because of its complexity, this method is used for a relatively small amount of problems.

The first step in the SVM training process is to define a function $f()$, which maps the training set to a higher dimensional space. Typically, such a function depends on the problem under consideration, and is found empirically. If a problem is relatively new, the Gaussian or another type of polynomial function is used.

A good and yet a simple example of using SVM is solving the XOR problem. It defines four points, as shown in Table 2.

<table>
<thead>
<tr>
<th>Table 2</th>
<th>XOR functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>X2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>

As the planes of the two classes overlap each other, it is difficult to find a line that separates one from another. Thus, the space defined with the XOR problem is converted into linearly separable space. These four training patterns are mapped into six dimensional spaces by:
1; \sqrt{2} x_1, \sqrt{2} x_2, \sqrt{2} x_1 x_2, x_1^2, x_2^2

Therefore, the optimal hyperplane is found to be \( g(x_1, x_2) = x_1 x_2 = 0 \), and the margin is \( b = \sqrt{2} \) \[^{[13]}\]. Support vector machines are better than some of the other existing methods because they make it more difficult for overfitting as the decision rule is defined by several support vectors not only by transforming to some other space as linear discriminant analysis (LDA), for example.

**Neural networks (NN)**

Neural networks are vastly used techniques for solving non-linear problems. It started to evolve as a concept in the early 1950’s, but it really became popular in the early 1980’s. Since then, NN have become a base for many solutions where modeling is a difficult and time-consuming process. Today, NN are used to solve problems such as function approximation, classification, and time series prediction, among others. Another popular name for neural networks is artificial neural networks (ANN) because NN resemble biological neurons and their connectivity to some degree. Artificial networks in general consist of modules called perceptrons. Figure 12 shows the basic construction of a perceptron.
The perceptron concept was invented in 1962 by Rosenblatt. Every perceptron has inputs, weighted with some coefficients $w_i$, which, along with the weighted bias, made the input to activation function. The simplest activation function compares if the sum of the weighted inputs is above a threshold level: if so, it fires 1 as output, in any other case, the output is 0. There are different types of activation function, but the most popular ones are the sigmoid and step functions. The sigmoid function is used in a large scale because it is differentiable, and this is an important condition for back-propagation networks. The formula of the sigmoid function is:

$$f(x) = \frac{1}{1 + e^{-ln}}$$

(31)

Another popular activation function is the step function, which is defined by:

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ 1 & \text{if } x > 0 \end{cases}$$

(32)
One of the major problems, which occurs when NN are used, is the problem of learning. There are three major learning concepts:

Supervised learning
Unsupervised learning
Reinforcement learning

The goal for supervised learning is to discover the mapping function between the input and the given output vectors. On the other hand, unsupervised learning has a set of input parameters, x, but does not have a reference output. Its goal is to minimize the cost function, which connects the input x and the network outputs. Reinforcement is characterized by learning for which the input and outputs are not given. The goal in this learning is to minimize some cost, like cumulative cost. In practice, a learning process is just changing the weights and the bias of the perceptron until given conditions are met.

Neural networks are also classified by their structural identity. NN can be a single layer and multilayer networks. Multilayer networks consist of several layers, usually described as input layer, hidden layers, and output layer. Introducing more layers increases the complexity of the network and its computational cost. Less complex problems are solved with three-layer NN having inputs, one hidden layer, and outputs.

The latter structure is used in this study for further classification. In such a structure, the activation processes can be described as:

\[
net_j = \sum_{i=1}^{d} x_i w_{ij} + w_{0j} = \sum_{i=0}^{d} x_i w_{ij} \equiv w^i_j x
\]

(33)

where \(i\) is the index of the input vector units, and \(j\) indicates the consequent hidden layer neuron. Every hidden layer of neurons computes its activation function \(net_j\), summing the weighted inputs. Each hidden layer of neurons emits an output, which is
where $f(.)$ could be any function, but the most commonly used (as described above) are the step function and sigmoid function. Similarly to the hidden activation, each output neuron calculates its activation function by summing the weighted outputs of each hidden layer as:

$$net_k = \sum_{j=0}^{n_h} y_j w_{kj} + w_{k0} = \sum_{j=0}^{n_h} y_j w_{kj} = w_k^T y$$

(35)

where the index k indicates the output layer neurons, and $n_h$ is the number of hidden units. The output of the NN is calculated by:

$$z_k = f(net_k)$$

(36)

where again $f(.)$ is the activation function, but this time of the output layer.

Using the above equations, we can model a function that connects any given input to any given output, but the major remaining question is how to establish proper weights and biases for the mapping function. One answer to this question gives the so-called backpropagation algorithm. That is one of the simplest methods for supervised training. Once some outputs are produced, they are compared to some desired set of answers. This generates an error, described as:

$$J(w) = \frac{1}{2} \sum_{k=1}^{c} (t_k - z_k)^2 = \frac{1}{2} \| r - z \|^2$$

(37)

where $t$ and $z$ represent targets and outputs of the NN, respectively, $c$ is the length of $t$ and $z$, and $w$ represents all the weights in the network. The weights are initialized with random values and adjusted later according to direction, which reduces the error, i.e.:
where $\eta$ indicates the learning rate. The weight update from the output to the hidden layer is defined as:

$$\Delta w_{pq} = -\eta \frac{\partial f}{\partial w_{pq}}$$

(38)

Analogically, we could write the weight compensation formula for the hidden to input layer:

$$\Delta w_{jk} = \eta \left( t_k - z_k \right) f'(net_k) y_j$$

(39)

where $k$ is defined as:

$$\delta_k = (t_k - z_k) f'(net_k)$$

(41)

These calculations are valid only for a three-layer NN, but they could be expanded to any number of layers. The simplicity of a three-layer NN does not give too much space for varying the parameters in a case of finding a more profitable solution. However, varying the learning rate, the number of neurons in the hidden layer, and the activation function could possibly improve the accuracy.

In our study, we use the sigmoid function as the activation function because of its simplicity. Some other characteristics of the sigmoid function that gives its advantage are smoothness, differentiability, nonlinearity, and saturation. A nonlinear function is desired because the NN in other cases will not be capable of supporting more than a two-layer network. Saturation, meaning that a function has its minimum and maximum, ensures that the training time is limited at some point. Smoothness and differentiability are
desired properties as the backpropagation process requires \( f(.) \) and \( f'(.) \) to calculate adjustments in the weight values.

**Practical evaluation of the two classification methods**

**Distance functions and Mahalanobis distance classification method**

One simple way of evaluating distance classifiers is for the case of two classes’ classification. For example, we can assume that two classes are statistically independent, having adequate expectations \( \mu_1 \) and \( \mu_2 \) and equal variance \( \sigma \). We can construct a discriminant function based on the equation:

\[
g_i(x) = \ln p(x; \omega_i) + \ln P(\omega_i)
\]  

We obtain the discriminant function in the form of:

\[
g_i(x) = -\frac{||x - \mu_i||^2}{2\sigma^2} + \ln P(\omega_i)
\]  

where \( ||.|| \) is the Euclidian norm, given by:

\[
||x - \mu_i||^2 = (x - \mu_i)^T (x - \mu_i)
\]  

The decision surface for this classifier can be expressed as:

\[
g_i(x) = g_j(x)
\]  

A much more difficult and interesting case is when the covariance matrices for the classes are identical but arbitrary. In this case, the discriminant function is defined as:

\[
g_i(x) = -\frac{1}{2} (x - \mu_i)^T \Sigma^{-1} (x - \mu_i) + \ln P(\omega_i)
\]
which is the same as the Mahalanobis distance between an arbitrary sample \( x \) and the observed patterns. Thus, the classification in this case is to measure the squared Mahalanobis distance from any given vector \( x \) to all present patterns and recognize \( x \) as part of the class with the smallest Mahalanobis distance. Note that there is always a smallest Mahalanobis distance in such a scenario, but how small to assign \( x \) to some of the present patterns? The answer to this question gives what is known as the chi-square test.

**Chi-square test**

The decision is classified as informative if it is far from random. The mathematical definition of the solution significance is given by the chi-square test. Chi-square statistics are defined as:

\[
\chi^2 = \sum_{i=1}^{2} \frac{n_{iL} - n_{iL}}{n_{iL}}
\]

where the above equation is valid for a two-class case (\( \omega_1 \) and \( \omega_2 \)) and \( n_{iL} \) is the number of patterns from \( \omega_i \) placed in the left category. Leaning to zero chi-square value means that the decision made is statistically correct. Table 3 gives the most popular values of chi-square for level of significance—0.001; 0.01; 0.05.
One important parameter, which helps to determine which value to use from the chi-square table, that is appropriate for certain cases is the degree of freedom (df). For the two-class case, if the number of patterns placed into the left category is known, all the other parameters are derived from that. Therefore, there is a one degree of freedom, which corresponds to the first row from Table 3. In this study, we have 9 categories; therefore, we consider 8 degrees of freedom to make a statistically correct decision. Thus, we have to choose from values in row number eight. We conclude that after calculating all Mahalanobis distances and selecting the smallest one, we should compare that distance to the 15.51 value (P=0.05). If the Mahalanobis distance is smaller than that value, then the decision is correct, and if not, then the investigated pattern does not belong to any of the present patterns.
Neural networks (NN) recognition method

Another classification method, investigated in this study, is the backpropagation feedforward neural network method. This method uses all the advantages of NN methods, and is simple to implement as a software solution. Because of its simplicity, it is a good candidate for performance measurements.

We choose a three-layer NN representing inputs, hidden, and output layers. The number of neurons of the input layer is equal to the number of the extracted features. The exact number of the hidden layer neurons will be investigated later, and some optimal number will be selected. The output layer has nine neurons, corresponding to the number of muscle groups, investigated in our study. Figure 13 shows a graphical representation of the selected network format.

![Backpropagation neural network structure](image)

**Figure 13** Backpropagation neural network structure. Connections from only one neuron per layer are shown; the rest of the neurons has similarly situated connection.
The theory behind BP NN has been described earlier and some concrete practical suggestions used in NN implementation are considered in this section. As discussed earlier, we have to find the most accurate way but still maintain good performance, which will be the base of NN implementation. We have already investigated (Chapter II) the best order of AR model representation for this research study and concluded that the AR model with order \( p = 15 \) is a good starting point. In other words, we should test the BP NN from 5 up to 15 inputs and conclude how the number of inputs influences the accuracy.

Figure 14 represents the dependency of the accuracy (opposite to recognition error) of the AR model order for a different number of hidden layer neurons. Smaller values of the AR model \( p \) mean low computational cost. The same is valid for a smaller number of neurons in the hidden layer. From the graphics observation, it can be empirically found that a good choice in terms of accuracy and computational cost corresponds to an AR model of order \( p = 5 \) and the number of neurons in the hidden layer is equal to 15. Although the corresponding results are relative, as the NN is initiated with random weights and the results obtained vary, they could be considered as a good estimation for the NN required structure.
Once the number of the hidden layer neurons is found, we can tweak other parameters of the BP NN to find the optimal variant for the solution. Another important parameter that can be investigated is the learning rate. Table 4 shows how the accuracy is altered with a change in the learning rate for fixed values of AR model $p = 5$ and the number of hidden layers equal to 15 (found above to be optimal for this project). As seen, the learning rate does influence the final result from recognition, but its role is not that crucial. All of the results are averaged as the NN does not show steady results because it depends on the initial values, which again are random. For typical problems addressed with the sigmoid function, it is found that a learning rate of $\eta \sim 0.1$ is often adequate as a first choice [13]. In our study, we assume the following learning rates: The learning rate of the hidden layer equal to 0.25 and the learning rate of the output layer equal to 0.35.

Figure 14  Accuracy vs. AR model order representation for different number of neurons in the hidden layer (From left to right; top to bottom: number of hidden layer neurons = 10; 15; 20; 25)
Another important parameter of the NN structure, which influences not only the accuracy but also performance, is the desired error. The smallest desired error leads to more accurate results, but the computational time may increase, which is why some treading should be considered when the goal is set. Table 5 shows how setting a desired error influences the final results. From this table and other practical results, it is found that a value of $1.e^{-5}$ is good enough for practical purposes.
<table>
<thead>
<tr>
<th>Desired error</th>
<th>Time for calculation [s]</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1e^{-5}</td>
<td>9.97</td>
<td>41.97</td>
</tr>
<tr>
<td>1e^{-4}</td>
<td>9.69</td>
<td>88.88</td>
</tr>
<tr>
<td>1e^{-3}</td>
<td>9.66</td>
<td>40.74</td>
</tr>
<tr>
<td>1e^{-2}</td>
<td>9.70</td>
<td>60.49</td>
</tr>
</tbody>
</table>
CHAPTER V
FEATURES EXTRACTION METHODS

Temporal feature extraction methods

One of the aims of this study is to find representative temporal parameters, whose calculations do not have high complexity.

Overview of temporal feature extraction methods

As discussed earlier, the EMG signal in its nature is a non-stationary signal, meaning that the signal somehow should be first “linearized” then processed with temporal methods. One of the easiest ways for treating EMG signals as linear is to divide the signal into segments and average the results obtained from all the segments.

The most popular features according to some authors representing raw EMG signals are:
1. Mean value

$$
\overline{X} = \frac{1}{N} \sum_{i=1}^{N} |x_i|
$$

(48)

The signals used in this study have a mean value close to zero. It is my personal opinion that this parameter is not relevant for recognition.

2. Mean absolute value

$$
\Delta \overline{X}_i = \Delta \overline{X}_{i+1} - \Delta \overline{X}_i
$$

(49)

where $\overline{X}_i$ is the mean value of the i-th sequence
3. Number of zero-crosses – that is, the number of times the signal curve crosses the zero-axes line.

4. Slope sign changes: The number of times the signal’s derivate changes its sign. My personal opinion is that this parameter is closely related to zero crosses, and either of the two parameters should be used.

5. Waveform length given by:

\[ \sum_{k=1}^{N} |\Delta x_k|, \text{defining } \Delta x_k = x_k - x_{k-1} \]  

(50)

6. Amplitude of the first burst

After averaging the raw EMG signal, it is passed through a FIR averaging filter with a Hamming window of length 300 ms. First, the saddle point of the resulting signal is used as a feature. Even though this feature may be representative, filtering is a time-consuming operation. This introduces a lot of calculations for each of the investigated signals, which would slow down the entire search process.

7. Square integral

The square integral in fact calculates the signal’s energy as:

\[ E = \int_{t_0}^{t_f} |x(t)|^2 dt \]  

(51)

The energy of the EMG signal is not evenly distributed in time; thus it is a good idea to separate the signal into segments, estimate the energy of each of the segments, and average the result obtained. Based on earlier investigated methods for feature extraction, this study proposes to use some additional features to refine the search and to improve the overall performance.
Newly proposed feature extraction techniques

In this thesis, we evaluate the following five parameters as extracted features capable for further recognition. These newly proposed features are selected in such a way that they are unique for certain types of signals but differ from a signal collected from another type of muscle.

1. Number of zero crosses (just the same as previous researchers use)
2. Pathway function (resemble the waveform length from above)
3. Area under the curve (this parameter is similar to the square integral, introduced previously). The difference with the square integral feature extraction method is that here, the signal is first normalized, then rectified, and finally the area under the curve is found. This helps to compare signals obtained with different hardware, i.e., different amplification might be used. The area under the curve uses the trapezoid rule for calculating the enclosed area, given by:

\[ A = \frac{1}{2} \sum_{i=1}^{N} |x_i|, \]  

(52)

where \(|x_i|\) is the absolute value of the consequent sample from the signal. The number of calculations for these parameters is almost the same as for the square integral.

In addition to the above features, two new features are proposed as feature parameters 4. Standard deviation (STD)

\[ STD = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2} \]  

(53)

4. Form factor

The form factor is another interesting feature, which characterizes the EMG signal. It is defined as:
where $\sigma_x$ is the variance of the signal and $x'$ and $x''$ correspond to the first and second order derivates of the signal. This feature was initially introduced to characterize EEG signals, but it was later used as an EMG identification characteristic.

As previously mentioned, in order to calculate the area under the curve feature, the signal has to be normalized. The normalization is done via:

$$x[i] = \frac{x[i] - \bar{x}}{STD}$$

(55)

However, the question remains as to how the proposed new features have been found to represent the investigated groups of signals. One answer can be found by looking at the table of the calculated parameters of an EMG signal. These newly introduced features differ from signal to signal, but are relatively constant for numerous sets of one particular muscle as shown in Tables 6 and 7.

<table>
<thead>
<tr>
<th>Zero crosses</th>
<th>STD</th>
<th>Pathway</th>
<th>FF</th>
<th>Area under the curve</th>
</tr>
</thead>
<tbody>
<tr>
<td>421</td>
<td>0.4898</td>
<td>349.6289</td>
<td>3.4045</td>
<td>1.2834</td>
</tr>
<tr>
<td>386</td>
<td>0.4965</td>
<td>323.5827</td>
<td>3.1880</td>
<td>1.2525</td>
</tr>
<tr>
<td>439</td>
<td>0.4756</td>
<td>327.5569</td>
<td>3.3697</td>
<td>1.2614</td>
</tr>
<tr>
<td>419</td>
<td>0.4457</td>
<td>315.8947</td>
<td>2.8657</td>
<td>1.2455</td>
</tr>
</tbody>
</table>

Table 6 A Extracted feature for EMG1

<table>
<thead>
<tr>
<th>Zero crosses</th>
<th>STD</th>
<th>Pathway</th>
<th>FF</th>
<th>Area under the curve</th>
</tr>
</thead>
<tbody>
<tr>
<td>562</td>
<td>0.2871</td>
<td>346.0796</td>
<td>2.6725</td>
<td>1.2834</td>
</tr>
<tr>
<td>514</td>
<td>0.3656</td>
<td>354.9084</td>
<td>2.7995</td>
<td>1.2525</td>
</tr>
<tr>
<td>483</td>
<td>0.5673</td>
<td>463.9417</td>
<td>3.3780</td>
<td>1.2614</td>
</tr>
<tr>
<td>520</td>
<td>0.2136</td>
<td>237.8562</td>
<td>2.8535</td>
<td>1.2455</td>
</tr>
</tbody>
</table>

Table 7 Extracted feature for EMG4

62
Spectral feature extraction methods

Overview of spectral features methods

Different techniques like Generic algorithm estimation of the frequency band of an EMG signal [6], Yule-Walker algorithm [7], utilizing of the direct transfer function [8], and classification based on energy spectra change [14], among others are used for spectral feature extraction. The goal of this study is to find an accurate and fast recognition algorithm. We are going to compare temporal and spectral features in order to come up with the most appropriate solution. The AR model is a good choice for the representation of EMG signals due to its easy implementation, fast execution, and accuracy in non-linear signal representation. Also, it is quite accurate for non-linear signal representation (as shown in Chapter II). More details for this method are provided below.

Parametric spectral estimation

When a model is used, it is assumed to include all data samples, not just those that appear. In this case, no windowing occurred and the resolution limitations may be overcome [15]. The approach work well with an EMG signal; using AR models may help to overcome ‘linearization’ by dividing the signals into segments.

In general, AR modeling is similar to least square minimization or least square prediction and the AR coefficients can be simply obtained using the Yule-Walker equations, where these equations can be solved using Levinson-Durbin recursion. The least-square prediction coefficients are obtained from:

\[
J = E \left\{ (x(n) - \hat{x}(n))^2 \right\}
\]  
(56)
Which are also given by:

\[ Rc = g \] (57)

Where \( R \) is \((p,p)\) autocorrelation matrix of \( x(n) \):

\[ R_{ij} = r(i-j) \quad \text{and} \quad g = [r(1), r(2), \ldots, r(p)]^T \] (58)

If the signal is generated via an AR(M) model, and provided \( p > M \), then the signal is now given by [15]:

\[ x(n) = a_1 x(n-1) + a_2 x(n-2) + \ldots + a_M x(n-M) + w(n) \] (59)

while the prediction has the form

\[ \hat{x} = \sum_{i=1}^{p} c_i x(n-i) \] [15] (60)

Estimating the prediction error, we obtain:

\[ J = E\{ e^2(n) \} = E\left\{ \left( (a_1 - c_1) x(n-1) + \ldots + (a_M - c_M) x(n-M) + (0 - c_{M+1}) x(n-M-1) + \ldots \\
+ (0 - c_p) x(n-p) + w(n) \right)^2 \right\} \] (61)

It is proven that solution which minimizes \( J \) is:

\[ c = [a_1, a_2, \ldots, a_M, 0, \ldots 0]^T \] (62)

**AR model order selection**

The order section of the AR model is an important question as it defines the accuracy of the approximation. In general, a low order approximation creates a too smooth signal and if the order is too high some false peaks may occur in the
approximated signal’s spectrum. A high order approximation also slows down the calculation process, which results in undesired effect.

Akaike has defined two criteria, which could objectively determine the order \( p \). The first one is based on the Final Error Prediction (FPE), as described by:

\[
(FPE)_p = N^p \frac{N + p + 1}{N - p - 1}
\]  

(63)

where \( N \) is the number of the signal’s sample and \( P_p \) is the prediction error power. The order \( p \) is selected such that \( FPE_p \) has minimum for that \( p \). The second Akaike criterion uses the information theoretic function, given by:

\[
AIC_p = \ln(P_p) + 2 \frac{p + 1}{N}
\]  

(64)

And the optimum order corresponds to the order for which \( AIC_p \) is minimum. Figure 15 shows how the AIC changes when \( p \) varies in the interval [5…15]. This interval has been chosen based on \( p = 15 \) as discussed previously. The PSD function of all the EMG signals has a good approximation of the original signals’ PSDs. On the other hand, \( p = 15 \) requires vast amount of calculation; thus an order \( p \) equal or less to 15 would be appropriate. For orders less than \( p = 5 \), the signal becomes too smooth and does not accurately resemble the original signals.
Figure 15  Akaike criterion vs. AR model order representation (p vary from 5 to 15)

From Figure 15, it can be seen that the AIC has its minimum at different values of p for different signals. The AIC was calculated through randomly chosen sets from adequate EMG signal realization. It also seems like these signals predominantly have a local minimum at p = 5; close to the AIC global minimum. As shown in the previous chapter, an order of p = 5 was also sufficient when used in neural networks recognition.

Pros and cons of Yule-Walker model

The AR model can also be expressed by a transfer function via z-transformation. It is obvious that the transform function will only have poles, which are the roots of the following equation:
To ensure stability, all the poles of the AR model must be inside the unit circle. According to [16], a pole that lies on the unit circle resembles a harmonic process. Priestley states that the YW approximation may lead to poor parameter estimates, even for large signals, if the AR model has a pole near the unit circle [16].

Although it is simple and easy to implement, in general, the Yule-Walker estimation for autoregressive modeling is not appropriate to use, especially for cyclic signals, due to the instability (near singular) of the corresponding autocovariance matrix [16]. To verify the stability of the estimated parameters, we can substitute the AR coefficients into the above equation and see if the roots of the polynomial lie near the unity circle. Figure 16 corresponds to the locations of the poles for all EMG signals considered in this study. It is obvious that no roots lie on, or close to, the boundaries of the unit circle. Therefore, the YW model is suitable for modeling in this case.
Figure 16  Roots of the polynomial constructed by AR model coefficients lie into the unit circle for the entire test set.
CHAPTER VI
RESULTS AND DISCUSSION

Research goal and results workability

The main goal of this research study is to find the most appropriate method for searching signals in an archive and/or on a hard-drive. Thus, a very crucial parameter is the complexity of the selected algorithm. A more complex algorithm will take more time for calculation and will slow the entire search process. A second parameter is accuracy, but it may not have such importance as the execution time. Even if some of the signals are not recognized correctly, the user will manually select which of the signals will be further processed.

Data collection environment

The data used in this study have been downloaded from the following ftp site: ftp.emgsrus.com/sample/emgdata, as a courtesy of Motion Lab Systems, Inc and are freely posted for educational use. These data represent raw EMG signals collected from North American male teen. All files contain identical analog channel data. The file sets consist of C3D (binary) files, as well as DST (ASCII) and CSV (comma separated value) files that have been created from the C3D file data. In our study, we use CSV file format as the input data source. The scanning frequency as reported is 800 Hz. Nine trials of the following muscles have been investigated:

Adductors
Gastrocnemius
Gluteus Maximus
Gluteus Medialis
Lateral Hamstrings
Medial Hamstrings
Rectus Femoris
Tibialis Anterior
Vastus Lateralis,

These are denoted as EMG1, EMG2, …, EMG9 for simplicity. The signal contains 2,400 samples with an overall length of 3.0 seconds. The data were collected from the right side of the above muscle groups. Figure 17 shows all of the nine trials of the test signal (EMG1).

Figure 17  EMG1 signal- nine trails.

**Algorithm implementation**

Two feature extraction methods along with two recognition algorithms and their combinations have been investigated. Feature extraction methods are characterized as
temporal and spectral. When the number of signals required to be recognized is small (less than three), it is enough to have only two features. This is illustrated in Figure 18. This figure represents three signals EMG1, EMG2, and EMG3, depicted by the pathway value and the zero-crossing rate. It is easy to observe that these two features are enough to distinguish these signals.

![Figure 18 Features distribution for EMG1, EMG2 and EMG3 training set of signals](image)

Since our goal is to recognize all nine signals, adding more signals for recognition makes recognition more complex due to features overlapping, as shown in Figure 19. The corresponding error as result of the recognition is listed in Table 8. It is clearly seen the resulting error values are quite high for most of the EMG signals under consideration.
Figure 19  Features distribution for EMG1 through EMG9 training set of signals

Table 8  Error rates from recognition, using two factors (x1 = Zero Crosses, x2 = STD) and Mahalanobis distance recognition algorithm

<table>
<thead>
<tr>
<th>Signal</th>
<th>% Error rate (simple Mahalonobis distance)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>55.6</td>
</tr>
<tr>
<td>EMG2</td>
<td>33.3</td>
</tr>
<tr>
<td>EMG3</td>
<td>55.6</td>
</tr>
<tr>
<td>EMG4</td>
<td>44.4</td>
</tr>
<tr>
<td>EMG5</td>
<td>55.6</td>
</tr>
<tr>
<td>EMG6</td>
<td>11.1</td>
</tr>
<tr>
<td>EMG7</td>
<td>11.1</td>
</tr>
<tr>
<td>EMG8</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG9</td>
<td>22.2</td>
</tr>
</tbody>
</table>

An attempt to reduce the error rates can be achieved by increasing the number of extracted features. It can be shown that adding one more feature improves the overall error distribution. This is clearly illustrated in Table 9 where by adding only one
additional feature $x_3$ (Pathway function) the error rate drops significantly and become even zero for some of the signals. Therefore, adding five features, as mentioned previously, would be enough to represent all of the muscle groups, will further reduce the error rates.

The five temporal features: the number of zero crosses, standard deviation, pathway function, form factor, and area under the curve, are in general good candidates for feature representation of the investigated signals. A time series feature extraction representation for EMG1 is illustrated in Table 10. Another important problem, as discussed earlier, is that all of the temporal features are relevant if the signal is divided into segments. The question remains as to what the segment length should be; the answer can be obtained empirically. As shown in Table 11, the best results for these particular signals occur when the window length is 500 samples long.

<table>
<thead>
<tr>
<th>Signal</th>
<th>% Error rate (simple Mahalonobis distance)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>33.3</td>
</tr>
<tr>
<td>EMG2</td>
<td>22.2</td>
</tr>
<tr>
<td>EMG3</td>
<td>33.3</td>
</tr>
<tr>
<td>EMG4</td>
<td>44.4</td>
</tr>
<tr>
<td>EMG5</td>
<td>11.1</td>
</tr>
<tr>
<td>EMG6</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG7</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG8</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG9</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Table 10  Time series feature extraction representation for EMG1

<table>
<thead>
<tr>
<th>Feature</th>
<th>476</th>
<th>491</th>
<th>504</th>
<th>519</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero cross</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard deviation</td>
<td>0.8641</td>
<td>0.8153</td>
<td>0.8034</td>
<td>0.6077</td>
</tr>
<tr>
<td>Pathway</td>
<td>816.6201</td>
<td>735.2214</td>
<td>737.1399</td>
<td>555.3110</td>
</tr>
<tr>
<td>Form factor</td>
<td>2.5481</td>
<td>2.5368</td>
<td>2.5748</td>
<td>2.4929</td>
</tr>
<tr>
<td>Area under the curve</td>
<td>1.4893</td>
<td>1.3750</td>
<td>1.4031</td>
<td>1.3802</td>
</tr>
</tbody>
</table>

Table 11  Error dependency on window length of signal’s segment

<table>
<thead>
<tr>
<th>Win length (Hanning)</th>
<th>Mean error</th>
<th>Std Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>24.69</td>
<td>10.79</td>
</tr>
<tr>
<td>300</td>
<td>19.75</td>
<td>13.35</td>
</tr>
<tr>
<td>500</td>
<td>16.04</td>
<td>9.79</td>
</tr>
<tr>
<td>700</td>
<td>16.04</td>
<td>12.55</td>
</tr>
<tr>
<td>900</td>
<td>27.16</td>
<td>8.07</td>
</tr>
</tbody>
</table>

The results from Table 11 were obtained using the five temporal features and the Mahalanobis distance recognition method. That same window length is also used for spectral features extraction. All the segments are windowed with the Hanning function:

\[ w(n) = 0.5 \left( 1 - \cos\left(\frac{2\pi n}{N}\right) \right), \quad 0 \leq n \leq N \]

(66)

where the window length is N+1 samples. The above equation is called the periodic Hanning function and is useful for DFT/FFT spectral analysis. The second feature extraction method, the spectral method, relies on the Yule-Walker equation with an optimum order of p = 5, as found earlier to be suitable for these type of signals. For the actual recognition, two types of algorithms have been used:

Mahalanobis distance recognition

Backpropagation neural network
These feature extraction methods along with the recognition algorithms allow for the following combinations:

- Time series features with the Mahalanobis distance recognition
- Spectral estimation based on AR modelling with NN recognition
- Time series features with NN recognition
- Spectral estimation based on AR modeling with the Mahalanobis distance recognition

Both MATLAB and Microsoft Visual Studio 2005 have been used for the implementation of these methods. MATLAB sources have been used to produce reference values for the training sets and optimization practices. C# has been used as a programming language under .NET 2.0 environment. The software used can be divided into three main levels: user interface, business logic, and data. Note that it is a good practice to mention how the current work has been spread between these levels.

The current study aims to find an appropriate algorithm for data search; therefore, it does not need a comprehensive user interface. The test application is organized as a console application with no user input. Once the appropriate algorithm is selected, then the user interface should be more user’s friendly. In such a case, the user should be prompted to select several signals that he/she considers as representative of a new group. After the selection of these signals, their temporal or spectral features will be extracted and stored in memory (or registry). If for example the NN algorithm is chosen, here is the moment for that algorithm to be trained. Every time a new group is defined, the NN should be trained. No less than six signals should be allowed as training sets for a certain group. More signal selection may make the use of such software difficult and, on the other hand, it may smear the actual recognition.
The data level is organized in class Data, as part of the projects “namespace TestNN”. There are several methods for obtaining the training sets. The training sets are organized into double arrays where the number of rows is equal to the number of signal groups (nine), while the number of columns is equal to the training sets (six in this case). Two other methods are defined: one to get the training data for the Mahalanobis distance algorithm and another to get the training data for the NN algorithm. The Mahalanobis distance matrix contains 5 rows (number of extracted features) and six columns (number of training sets). Once the features for the investigated signal are calculated, then, another function calculates the Mahalanobis distance between the current 5 elements array and a 5x6 matrix. That process is repeated for all the signals present and the smallest value from the calculated distances is chosen. The index of that distance corresponds to the index of the investigated signal. If not, the current results are considered wrong. The number of wrong results is divided to the number of signals (nine in this case), which produces the recognition error. Finally, the mean and standard deviation of this error array is computed and used as statistical measures of how well the method performs.

Time performance is an important parameter and is one of the reasons for the using c# for implementation. It has built-in classes for measuring the performance time and it produces executable files, in addition to MATLAB environment, which emulates the environment. Performance measurement has been done using the ‘Stopwatch’ predefined class. The Mahalanobis distance implementation along with functions for calculating the mean, STD, form factor, and area are stored in one class. The neural network implementation resides in another class. All the logic is executed from the ‘main ()’ function in Program class.
Actual results

Tables 12 and 13 represent the actual results obtained from all the methods used in this investigation. To decide which method is best, three parameters are used for this purpose. These are the mean error produced from the recognition error for all the signals, the standard deviation of the error, and the execution time of the algorithm. The criterion for best signal representation can be found by calculating the Euclidian distance from the parameters (mean error; STD of the error; time for execution) point to the (0;0;0) point, as the goal for all three parameters is to lean to zero. The smallest such Euclidian distance would objectively judge the best representation of the investigated muscles.

Table 12 Temporal feature extraction method results

<table>
<thead>
<tr>
<th>Recognition method: Mahalanobis distance</th>
<th>Recognition error [%]</th>
<th>Recognition method: neural network</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>11.11</td>
<td>EMG1</td>
<td>11.11</td>
</tr>
<tr>
<td>EMG2</td>
<td>22.22</td>
<td>EMG2</td>
<td>11.11</td>
</tr>
<tr>
<td>EMG3</td>
<td>11.11</td>
<td>EMG3</td>
<td>22.22</td>
</tr>
<tr>
<td>EMG4</td>
<td>22.22</td>
<td>EMG4</td>
<td>44.44</td>
</tr>
<tr>
<td>EMG5</td>
<td>11.11</td>
<td>EMG5</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG6</td>
<td>22.22</td>
<td>EMG6</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG7</td>
<td>11.11</td>
<td>EMG7</td>
<td>11.11</td>
</tr>
<tr>
<td>EMG8</td>
<td>0.0</td>
<td>EMG8</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG9</td>
<td>33.33</td>
<td>EMG9</td>
<td>22.22</td>
</tr>
<tr>
<td>Mean error [%]</td>
<td>16.04</td>
<td></td>
<td>13.58</td>
</tr>
<tr>
<td>Std of the error [%]</td>
<td>9.79</td>
<td></td>
<td>14.46</td>
</tr>
<tr>
<td>Time for recognition [sec]</td>
<td>7.56</td>
<td></td>
<td>7.48</td>
</tr>
</tbody>
</table>
Table 13 Spectral feature extraction method results

<table>
<thead>
<tr>
<th>Recognition method: Mahalanobis distance</th>
<th>Recognition error [%]</th>
<th>Recognition method: neural network</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>11.11</td>
<td>EMG1</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG2</td>
<td>33.33</td>
<td>EMG2</td>
<td>11.11</td>
</tr>
<tr>
<td>EMG3</td>
<td>33.33</td>
<td>EMG3</td>
<td>11.11</td>
</tr>
<tr>
<td>EMG4</td>
<td>22.22</td>
<td>EMG4</td>
<td>11.11</td>
</tr>
<tr>
<td>EMG5</td>
<td>11.11</td>
<td>EMG5</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG6</td>
<td>33.33</td>
<td>EMG6</td>
<td>22.22</td>
</tr>
<tr>
<td>EMG7</td>
<td>22.22</td>
<td>EMG7</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG8</td>
<td>22.22</td>
<td>EMG8</td>
<td>0.0</td>
</tr>
<tr>
<td>EMG9</td>
<td>22.22</td>
<td>EMG9</td>
<td>11.11</td>
</tr>
<tr>
<td>Mean error [%]</td>
<td>23.45</td>
<td></td>
<td>7.407</td>
</tr>
<tr>
<td>Std of the error [%]</td>
<td>8.68</td>
<td></td>
<td>7.85</td>
</tr>
<tr>
<td>Time for recognition [sec]</td>
<td>7.67</td>
<td></td>
<td>7.64</td>
</tr>
</tbody>
</table>

Four combinations are presented in this study; the temporal feature extraction methods in combination with the Mahalanobis distance and the Neural Network recognition algorithms (Table 12) and the spectral methods in combination with two above methods for recognition (Table 13). During the NN development and testing, an interesting fact is observed. That is, when setting the NN outputs to ‘0’ and ‘1’, the network is trained very fast (<< 1s), but it is very inaccurate (Mean error = 88.2%). On the other hand, setting the output values to 0.8 for the expected output and 0.015 for the rest increases the training time (~ 4 s) and the accuracy (Mean error ~14%). Also, relatively high values for the zero-cross rate, pathway, and area under the curve make the NN produce an unacceptably high error (Mean error was above 88%). When such values are maintained (by linear transformation) small (< 5), the NN produces accurate results. Six sets of signals for every muscle are used as training sets. Due to the small amount of training sets, only nine per muscle training sets are included as validation sets as well. In other words, the error array from recognition is produced for every signal and every set.
The number of wrongly recognized sets from every signal is divided by 9 (total number of sets); thus we obtain an array of errors which differ from signal to signal. The final representation of the recognition error is given by the mean value and the STD of the error array.

**Cross-validation analysis and kappa coefficient**

Teaching with six signals, but recognizing nine is a good way to show the tendency of which method is preferable, but does not give a real estimation of how accurate every of these methods is.

More relevant estimation is given by the cross-validation estimation. This method is used to estimate how good the model for an independent dataset is. There are different types of cross-validation such as: K-fold, k x 2 cross-validation, and leave-one-out cross validation.

Leave-on-out cross validation is more expensive than other methods in terms of the calculation time, but it gives more accurate results. In this method one sample is left as validation sample, but the system is taught with the rest of the samples. Then this sample is used to validate the created model.

In this project we have nine groups of muscles and every group has nine sets of samples. To apply the cross-validation estimation one set of data from every group will be used as a validation set, but the rest of sets are used as training sets. A different number of features will be used in that estimation: five spectral, five temporal, seven mixed (five spectral and two temporal), and ten mixed (five spectral and five temporal features) along with two recognition methods: The Mahalanobis distance and NN recognition methods.
The Fleiss’ kappa coefficient is also used to measure the relevancy of the cross-validation for the above methods. If the given raters agree on all groups, then the kappa coefficient is equal to 1, if no rater agree with other raters then the kappa coefficients is ≤ 0.

Landis and Koch (1977) proposed the following interpretation of kappa coefficients (See Table 14)

Table 14 Landis and Koch interpretation of kappa coefficients

<table>
<thead>
<tr>
<th>K</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
<td>Poor agreement</td>
</tr>
<tr>
<td>0.0 – 0.20</td>
<td>Slight agreement</td>
</tr>
<tr>
<td>0.21 – 0.40</td>
<td>Fair agreement</td>
</tr>
<tr>
<td>0.41 – 0.60</td>
<td>Moderate agreement</td>
</tr>
<tr>
<td>0.61 – 0.80</td>
<td>Substantial agreement</td>
</tr>
<tr>
<td>0.81 – 1.00</td>
<td>Almost perfect agreement</td>
</tr>
</tbody>
</table>

For the kappa coefficient estimation we can define the followings:

N – Total number of subjects to be estimated

n- Number of ratings for every subject

k- Number of categories into which a subject could be categorized.

In the case of this study, N = 9 subjects (groups of signals), n = 9 as cross validation methods require to make every set validation set, and k = 9 as we define 9 categories of signals.

The Fleiss’ kappa coefficient is defined as:
The Numerator and denominator values are defined as follow:

- assignment proportions for every \( j \)-th category:

\[
P_j = \left( \frac{1}{Nn} \right) \sum_{j=1}^{k} n_{ij}
\]  

(68)

\[
\overline{P}_e = \sum_{j=1}^{k} \left( p_{ij} \right)^2
\]  

(69)

- raters agreement extend for every \( i \)-th subject

\[
P_i = \frac{1}{n \times (n-1)} \left[ \sum_{j=1}^{k} (n_{ij})^2 - n \right]
\]  

(70)

\[
\overline{P} = \left( \frac{1}{N} \right) \sum_{i=1}^{N} P_i
\]  

(71)

Tables 15 through 21 show the exact results from cross-validation of all investigated methods and features.
### Table 15  Ten mixed features, NN recognition method

<table>
<thead>
<tr>
<th></th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 1</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 2</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 4</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>Signal 9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

\[ p_j = 0.0988, 0.111, 0.0988, 0.0988, 0.0988, 0.1235, 0.1358, 0.111, 0.1235, \]
\[ \overline{P_e} = 0.1126; \]
\[ P_i = 0.5833, 0.7778, 0.4167, 0.4444, 0.7778, 1.0, 1.0, 0.7778, 0.5833 \]
\[ \overline{P} = 0.7068 \]
\[ \kappa = \frac{\overline{P} - \overline{P_e}}{1 - \overline{P_e}} = (0.7068 - 0.1126)/(1 - 0.1126) = 0.6696 \]

### Table 16  Five temporal features, NN recognition method

<table>
<thead>
<tr>
<th></th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 1</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 2</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 4</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Signal 5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Signal 6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Signal 9</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

\[ p_j = 0.1852, 0.0864, 0.1111, 0.0617, 0.0864, 0.1111, 0.1358, 0.1111, 0.1111 \]
\[ \overline{P_e} = 0.1209; \]
\[ P_i = 1.00, 0.3611, 0.1944, 0.1944, 0.5833, 0.4444, 1.00, 0.5833, 0.3056 \]

\[ \overline{P} = 0.5185; \]

\[ \kappa = (0.5185 - 0.1209)/(1 - 0.1209) = 0.4523; \]

**Table 17**  Five spectral features, NN recognition method

<table>
<thead>
<tr>
<th>Signal 1</th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 2</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Signal 5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 6</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>Signal 9</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ \rho_j = 0.0741, 0.1358, 0.1481, 0.1358, 0.1111, 0.0617, 0.1605, 0.0988, 0.0741 \]

\[ \overline{P}_r = 0.1215; \]

\[ P_1 = 0.3056, 0.7778, 0.3333, 0.5833, 0.7778, 0.2500, 0.7778, 0.4444, 0.4167 \]

\[ \overline{P} = 0.5185; \]

\[ \kappa = (0.5185 - 0.1215)/(1 - 0.1215) = 0.4519; \]
Table 18  Seven (5 spectral + 2 temporal) features, NN recognition method

<table>
<thead>
<tr>
<th></th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 1</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 2</td>
<td>0</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 6</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Signal 9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ p_j = 0.1481, 0.0988, 0.1235, 0.0988, 0.0741, 0.1111, 0.0988, 0.1481, 0.0988 \]

\[ \overline{P} = 0.1160; \]

\[ P_l = 0.7778, 0.4167, 1.00, 0.5972, 0.5000, 0.7778, 0.7778, 0.7778, 0.5972 \]

\[ \overline{P} = 0.6914; \]

\[ \kappa = (0.6914 - 0.1160)/(1 - 0.1160) = 0.6509; \]

Table 19  Seven (5 spectral + 2 temporal) features, Mahalanobis distance recognition method

<table>
<thead>
<tr>
<th></th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 1</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 2</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 4</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Signal 9</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

\[ p_j = 0.1728, 0.1235, 0.1235, 0.1111, 0.1358, 0.0864, 0.0494, 0.0988, 0.0741 \]
\[ \bar{P}_e = 0.1163; \]
\[ P_e = 0.4167, 0.3056, 0.4444, 0.0972, 0.1111, 0.1944, 0.4167, 0.5833, 0.3194 \]
\[ \bar{P} = 0.3210; \]
\[ \kappa = (0.3210 - 0.1163)/(1 - 0.1163) = 0.2316 \]

Table 20  Five (5 spectral) features, Mahalanobis distance recognition method

<table>
<thead>
<tr>
<th></th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 1</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Signal 2</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Signal 5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Signal 6</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>Signal 9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ p_j = 0.0741, 0.0988, 0.1605, 0.0988, 0.0864, 0.0617, 0.1975, 0.1235, 0.0988 \]
\[ \bar{P}_e = 0.1261; \]
\[ P_e = 0.3056, 0.6111, 0.6111, 0.5833, 0.5833, 0.3611, 0.5139, 0.5139, 0.4167 \]
\[ \bar{P} = 0.5000 \]
\[ \kappa = (0.5000 - 0.1261)/(1 - 0.1261) = 0.4279 \]
Table 21: Five (5 temporal) features, Mahalanobis distance recognition method

<table>
<thead>
<tr>
<th></th>
<th>EMG1</th>
<th>EMG2</th>
<th>EMG3</th>
<th>EMG4</th>
<th>EMG5</th>
<th>EMG6</th>
<th>EMG7</th>
<th>EMG8</th>
<th>EMG9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal 1</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 2</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Signal 3</td>
<td>0</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 6</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Signal 7</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Signal 8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Signal 9</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

\[ p_1 = 0.0741, 0.1235, 0.1605, 0.1852, 0.1605, 0.0617, 0.0864, 0.0617, 0.0864 \]

\[ \overline{P} = 0.1291; \]

\[ P_1 = 0.3333, 0.3611, 0.5833, 0.3056, 0.7778, 0.1806, 0.5139, 0.3611, 0.2222 \]

\[ \overline{P} = 0.4043; \]

\[ \kappa = \frac{(0.4043 - 0.1291)}{(1 - 0.1291)} = 0.3160; \]

Table 22 compares kappa coefficients for the used methods and features. It is clear that 10 mixed features along with NN recognition method is most relevant of all investigated methods. Also, its kappa coefficient is in the range \([0.61 – 0.8]\), defined by Landis and Koch as “substantial agreement”. The NN recognition method shows good results, but it requires training. In some cases the training could take substantial amount of time, even there are cases when practically it is infinite. In this project the training times are observable, even 10 mixed features training time is in the range of five features training time, but the 10 mixed features have much better accuracy as shown in Table 22.

The Mahalanobis distance method requires the number of rows (training sets) in the Mahalanobis matrix to be bigger than the number of columns (features). As we have only eight training sets, the 10 features method cannot be estimated in this study.
Table 22  Kappa coefficients

<table>
<thead>
<tr>
<th></th>
<th>Mahal</th>
<th>NN</th>
<th>Training Time [sec] (for NN only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 temporal</td>
<td>0.3160</td>
<td>0.4523</td>
<td>7</td>
</tr>
<tr>
<td>5 spectral</td>
<td>0.4279</td>
<td>0.4519</td>
<td>5</td>
</tr>
<tr>
<td>7 mixed</td>
<td>0.2316</td>
<td>0.6509</td>
<td>53</td>
</tr>
<tr>
<td>10 mixed</td>
<td>------</td>
<td>0.6696</td>
<td>10</td>
</tr>
</tbody>
</table>

The overall accuracy is another measurement, which gives objective estimation as to which method is preferable. It is found as the number of all correctly recognized sets is divided by the number of all sets (9 signals x 9 sets). Results are shown in the Table 23.

Table 23  Overall accuracy [%]

<table>
<thead>
<tr>
<th></th>
<th>Mahal</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 temporal</td>
<td>59</td>
<td>54</td>
</tr>
<tr>
<td>5 spectral</td>
<td>69</td>
<td>69</td>
</tr>
<tr>
<td>7 mixed</td>
<td>53</td>
<td>79</td>
</tr>
<tr>
<td>10 mixed</td>
<td>------</td>
<td>83</td>
</tr>
</tbody>
</table>

The overall accuracy gives a general estimation of the method, but does not give what’s the accuracy for every investigated group of signals. To find out the accuracy achieved for every group and the average accuracy per signal, we have to divide the number of correctly recognized sets for every signal, divided by the number of all
signal’s sets. Tables 24 through 27 shows the achieved accuracy per signal along with the average accuracy and standard deviation of the accuracy.

Table 24 Average accuracy and standard deviation of the measured accuracy (5 Spectral features)

<table>
<thead>
<tr>
<th>Recognition method: Mahalanobis distance</th>
<th>Accuracy [%]</th>
<th>Recognition method: neural network</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>55.5</td>
<td>EMG1</td>
<td>55.6</td>
</tr>
<tr>
<td>EMG2</td>
<td>77.8</td>
<td>EMG2</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG3</td>
<td>77.8</td>
<td>EMG3</td>
<td>55.6</td>
</tr>
<tr>
<td>EMG4</td>
<td>77.8</td>
<td>EMG4</td>
<td>77.8</td>
</tr>
<tr>
<td>EMG5</td>
<td>77.8</td>
<td>EMG5</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG6</td>
<td>55.5</td>
<td>EMG6</td>
<td>33.3</td>
</tr>
<tr>
<td>EMG7</td>
<td>66.7</td>
<td>EMG7</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG8</td>
<td>66.7</td>
<td>EMG8</td>
<td>66.7</td>
</tr>
<tr>
<td>EMG9</td>
<td>66.7</td>
<td>EMG9</td>
<td>66.7</td>
</tr>
<tr>
<td>Mean accuracy</td>
<td>69.14</td>
<td></td>
<td>69.15</td>
</tr>
<tr>
<td>Std of the accuracy [%]</td>
<td>9.2</td>
<td></td>
<td>19.07</td>
</tr>
</tbody>
</table>
Table 25  Average accuracy and standard deviation of the measured accuracy (5 Temporal features)

<table>
<thead>
<tr>
<th>Recognition method: Mahalanobis distance</th>
<th>Accuracy [%]</th>
<th>Recognition method: neural network</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>55.6</td>
<td>EMG1</td>
<td>100</td>
</tr>
<tr>
<td>EMG2</td>
<td>55.6</td>
<td>EMG2</td>
<td>55.6</td>
</tr>
<tr>
<td>EMG3</td>
<td>77.8</td>
<td>EMG3</td>
<td>44.4</td>
</tr>
<tr>
<td>EMG4</td>
<td>55.6</td>
<td>EMG4</td>
<td>11.1</td>
</tr>
<tr>
<td>EMG5</td>
<td>88.9</td>
<td>EMG5</td>
<td>77.8</td>
</tr>
<tr>
<td>EMG6</td>
<td>33.3</td>
<td>EMG6</td>
<td>66.7</td>
</tr>
<tr>
<td>EMG7</td>
<td>66.7</td>
<td>EMG7</td>
<td>100</td>
</tr>
<tr>
<td>EMG8</td>
<td>55.6</td>
<td>EMG8</td>
<td>77.8</td>
</tr>
<tr>
<td>EMG9</td>
<td>44.4</td>
<td>EMG9</td>
<td>55.6</td>
</tr>
<tr>
<td>Mean accuracy [%]</td>
<td>59.2</td>
<td></td>
<td>65.4</td>
</tr>
<tr>
<td>Std of the accuracy [%]</td>
<td>16.6</td>
<td></td>
<td>28.02</td>
</tr>
</tbody>
</table>

Table 26  Average accuracy and standard deviation of the measured accuracy 7 mixed features (five spectral + two temporal)

<table>
<thead>
<tr>
<th>Recognition method: Mahalanobis distance</th>
<th>Accuracy [%]</th>
<th>Recognition method: neural network</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>66.7</td>
<td>EMG1</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG2</td>
<td>55.5</td>
<td>EMG2</td>
<td>66.7</td>
</tr>
<tr>
<td>EMG3</td>
<td>66.7</td>
<td>EMG3</td>
<td>100</td>
</tr>
<tr>
<td>EMG4</td>
<td>33.3</td>
<td>EMG4</td>
<td>77.8</td>
</tr>
<tr>
<td>EMG5</td>
<td>33.3</td>
<td>EMG5</td>
<td>66.7</td>
</tr>
<tr>
<td>EMG6</td>
<td>66.7</td>
<td>EMG6</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG7</td>
<td>44.4</td>
<td>EMG7</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG8</td>
<td>77.8</td>
<td>EMG8</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG9</td>
<td>44.4</td>
<td>EMG9</td>
<td>77.8</td>
</tr>
<tr>
<td>Mean accuracy [%]</td>
<td>54.31</td>
<td></td>
<td>82.73</td>
</tr>
<tr>
<td>Std of the accuracy [%]</td>
<td>16.17</td>
<td></td>
<td>11.25</td>
</tr>
</tbody>
</table>
Table 27  Average accuracy and standard deviation of the measured accuracy 10 mixed features (Five spectral + five temporal)

<table>
<thead>
<tr>
<th>Recognition method: Mahalanobis distance</th>
<th>Accuracy [%]</th>
<th>Recognition method: neural network</th>
<th>Recognition error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMG1</td>
<td>---</td>
<td>EMG1</td>
<td>77.8</td>
</tr>
<tr>
<td>EMG2</td>
<td>---</td>
<td>EMG2</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG3</td>
<td>---</td>
<td>EMG3</td>
<td>66.7</td>
</tr>
<tr>
<td>EMG4</td>
<td>---</td>
<td>EMG4</td>
<td>66.7</td>
</tr>
<tr>
<td>EMG5</td>
<td>---</td>
<td>EMG5</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG6</td>
<td>---</td>
<td>EMG6</td>
<td>100</td>
</tr>
<tr>
<td>EMG7</td>
<td>---</td>
<td>EMG7</td>
<td>100</td>
</tr>
<tr>
<td>EMG8</td>
<td>---</td>
<td>EMG8</td>
<td>88.9</td>
</tr>
<tr>
<td>EMG9</td>
<td>---</td>
<td>EMG9</td>
<td>77.8</td>
</tr>
<tr>
<td>Mean accuracy [%]</td>
<td>---</td>
<td></td>
<td><strong>83.96</strong></td>
</tr>
<tr>
<td>Std of the accuracy [%]</td>
<td>---</td>
<td></td>
<td><strong>12.54</strong></td>
</tr>
</tbody>
</table>

A more clear view of the accuracy obtained from the cross-validation method is illustrated in Figures 20 and 21.

Figure 20  Average accuracy [%] (10 mixed features not applicable with Mahalanobis distance)
Figure 21  STD of accuracy [%] (10 mixed features not applicable with Mahalanobis distance)

**Discussion**

We have to make an objective judgment of the results obtained in terms of accuracy, execution time, and difficulties found during the developmental process. Calculating the Euclidian distance between the parameters (mean error, STD of error, and execution time) point to the (0;0;0) point shows which method is closer to the ideal recognition method. Note that the ideal method produces 0.0% error and its execution time is unnoticeably small. Table 28 illustrates the results obtained.

**Table 28**  Euclidian distance between the parameters (Mean error; STD of error; Time for execution) point and (0;0;0) point

<table>
<thead>
<tr>
<th></th>
<th>Mahalanobis distance recognition</th>
<th>Neural network recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Five temporal features</td>
<td>20.25</td>
<td>21.2</td>
</tr>
<tr>
<td>Five spectral features</td>
<td>26.15</td>
<td><strong>13.22</strong></td>
</tr>
</tbody>
</table>

From the above results, we can objectively conclude that it is best to use spectral feature extraction with a combination of the neural network algorithm for recognition,
although the NN algorithm has some weaknesses, which may discredit its performance. Some of the NN disadvantages are:

- The NN does not always produce the same results, as the initial values for weights are random. Thus, claiming that the error produced from the NN is ‘usually’ below 15% does not mean that it is always in this range.
- The NN requires additional time for training, which is about 4-5 seconds for 9x9 sample sets with 5 inputs, 15 nodes in the hidden layer, and 9 outputs. Although the training is done only once there may be cases where the NN may never finish or it continues undesirably for a long time.
- For most cases, having more training sets would produce better results in terms of accuracy. However, as for this particular study, the user will be required to select some representative signals from the group, which is a tedious and slow process; thus pushing the user to select as many signals as possible.

On the other hand, the mix of temporal and spectral features, as shown, introduces a new level of accuracy which helps the NN approach to overcome its weaknesses. Increasing the number of features not always lead to better results. This is shown with the seven mixed features case used along with Mahalanobis distance recognition method.

For ten mixed features and NN recognition method, relatively high kappa coefficient (0.669) shows “substantial agreement” according Landis and Koch table. No other of investigated methods has such objectively proven accuracy. Learning time, one of the main obstacles for NN to be considered for practical use is relatively small for ten mixed features case- only 10 seconds. In practice that means that for every newly introduced group, user will wait about ten seconds for such group to be created and become functional. This is observable time for this application.
CHAPTER VII
CONCLUSION

Summary

All evaluated methods show good time performance and any of them could be used with certain assumptions as the base algorithm for searching similar signals in the archive and/or on a hard disk.

In conclusion, the NN with 10 features has the biggest accuracy for all investigated methods, relatively small accuracy STD, highest kappa coefficient, as well very fast learning curve. Therefore it can be considered as most appropriate for recognizing and organizing EMG signals related to human gait.

The NN problems mentioned in the previous chapter are not relevant for this particular case because as proven, learning has finite time and that time is observable- 10 seconds for 10 features case.

This study introduces a new application field for EMG signal recognition, organizing and searching of particular type of already collected signals designated for further processing or backup. Most of the current studies are oriented to prosthesis control, EMG data compression and recognition in terms of searching patterns in signal for diagnostic purposes. In search of the most efficient method for recognizing signals designated for further processing, a new promising combination of features has been introduced – mix of temporal and spectral features. Two of the newly introduced
temporal features- STD and Form factor have not been used for recognizing EMG gait signals and this study confirm their usefulness.

**Limitations and obstacles**

The recognition test should involve different groups of people in terms of age, sex, and training abilities. Some researchers suggest using from 10 to 400 participants. In order to have a more objective test, the number of training sets should be increased. In the current study, 6 to 9 sets are used for training, but recognition is applied to all 9 sets. That could lead to errors as some groups of signals are used for training and validation.

Some general limitations of the implementation of the spectral methods are associated with the fact that, for certain cases, the autocovariance matrix can be singular or near singular. However, as shown in this study and for this group of investigated muscles, we can be certain that the autocovariance matrix is not singular, but applying this method to other groups of muscles or to other signals may not give a stable result.

**Future development**

Future investigation of the proposed methods can be performed with larger groups of people, as well with comparison to some other recognition methods with the intention to find the best fit for the search algorithm. The idea of the search for certain groups of muscles can be feasibly extended to other groups of collected muscles as well to other signal such as the electro-encephalography (EEG). This same idea can also be applied to signals obtained from force plate measurements.
REFERENCES


[14] Xiao Hu, Ping Yu, Qun Yu, Waixi Liu, Jian Qin, “Classification of surface EMG signal based on energy spectra change”, 2008 International Conference on BioMedical Engineering and Informatics


APPENDIX A

MATLAB PROGRAM FOR INVESTIGATING THE OPTIMAL AR MODEL,
ACCOUNTING DIFFERENCES BETWEEN PSDS OF THE ORIGINAL
AND THE MODELED SIGNAL
clear; clc;
figure; hold on;

%Load the test signal
signal = load('10_EMG10_TEST_Signals.csv','-ascii');
%Scan Frequency
fs = 800;

win_len = 200; %Moving window length [samples]
overlap_percent = 65; %Moving window overlap [%]
overlap = floor((overlap_percent*win_len)/100); %Overlap [samples]
idx = [];

for j = 1:9
    %Select representative set
    x = signal(:,j);
    N = length(x);

    %PSD of original signal
    [half_f SumPyy] = PSD(x, fs);

    e = randn(1, N); %rand noise

    MSE = [];
k = [];

    for p = 5:30
        k = [k p];
        a = zeros(1,p+1);
P = 0;
        start = 1;

        %Calculate average(!) AR model coefficients
        % for certain order p
        for i = 1:(N/overlap)-1
            xn(1:N) = 0.0;
            if ((start+win_len) < N)
xn(start:start+win_len)=
x(start:start+win_len);
            end
        a = a + aryule(xn,p);
P = P + 1; %Number of windows
        start = start + overlap;
    end
        a = a/P;

end
% Apply AR coefficients to rand noise
b = 1;
x_AR = filter(b, a, e);

% Obtain PSD of AR model
[half_f SumPyy_AR] = PSD(x_AR, fs);
APPENDIX B

MATLAB CODE FOR INVESTIGATING THE NEURAL NETWORK PARAMETERS
clear; clc;

fs = 800;
win_len = 200; %Moving window length [samples]
overlap_percent = 65; %Moving window overlap [%]
overlap = floor((overlap_percent*win_len)/100); %Overlap [samples]

% Obtain set
x = signal(:,j);
N = length(x);

% Calculate average(!) AR model coefficients
% for certain order p
P = 0;
start = 1;
for i = 1:(N/overlap)-1
    xn(1:N) = 0.0;
    if ((start+win_len) < N)
        xn(start:start+win_len) = x(start:start+win_len);
    end
    a = a + aryule(xn,p);
P = P + 1; %Number of windows
    start = start + overlap;
end%i
a = a/P;

y = [y a']; %inputs
yt = [yt t(:, l-1)];%targets

end %j
end %l

% create net: one hidden layer with 10 nodes (output layer size is inferred: 3)
% net = newff(y, yt, 10, {'logsig' 'logsig'},
% 'trainscg');
% net.trainParam.perf = 'sse';
% net.trainParam.epochs = 50;
% net.trainParam.goal = 1e-5;
% view(net)
net = newff(y, yt, 15);
net.trainParam.lr = 0.25;
net.trainParam.goal = 1e-2;

% training
net = init(net); % initialize
[net, tr] = train(net, y, yt);

c = 0; %wrong answers
all = 0; %all results
y = [];
yt = [];

%Simulate
for l = 2:10
s1 = sprintf('..\Signals\10_EMG%d_TEST_Signals.csv', l);
signal = load(s1,'-ascii');

for j = 1:9

    %Obtain set
    x = signal(:,j);
    N = length(x);

    %Calculate average(!) AR model coefficients
    %for certain order p
    P = 0;
    start = 1;
    for i = 1:(N/overlap)-1
        xn(1:N) = 0.0;
        if ((start+win_len) < N)
            xn(start:start+win_len)= x(start:start+win_len);
        end
        a = a + aryule(xn,p);
        P = P + 1; %Number of windows
        start = start + overlap;
    end
    a = a/P;

    %Simulate
    y1 = sim(net,a');

    [ca ia] = max (y1);
    [ct it] = max (t(:,l-1));

    if (ia ~= it)
        c = c +1;
    end
    all = all + 1;

end %j
end %l

%ap = [ap p];
%cp = [cp (c/all)*100];

measuredTime = toc;
disp (measuredTime)
disp ((c/all)*100);
% plot (ap, cp);
% xlabel('AR model order');
% ylabel('Overall error from recognition [%]');
APPENDIX C

MATLAB CODE FOR INVESTIGATING THE AKAIKE CRITERION
clear; clc;

fs = 800;
win_len = 200; %Moving window length [samples]
overlap_percent = 65; %Moving window overlap [%]
overlap = floor((overlap_percent*win_len)/100); %Overlap [samples]

AIC = []; 
pi = [];

for p = 5:15 %AR model order
    a = zeros(1,p+1); %AR model coefficients
    for l = 10
        s1 = sprintf('..\Signals\10_EMG%d_TEST_Signals.csv', l);
        signal = load(s1,'-ascii');
        for j = 6

            %Obtain set
            x = signal(:,j);
            N = length(x);
            e = randn(1, N); %rand noise

            %Calculate average(!) AR model coefficients
            %for certain order p
            P = 0;
            start = 1;
            for i = 1:(N/overlap)-1
                xn(1:N) = 0.0;
                if ((start+win_len) < N)
                    xn(start:start+win_len) = x(start:start+win_len);
                end
                a = a + aryule(xn,p);
                P = P + 1; %Number of windows
                start = start + overlap;
            end
            a = a/P;

        end
    end

end

106
%Apply AR coefficients to rand noise- e
b = 1;
x_AR = filter(b, a, e);

x_hat = abs(x-x_AR');
PP = sum(x_hat.*x_hat);

cr = log(PP) + 2*(p + 1)/N;
AIC = [AIC cr];
pi = [pi p];

end %j
end %l
end %p

plot (pi, AIC);
xlabel('AR model order');
ylabel('AIC');
APPENDIX D

MATLAB CODE FOR INVESTIGATING THE STABILITY OF THE MODEL
clear; clc;
figure; hold on;
fs = 800;
win_len = 200; %Moving window length [samples]
overlap_percent = 65; %Moving window overlap [%]
overlap = floor((overlap_percent*win_len)/100); %Overlap [samples]

p = 5;
a = zeros(1,p+1); %AR model coefficients

tic;
for l = 2:10

    s1 = sprintf('..\Signals\10_EMG%d_TEST_Signals.csv', l);
signal = load(s1,'-ascii');

    for j = 1:9

        %Obtain set
        x = signal(:,j);
        N = length(x);

        %Calculate average(!) AR model coefficients
        %for certain order p
        P = 0;
        start = 1;
        for i = 1:(N/overlap)-1
            xn(1:N) = 0.0;
            if ((start+win_len) < N)
                xn(start:start+win_len)=x(start:start+win_len);
            end
            a = a + aryule(xn,p);
            P = P + 1; %Number of windows
            start = start + overlap;
        end%i
        a = a/P;

        r = roots(a);
        plot(r, 'x');
circle = rsmak(’circle’)
APPENDIX E

C# CODE FOR CALCULATING THE MAHalanobis DISTANCE BETWEEN THE FIVE FEATURES POINT AND THE GROUP OF DEFINED POINTS
public void Mahal(double[] a, int eLen,
        double[,] X, int XCol, int XRow,
        ref double result)
{

    //Calculate means
    double[] meansX = new double[XCol];
    Mean(X, XCol, XRow, meansX);

    //Calculate covariance matrix
    double[,] SIGMA = new double[XCol, XCol];
    Covariance(X, XCol, XRow, SIGMA);

    //Invert covariance matrix
    Inverse(SIGMA, XCol);

    //Prepare (Y(I,:)-mu)
    double[] e = new double[XCol];
    for (int i = 0; i < XCol; i++)
    {
        e[i] = a[i];
        e[i] -= meansX[i];
    }

double [,] eT = new double[eLen, 1];
    Transpose(e,eLen, eT);

    //Calculate Mahalanobis
    //d(I) = D1*D2
    //D1 = (Y(I,:)-mu)*inv(SIGMA)
    double [] D1 = new double[XCol];
    double tmp = 0.0;
    double temp_result = 0.0;
    for (int i = 0; i < XCol; i++)
    {
        for (int j = 0; j < XCol; j++)
        {
            tmp += e[j] * SIGMA[j, i];
        }
        D1[i] = tmp;
        tmp = 0.0;
    }

    //Mahal = D1*(Y(I,:)-mu)'
    for (int i = 0; i < XCol; i++)
    {

temp_result += D1[i] * eT[i, 0];
}

result = temp_result;