T.C.
BAHÇEŞEHİR ÜNİVERSİTESİ

FINGERPRINT PROCESSING VIA WAVELET AND SUPPORT VECTOR MACHINES

Master Thesis

BURHAN AYVAZOĞLU

ISTANBUL, 2009

T.C.
BAHÇEŞEHİR ÜNİVERSİTESİ

Institute of Sciences
Computer Engineering Graduate Program

FINGERPRINT PROCESSING VIA WAVELET AND SUPPORT VECTOR MACHINES

Master Thesis

Burhan AYVAZOĞLU

SUPERVISOR: ASSOC. PROF.DR. ADEM KARAHOCA

ISTANBUL, 2009

T.C

BAHÇEŞEHİR ÜNİVERSİTESİ

Institute of Sciences

Computer Engineering Graduate Program

Name of the thesis: **FINGERPRINT PROCESSING VIA WAVELET AND SUPPORT VECTOR MACHINES**

Name/Last Name of the Student: Burhan AYVAZOĞLU

Date of Thesis Defense:

This thesis has been approved by the Institute of Sciences.

Director

Signature

I certify that this thesis meets all the requirements as a thesis for the degree of Master of Science.

Head of Department

This is to certify that we have read this thesis and that we find it fully adequate in scope, quality and content, as a thesis for the degree of Master of Science.

Examining Committee Members

Prof.Dr. Nizamettin Aydın

Doc.Dr. Yalçın Çekiç

Supervisor

Assoc.Prof.Dr. Adem Karahoca

ACKNOWLEDGEMENTS

I want to thank to my supervisors, family and friends who have helped me generously.

Firstly, I would like to thank my supervisors especially Assoc. Prof. Dr. Adem Karahoca, Prof. Dr. Osman Nuri Uçan and Dr. Niyazi Kılıç who helped me to introduce many concepts that is used in this thesis. Their help is greatly appreciated.

I would like to thank my supervisor Assoc. Prof. Dr. Adem Karahoca for his interest and patience.

Finally, I would like to express my gratitude to my family and friends. Although they do not understand what is written in this thesis, their love and support were essential for the completion of my thesis. It is to them I would like to dedicate this thesis.

ABSTRACT

FINGERPRINT PROCESSING VIA WAVELET
AND SUPPORT VECTOR MACHINES

Burhan AYVAZOĞLU

Institute of Sciences, Computer Engineering Graduate Program

Supervisor: Assoc. Prof. Dr. Adem Karahoca

May 2009, 59 pages

In this thesis, fingerprint recognition process is considered. The fingerprints' data set constructed by using numerical values by applying wavelet and training with SVM(support vector machines). While obtaining these values, Haar and Daubechies filters are executed by using Matlab environment. During the classification process which is done by using Weka environment, Weka data mining tool is used to compute SVM. The expected correctness of classification is 75% but in this thesis, 80, 6061% correctness of classification is obtained for SVM.

Keywords: Wavelet,Wavelet (Haar Filter), Wavelet (Daubechies Filter), SVM (Support Vector Machines), SMO (Support Machines Optimization), Weka(Waikato Environment for Knowledge Analysis)

ÖZET

PARMAK İZİ İŞLEMEDE WAVELET
VE
DESTEK VEKTÖR MAKİNALARININ KULLANILMASI

Burhan AYVAZOĞLU

Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Yüksek Lisans Programı

Tez Danışmanı: Doç. Dr. Adem Karahoca

Mayıs 2009, 59 Sayfa

Bu tezde parmak izi tanıma yöntemi ele alındı. Parmak izi verileri Wavelet ile üretilen ve SVM(destek vektör makinaları) ile eğitilen numeric değerlerden oluşmaktadır. Bu değerleri elde ederken Haar ve DB Filtreleri kullanıldı. Weka ortamı kullanılarak yapılan sınıflandırma işlemi sırasında, veri ambarı aracı kullanılarak SVM hesaplanır. SVM için istenilen başarı oranı %75 iken bu tezde %80,6061 başarı oranı elde edildi.

Anahtar Kelimeler: Wavelet,Wavelet (Haar Filter), Wavelet (Daubechies Filter), SVM (Support Vector Machines), SMO (Support Machines Optimization), Weka(Waikato Bilgi Analizi Ortamı)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

Daubechies	:	DB
Support Vector Machines	:	SVM
Support Machines Optimization	:	SMO
Radial Basis Function	:	RBF
Approach of Adopted from The Digital Communication Theory	:	ECOC
True Pozitive	:	TP
False Pozitive	:	FP
Receiver Operating Characteristic	:	ROC
Mean Square Error	:	MSE
Mean Absolute Error	:	MAE
Root Mean Squared Error	:	RMSE
Data Mining	:	DM
Area Under the Curve	:	AUC
Automatic Fingerprint Identification Systems	:	AFIS

1-INTRODUCTION

Fingerprints are the result of oils from the skin being left behind. Ridges on the skin, called friction ridges, create a pattern that is transferred as an individual print [1].

Prints pressed into a soft substance, resulting in a negative impression are called plastic prints. Prints contaminated with a substance (oil, blood, etc.) are sometimes very clear replicas. Latent prints are hidden and can be found using chemicals, powders, or alternate light sources, such as the lasers you might see used on CSI(crime science investigation). Prints will stay where they are until they are obliterated by some physical action-they don't evaporate.

Examination of finger- and palm-prints has three possible outcomes: The person who made them is identified, the person who made them is not identified, or the prints are not adequate to determine identification [1].

1.1 History of Fingerprinting

Human fingerprints have been discovered on a large number of archaeological artifacts and historical items . Although these findings provide evidence to show that ancient people were aware of the individuality of fingerprints, such awareness does not appear to have any scientific basis [27]. It was not until the late sixteenth century that the modern scientific fingerprint technique was first initiate [29]. In 1684, the English plant morphologist, Nehemiah Grew, published the first scientific paper reporting his systematic study on the ridge, furrow, and pore structure in fingerprints [8].

Since then, a large number of researchers have invested huge amounts of effort on fingerprint studies. In 1788, a detailed description of the anatomical formations of fingerprints was made by Mayer [28] in which a number of fingerprint ridge characteristics were identified and characterized. Starting in 1809, Thomas Bewick began to use his fingerprint as his trademark, which is believed to be one of the most important milestones in the scientific study of fingerprint recognition [8].

Purkinje, in 1823, proposed the first fingerprint classification scheme, which classified fingerprints into nine categories according to the ridge configurations. Henry Fauld, in 1880, first scientifically suggested the individuality of fingerprints based on an empirical observation. At the same time, Herschel asserted that he had practiced fingerprint recognition for about 20 years. These findings established the foundation of modern fingerprint recognition. In the late nineteenth century, Sir Francis Galton conducted an extensive study on fingerprints. He introduced the minutiae features for fingerprint matching in 1888 [9].

An important advance in fingerprint recognition was made in 1899 by Edward Henry, who (actually his two assistants from India) established the well-known “Henry system” of fingerprint classification. By the early twentieth century, the formations of fingerprints were well understood. The biological principles of fingerprints are summarized below:

1. Individual epidermal ridges and furrows have different characteristics for different fingerprints;
2. The configuration types are individually variable, but they vary within limits that allow for a systematic classification;
3. The configurations and minute details of individual ridges and furrows are permanent

and unchanging.

The first principle constitutes the foundation of fingerprint recognition and the second principle constitutes the foundation of fingerprint classification. In the early twentieth century, fingerprint recognition was formally accepted as a valid personal identification method and became a standard routine in forensics. Fingerprint identification agencies were set up worldwide and criminal fingerprint databases were established. Various fingerprint recognition techniques, including latent fingerprint acquisition, fingerprint classification, and fingerprint matching were developed [10].

With the rapid expansion of fingerprint recognition in forensics, operational fingerprint databases became so huge that manual fingerprint identification became infeasible [8].

Automatic fingerprint recognition technology has now rapidly grown beyond forensic applications into civilian applications. In fact, fingerprint-based biometric systems are so popular that they have almost become the synonym for biometric systems [11].

1.2 Why Do You Need To FingerPrint Systems ?

As the need for personal authentication increases, many people are turning to biometric authentication as an alternative to traditional security devices. Biometrics is utilized in individual authentication techniques which identify individuals, i.e., living bodies by checking physiological or behavioral characteristics, such as fingerprints, voice, dynamic signatures, etc. Biometric systems are said to be convenient because they need neither something to memorize such as passwords or something to carry about such as ID cards. In spite of that, a user of biometric systems would get into a dangerous situation when her/his biometric data is abused. That is to say, the user cannot frequently replace or change her/his biometric data to prevent the abuse because of limits of biometric data intrinsic to her/himself. Therefore, biometric systems must protect the electronic information for biometrics against abuse, and also prevent fake biometrics [33].

Although the mostly used biometric attribute is fingerprint, the most confidential one is iris. However, using the fingerprint algorithm in fingerprints that aren't clean is a difficult task and the appliances used in identification iris are really expensive. Thus they are one of the major disadvantages while using these two attributes. Among these biometric characteristics, fingerprint needs physical contact and also can be captured or imitated [11].

Some of fingerprint systems may positively utilize artificial fingers as substitutes in order to solve the problem that a legitimate user cannot access. They are especially used in medical field while studying on genetical features and in criminal case used as evidence [2].

2-LITERATURE REVIEW

If manual comparison by a fingerprint expert is always done to say if two fingerprint images are coming from the same finger in critical cases, automated methods are widely used now.

Direct (optical) correlation is practically not used, as not very efficient for large database.

The general shape of the fingerprint is generally used to pre-process the images, and reduce the search in large databases. This uses the general directions of the lines of the fingerprint, and the presence of the core and the delta. Several categories have been defined in the Henry system: whorl, right loop, left loop, arch, and tented arch [3].

Similar to Henry system; algorithm named “feed back-based line” classifies a fingerprint image into one of the five classes: arch, left loop, right loop, whorl, and tented arch. We use a new low-dimensional feature vector obtained from the output of a novel oriented line detector.[32]

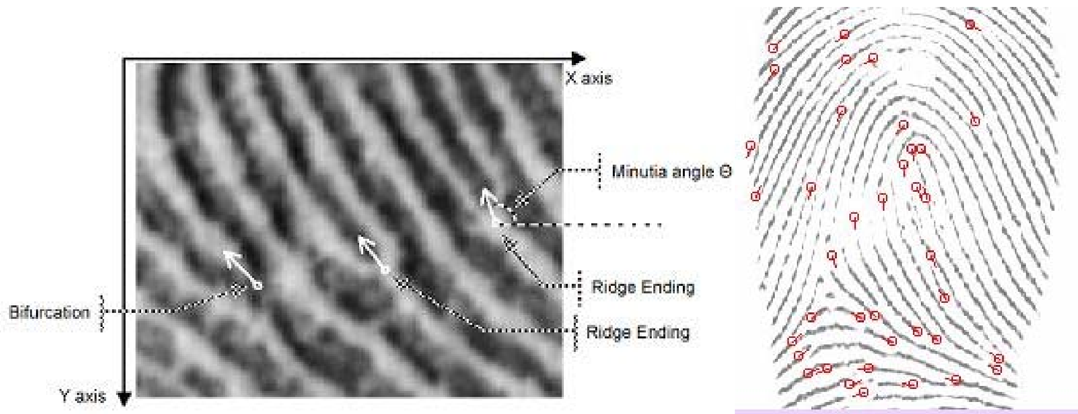

Figure 2.1 :Ridge Ending and Minutiae [3]

Most algorithms are using minutiae, the specific points like ridges ending, bifurcation.Only the position and direction of these features are stored in the signature for further comparison [3].

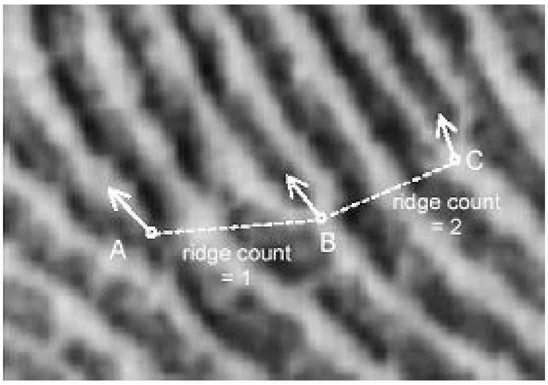

Figure 2.2 : Ridge Ending and Minutiae [3]

Some algorithms counts the number of ridges between particular points, generally the minutiae, instead of the distances computed from the position.

Pattern matching algorithms are using the general shape of the ridges. The fingerprint is divided in small sectors, and the ridge direction, phase and pitch are extracted and stored.

Very often, algorithms are using a combination of all theses techniques.

What is the probability of two fingerprints to match?

"On the individuality of fingerprints" (Pankanti/IBM) estimated to 6.10-8 the probability for 12 minutiae matching among 36 to match, but lots of estimations exist [3].

Table 2.1 :In the following array, M & R defines "regions", N minutiae [3].

Author	**P (Fingerprint Confguration)**	**N=36,R=24,M=72**	**N=12,R=8,M=72**
Galton (1982)	$(1/16)*(1/256)*(1/2)^{R}$	$1,45 * 10^{-11}$	$9,54 * 10^{-7}$
Pearson (1930)	$(1/16)*(1/256)*(1/36)^{R}$	$1,09 * 10^{-41}$	$8,65 * 10^{-17}$
Henry (1900)	$(1/4)^{N+2}$	$1,32 * 10^{-23}$	$3,72 * 10^{-9}$
Balthazard (1911)	$(1/4)^{N}$	$2,12 * 10^{-22}$	$5,96 * 10^{-8}$
Bose (1917)	$(1/4)^{N}$	$2,12 * 10^{-22}$	$5,96 * 10^{-8}$
Wentworth & Wilder (1918)	$(1/50)^{N}$	$6,87 * 10^{-62}$	$4,10 * 10^{-21}$
Cummins & Midlo (1943)	$(1/31)* (1/50)^{N}$	$2,22 * 10^{-63}$	$1,32 * 10^{-22}$
Gupta (1968)	$(1/10)*(1/10)*(1/10)^{R}$	$1,00 * 10^{-38}$	$1,00 * 10^{-14}$
Roxburgh (1933)	$(1/1000)*((1,5)/(10*2,412))^{N}$	$3,75 * 10^{-47}$	$3,35 * 10^{-18}$
Trauring (1963)	$(0.1944)^{N}$	$2,47 * 10^{-26}$	$2,91 * 10^{-9}$

Osterburg et al. (1980)	$(0.766)^{M-N}(0,234)^N$	$1,33 * 10^{-27}$	$3,05 * 10^{-15}$
Stoney (1985)	$(N/5)*0,6*(0,5*10^{-3})^{N-1}$	$1,2 * 10^{-80}$	$3,5 * 10^{-26}$

In the table above the algorithms used in fingerprints and the owners of these algorithms can be seen. Here the algorithms have been improved in the course of time and the image was minimized by being simplified. And the images whose color profundity is 8 bites are used.

There are a lot of methods and techniques in fingerprints algorithms. Many techniques have been developed so far. In addition to ones above Fourier and Wavelet are the best examples at the present day. We have preferred Wavelet in this study.

In addition to ones above Fourier and Wavelet are the best examples at the present days. [30].

The Short-Time Fourier Transform (STFT) maps a signal into a 2-D function of time and frequency. However, the time and frequency information can only be obtained with limited precision. The precision is determined by the size of the window used to analyze the signal [21].

Fourier analysis has a serious drawback. When a signal is transformed into the frequency domain, time information is lost. If you are mainly concerned with stationary signals, signals that don't change much over time, this drawback is not very important [20].

Wavelet analysis is a windowing technique, similar to the STFT, with variable-sized windows. It allows the use of long time intervals, when more low frequency information is sought, and shorter regions, when more high frequency information is what you are after [22].

Wavelet analysis is capable of revealing aspects of data that other signal analysis techniques miss, including aspects such as trends, breakdown points, discontinuities, and self- similarity. It is also often used to compress or denoise a signal without any appreciable degradation [22].

As we explained overhead there are many algorithms about fingerprint recognition. However biometric systems don’t end up with only fingerprint recognition. There are too many biometric systems nowadays. In the following table, we can see the comparison of these biometric systems and we can consider differences between fingerprint recognition systems and other biometric recognition systems.

Table 2.2 :Comparison of biometric technologies [8]

Biometric Identifer	**Universality**	**Distinctiveness**	**Permanence**	**Collectability**	**Performance**	**Acceptability**	**Circumvention**
DNA	H	H	H	L	H	L	L
Ear	M	M	H	M	M	H	M
Face	H	L	M	H	L	H	H
Facial thermogram	H	H	L	H	M	H	L
Fingerprint	M	H	H	M	H	M	M
Gait	M	L	L	H	L	H	M
Hand geometry	M	M	M	H	M	M	M
Hand vain	M	M	M	M	M	M	L
Iris	H	H	H	M	H	L	L
Keystroke	L	L	L	M	L	M	M
Odor	H	H	H	L	L	M	L
Retina	H	H	M	L	H	L	L
Signature	L	L	L	H	L	H	H
Voice	M	L	L	L	L	H	H

The data are based on the perception of the authors. High, Medium, and Low are denoted by H, M, and L, respectively [8].

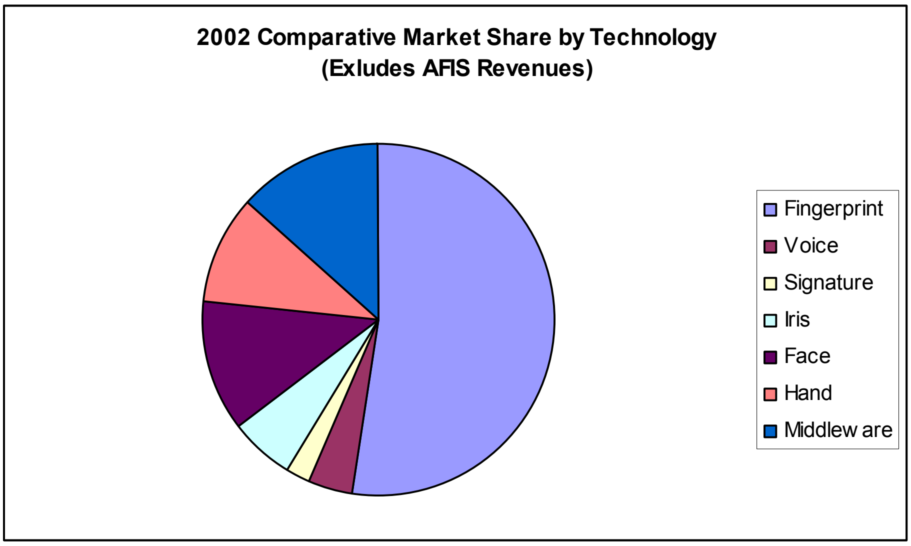

Figure 2.3 :Biometric Market Report (International Biometric Group)

Biometric Market Report (International Biometric Group) estimated the revenue of various biometrics in the year 2002 and showed that fingerprint-based biometric systems continue to be the leading biometric technology in terms of market share, commanding more than 50% of non-AFIS biometric revenue. Face recognition was second with 12,4%. Note that AFIS are used forensic applications [8].

3-MATERIAL AND METHODS

In this study, scanned finger print images are collected for fingerprint recognition. Totally 330 finger prints collected with 221x207 dimensions from 33 people. There are 10 finger prints for one person. We have gained 330 data input. Afterwards we transformed these data through wavelet filters (Haar & Db) to digital data in Matlab. In other words we obtained two different tables which full of digital data. We passed the digital data to excel and we assigned a class for each one of them, and we made them workable for Weka platform. We used SVM (Support Vector Machine or Super Machines Optimization (SMO)) model for best output. We used Radial Basis Function and Poly Kernel methods in SVM.(Poly- Percentage Poly-Cross validation , RBF – Percentage, RBF Cross validation) We used Matlab and Weka independent each other.We are going to analyze how the Wavelet which we use to turn the images into numeric values in Matlab and its filters do these processes and to do so what kind of mathematical formulas and methods they use.

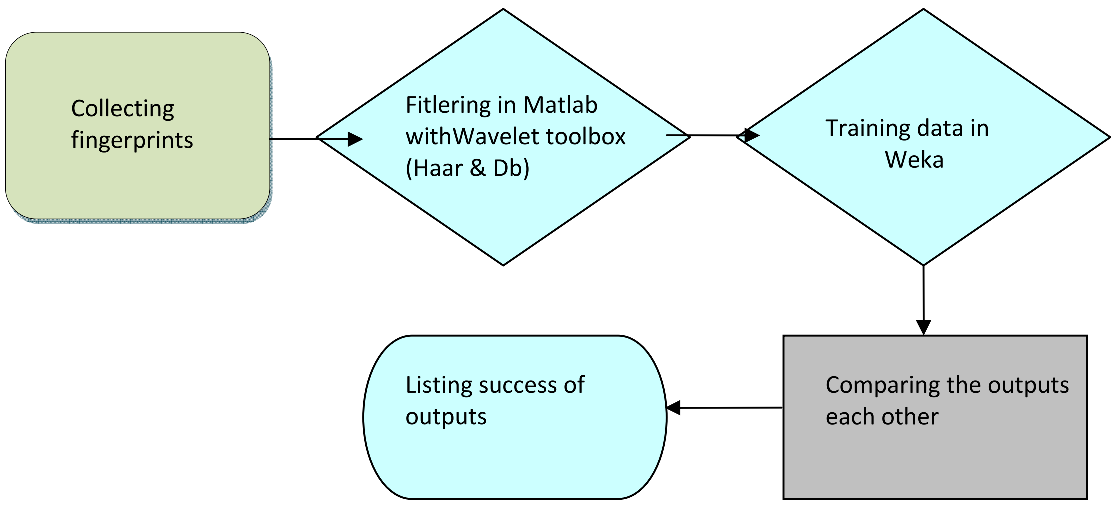

Figure-3.1:General schema of fingerprint recognition

3.1 WAVELET

Wavelet is analyzing technique which used often in digital signal processing area. With wavelet transform you should be able to divide the signal as much as you can. First, take the signal through with low pass and high pass filters; then down sample both of the two signals. Last, there would be half length of two signals which comes from the original signals that are high and low frequency bands combination. With hierarchy, we would be able to separate two signals with small band divisions. In jpeg2000 and mp3 formats compression would be used. Even this is a simple way to do, we still be able to get effective results. First, we will take a look at discrete wavelet transform implementation

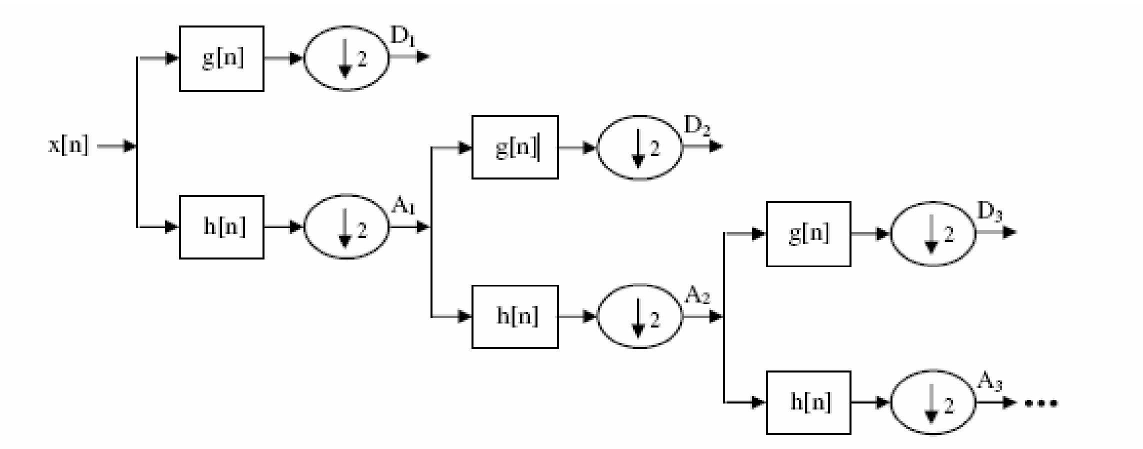

Figure 3.2 :Decomposition of Discrete Wavelet Transform Implementation [23]

The procedure of multiresolution decomposition of a signal x[n] is schematically shown in Figure-3.2 Each stage of this scheme consists of two digital filters and two downsamplers by 2. The first filter, g[n] is the discrete mother wavelet, high-pass in nature, and the second, h[n] is its mirror version, low-pass in nature. The downsampled outputs of first high-pass and low-pass filters provide the detail, D1 and the approximation, A1, respectively. The first Approximation, A1 is further decomposed and this process is continued as shown in Figure 3.2 [23].

We will see these processes in more detail with Haar and Daubechies. We will explain the difference between these two filters and see the images in Matlab environment. Once we look at the pictures, we may not see a major difference, but in training process we will choice which one to use by looking at the success percentage [23].

3.1.1 The Haar Wavelet Transform

Goal

The goals with this lab are [12]:

1. Understand the properties of the Haar basis.

2. Understand the averaging and differencing technique and how it relates to a wavelet decomposition.

3. Use the Haar transform to compress and de-noise one dimensional signals and images.

Introduction

This lab will illustrate some fundamental concepts about wavelets and multi-resolution theory. Most examples and definitions will be made with respect to the Haar scaling function and the Haar wavelet. However, the concepts covered in this lab are true for more ``advanced'' wavelets such as the Daubechies wavelets [12].

The Haar basis

Definition 1 (The Haar scaling function) Let $\Phi: R \rightarrow R$ be defined by [13]

$$\Phi(t) = \begin{cases} 1 & t \in [0,1) \\ 0 & t \notin [0,1) \end{cases} \tag{1}$$

Define $\Phi_i^{\,j} : R \rightarrow R$ as

$$\Phi_i^{\,j}(t) = \sqrt{2^j}\,\Phi(2^j t - i),\ j=0,1,\dots \text{ and } i=0,1,\dots,2^j-1 \tag{2}$$

Define the vector space V^j as

$$V^j = \mathbf{sp}\{ \Phi_i^{\ j} \}_{i=0,\ldots,2j-1} \qquad (3)$$

where **sp** denotes the linear span.

The index *j* refers to *dilation* and *i* refers to *translation*.

The constant $\sqrt{2^j}$ is chosen such that

$$<\Phi_i^{\ j}, \Phi_i^{\ j}> = \int_1^0 \Phi_i^{\ j}(t)^2 \, dt = 1$$

If one considers the scaling function on other intervals than [0,1], the normalization constant will change [13].

The Haar scaling function

a) Plot the basis vectors for V^j for *j*=0,1 and 2. Describe the significance of the index *j* in V^j

and the significance of the index i in $\Phi_i^{\ j}$ when j is fixed.
b) Claim: The following property holds for V^j: [23]

$$V^j \ V^{j+1} \qquad (4)$$

Definition 2 (The Haar wavelet function) Let Ψ: R→ R be defined by [5]

$$\Psi(t) = \begin{cases} 1 & t \in [0,1/2) \\ -1 & t \in [1/2,1) \\ 0 & t \notin [0,1) \end{cases} \qquad (5)$$

Define $\Psi_i^{\ j}$: R→ R as

$\Psi_i^{\ j}$ (t)=√2^j Ψ (2^jt-i), j=0,1,… and i=0,1,...,2^j-1 (6)

Define the vector space W^j

W^j=sp{ $\Psi_i^{\ j}$ }$_{i=0,\dots 2j-1}$ (7)

where **sp** denotes the linear span.

Again, The constant √2j is chosen such that

$$< \Psi_i^{\ j}, \Psi_i^{\ j} >= \int_1^0 \Psi_i^{\ j} (t)^2\, dt=1$$

If one considers the wavelet function on other intervals than [0,1], the normalization constant will change [23].

The Haar wavelet

a) Plot the basis vectors for W^j for *j*=0,1 and 2. Describe the significance of the index *j* in W^j [5]

and the significance of the index i in $\Psi_i^{\ j}$ when j is fixed.

b) Claim: The following property holds for W^j:

$W^j \subseteq V^{j+1}$ (8)

c) A standard argument for using wavelets rather than the Fourier basis for signal processing is that the Haar wavelet is *localized*. By looking at your plots from **a)** above, can you see why the Haar wavelet is called a ``localized" function? Is the Fourier basis localized? [23].

The structure of the Haar basis

The Haar basis (and other wavelet basis) has the very important property that

$$V^{j+1} = V^{j} \oplus W^{j} \qquad (9)$$

This property is explored in the following exercise.

The relation between V^j and W^j

Produce a few plots that illustrate the following relation:

$$V^{j+1} = V^{j} \oplus W^{j} \qquad (10)$$

Expansion of a signal in an ON-basis

Let $f \in R^N$ denote a finite signal with N samples (cf. an N-dimensional vector) given by

$$f=(f[0], f[1],\ldots . f[N-1]) \qquad (11)$$

Let $\{ \nu_k\}_{k=0,1,\ldots,N-1}$ be an ON- basis in R^N. Then

$$f=< f ,\nu_0 > \nu_0 + < f, \nu_1 > \nu_1 + \ldots\ldots + < f ,\nu_{N-1} > \nu_{N-1} \qquad (12)$$

First review how this was used for Fourier analyzing a signal and then see how this can be used for wavelet expansion of a signal [5].

Review of Fourier analyzing a signal

Let

$$\nu_{k=}\frac{1}{\sqrt{N}} (1,e^{2\pi i \frac{k}{N}}, e^{2\pi i \frac{2k}{N}} ,\ldots, e^{2\pi i \frac{nk}{N}},\ldots, e^{2\pi i \frac{(N-1)k}{N}}) \qquad (13)$$

Denote the (n+1):th element of the v_k vector as $v_k[n]$. Then f can be expanded as in (12) with

$$< f\ \mathrm{v_k} >= \sum_{n=0}^{N-1} = f[n] v_k[n]^* = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} = f[n] e^{-2\pi i \frac{nk}{n}} \qquad (14)$$

where * denotes complex conjugation. Numerically, we compute $<f,v_k>$ by using the Fast Fourier Transform [5].

Wavelet transformation of a signal

In this example we consider a signal with 8 samples for clarity. However, the procedure can be generalized to any finite signal. The analysis simplifies if the length (dimension) of the signal is 2k for some positive integer k [24].

Let $f \in \mathrm{R}^8$ and expand this signal $\mathrm{V}^0 = \mathrm{W}^0 \oplus \mathrm{W}^1 \oplus \mathrm{W}^2$ into spanned by the basis vectors according to Definitions 1 and 2 (discretized). Then

$$\begin{aligned} f = & < f, \Phi_0^{\ 0} > \Phi_0^{\ 0} + < f, \Psi_0^{\ 0} > \Psi_0^{\ 0} + \\ & < f, \Psi_0^{\ 1} > \Psi_0^{\ 1} + < f, \Psi_1^{\ 1} > \Psi_1^{\ 1} + \\ & < f, \Psi_0^{\ 2} > \Psi_0^{\ 2} + < f, \Psi_1^{\ 2} > \Psi_1^{\ 2} + < f, \Psi_2^{\ 2} > \Psi_2^{\ 2} + < f, \Psi_3^{\ 2} > \Psi_3^{\ 2} \end{aligned} \qquad (15)$$

The correspondence to Fourier expanding a signal is now to compute the inner products

$$< f, \Phi_{\mathrm{i}}^{\mathrm{j}} > \text{ and } < f, \Psi_{\mathrm{i}}^{\mathrm{j}} >$$

In principle, we could do this by evaluating an integral (or a sum in the discrete case). In practice, computation of the inner products for a wavelet basis is done as illustrated by the following example [24].

Example: Averaging and differencing of a finite signal

Consider f=(2,5,8,9,7,4,-1,1). We wish to expand this signal in the Haar basis according to (15). In practice, we do this by performing the following steps: [24]

Step 1:

f=(2+5,8+9,7+4,-1+1,2-5,8-9,7-4,-1-1)/ $\sqrt{2}$ =

= (7,17,11,0,-3,-1,3,-2) / $\sqrt{2}$

Step 2:

$$f=(\frac{7+17}{\sqrt{2}},\frac{11+0}{\sqrt{2}},\frac{7-17}{\sqrt{2}},\frac{11-0}{\sqrt{2}},-3,-1,3,-2)/\sqrt{2} =$$

$$f=(\frac{24}{\sqrt{2}},\frac{11}{\sqrt{2}},\frac{-10}{\sqrt{2}},\frac{11}{\sqrt{2}},-3,-1,3,-2)/\sqrt{2}$$

Step 3:

$$f=(\frac{24+11}{(\sqrt{2})^2},\frac{24-11}{(\sqrt{2})^2},\frac{-10}{\sqrt{2}},\frac{11}{\sqrt{2}},-3,-1,3,-2)/\sqrt{2} =$$

$$f=(\frac{35}{2},\frac{13}{2},\frac{-10}{\sqrt{2}},\frac{11}{\sqrt{2}},-3,-1,3,-2)/\sqrt{2} \cong$$

$\cong$ (12.4,4.60,-5.00,5.50,-2.12,-0.707,2.12,-1.41)

The numbers in the final vector are the coefficients in the expansion (15). You probably notice the pattern, we add/subtract a pair of numbers, divide by the normalization factor $\sqrt{2}$
and between every step we don't touch the last elements of the vector. This procedure is called ``averaging and differencing'' [24].

The mathematics of averaging and differencing

The example in the previous section shows the steps that we want to implement when we code up a one dimensional Haar wavelet transform. When we consider two dimensional signals (images) we will see that all we need to do is to act with the one dimensional transform on each row of the image followed by a transformation of each column of the image [5].

We still consider the signal f=(2,5,8,9,7,4,-1,1). The coefficients of the original signal are the coefficients of the vector f expanded in V^3, i.e., [5]

$< f , \Phi_0^3 >=2$, $< f , \Phi_1^3 >=5$, $< f , \Phi_2^3 >=8$ .etc

Recall that the basis vectors of V^3 are translation of the Haar mother scaling function which has been dilated so that each basis function has a support equal to 1/8 (=1 pixel) of the support of f which is 8 pixels [5].

Our goal is to expand f into $V^0 = W^0 \oplus W^1 \oplus W^2$ We will do this in three steps.

Step 1: Expand f into $V^2 \oplus W^2$

The first step in the example above can be described as the matrix-vector multiplication $f_1=W_1 f$ where [5]

$$W_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \qquad (16)$$

Realize that the first four rows correspond to the basis vectors Φ_0^2 , Φ_1^2 , Φ_2^2 and Φ_3^2

spectively which span V^2 [5].

The last four rows correspond to the basis vectors Ψ_0^2, Ψ_1^2, Ψ_2^2 and Ψ_3^2 which span W^2 [5].
Hence, the final vector in Step 1 in the example above is the coefficients of f in the expansion [5].

$$f=\frac{1}{\sqrt{2}}\left(7\,\Phi_0^2 + 17\Phi_1^2 + 11\Phi_2^2 + 0\Phi_3^2 - 3\Psi_0^2 - 1\Psi_1^2 + 3\Psi_2^2 - 2\Psi_3^2\right) \tag{17}$$

i.e., we have expanded our signal into $V^2 \oplus W^2$

Step 2: Expand f into. $V^1 \oplus W^1 \oplus W^2$

In this step we already have the coefficients for the basis vectors of W^2. Therefore we don't have to worry about these coefficients in this step, we simply just keep these coefficients (the last four entries of our vector in the previous step) and only work with the first four entries [5].
The second step can be described as the matrix-vector multiplication $f_2=W_2 f_1$ where

$$W_2=\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{18}$$

We can combine the first and the second step as $f_2=W_2W_1f$ where

$$W_1\ W_2 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \qquad (19)$$

Notice that the first two rows correspond to the basis vectors Φ_0^1 and Φ_1^1 which span V^1.

The third and fourth rows correspond to the basis vectors Ψ_0^1 and Ψ_0^1 which span W^1. The last four rows correspond to the basis vectors of W^2.

Hence, the final vector in Step 2 in the example above is nothing else than the coefficients of [5]

f in the expansion

$$f=\frac{1}{\sqrt{2}}\ (\frac{24}{\sqrt{2}}\ \Phi_0^1 + \frac{11}{\sqrt{2}}\ \Phi_1^1 - \frac{-10}{\sqrt{2}}\ \Psi_0^1 + \frac{11}{\sqrt{2}}\ \Psi_1^1 -3\ \Psi_0^2 + -1\ \Psi_1^2 +3\ \Psi_2^2 -2\ \Psi_3^2) \qquad (20)$$

Step 3: Expand f into $V^0 \oplus W^0 \oplus W^1 \oplus W^2$

In this step we already have the coefficients for the basis vectors of $W^1 \oplus W^2$. Hence we don't have to worry about these coefficients in this step, we simply just keep these coefficients (the last six entries of our vector in the previous step) and only work with the first two entries [5].

The third step can be described as the matrix-vector multiplication $f_3 = W_3 f_2$ where

$$W_3 = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{21}$$

We can combine the first, second and third step as $f3 = W_3 W_2 W_1 f$ where

$$W_1\, W_2\, W_3 = \begin{pmatrix} \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} \\ \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{-1}{2\sqrt{2}} & \frac{-1}{2\sqrt{2}} & \frac{-1}{2\sqrt{2}} & \frac{-1}{2\sqrt{2}} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \tag{22}$$

Notice that the first row corresponds to the basis vector Φ_0^0 which spans V^0, the second row corresponds to the basis vector Ψ_0^0 which spans W^0 and so on [5].

The final vector in Step 3 in the example above is the coefficients of f in the expansion

$$f=(\frac{35}{2}\Phi_0^0+\frac{13}{2}\Psi_0^0+\frac{-10}{\sqrt{2}}\Psi_0^1+\frac{11}{\sqrt{2}}\Psi_1^1-3\Psi_0^2-1\Psi_1^2+3\Psi_2^2-2\Psi_3^2)/\sqrt{2}\cong$$

(23)

$$\cong(12.4\,\Phi_0^0+4.60\,\Psi_0^0-5.00\,\Psi_0^1+5.50\,\Psi_1^1-2.12\,\Psi_0^2-0.707\,\Psi_1^2+2.12\,\Psi_2^2-1.41\Psi_3^2)$$

The wavelet transform as a unitary operator

The procedure in the example above can be expressed as

$$Wf=(\frac{35}{2}\Phi_0^0+\frac{13}{2}\Psi_0^0+\frac{-10}{\sqrt{2}}\Psi_0^1+\frac{11}{\sqrt{2}}\Psi_1^1-3\Psi_0^2-1\Psi_1^2+3\Psi_2^2-2\Psi_3^2)/\sqrt{2}$$

(24)

where *W* is the wavelet transform.

Wavelets and filters

The method of averaging and differencing explained above can also be expressed as *filtering* the data. Averaging corresponds to a low pass filtering. It removes high frequencies of the data. Since sharp changes in the data correspond to high frequencies, the averaging procedure tends to smooth the data. The low pass filter can be expressed as $\frac{1}{\sqrt{2}}$ (1,-1) in the Haar case and when we average the data, we move this filter along our input data [14].

The differencing corresponds to high pass filtering. It removes low frequencies and responds to details of an image since details correspond to high frequencies. It also responds to noise in an image, since noise usually is located in the high frequencies. The high pass filter can be expressed as $\frac{1}{\sqrt{2}}$ (1,-1) in the Haar case and when we difference the data, we simply move this filter along our input data. The low pass and high pass filters make up what in signal processing language is referred to as a *filter bank*. The method of averaging and differencing is referred to as *analysis*. The reverse procedure (going the opposite way in the example above) is called *synthesis* [14].

Hence, the wavelet transform separates low and high frequencies, just as the Fourier transform. Since different features of a signal (background, details, noise, edges, etc.) correspond to different frequencies, this is a key to use wavelets in signal processing. The nice thing is that wavelets are localized since they only live on part of the interval of the data, as opposed to the trigonometric functions used in Fourier analysis which live on the entire interval of the data [14].

3.1.2 The Daubechies Wavelet Transform

Goal

The goal with this lab is to design a Daubechies wavelet transform and use it to compress and de-noise one dimensional signals and images [6].

Introduction

This lab can be thought of as an extension to the second lab The Haar Wavelet Transform. This lab does not introduce any new mathematical concepts compared to the multiresolution analysis you studied in the last lab for the Haar basis. The multiresolution structure is true for the Daubechies basis you will work with in this lab, and just as the Haar transform is a unitary transform, so is the Daubechies transform you will use for this lab [6].

The main difference in this lab will be that your wavelet transformation code now has to take into account that the filter sometimes will ``run outside the boundary'' of the signal. Keep in mind that the tools you develop in this lab might be helpful for your final project and any future work in signal processing [6].

The computational cost of the wavelet transform

Let us compute the computational cost for performing a wavelet transform for a filter of length L and a signal of length N=2k. (Here we do not consider boundary effects but when the length of our signal is long, boundary consideration is not expensive relative to the interior of the signal.) [6]

At the first level we need to move our low pass filter along the whole signal. Since we down sample by a factor of two, this means that the first low pass filtering costs $\frac{LN}{2}$ operations. At the second level we need to move our low pass filter along half of the signal which costs $\frac{LN}{4}$ operations. Continuing in this manner we find that the cost for low pass filtering is given by

$$\text{cost}_{\text{low-pass}} = L\left(\frac{N}{2} + \frac{N}{4} + \ldots + \frac{N}{N^{\log_2 N}}\right) = NL\left(\frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{N}\right)$$

The cost for high pass filtering is the same. Hence,

$$\text{cost} = \text{cost}_{\text{low-pass}} + \text{cost}_{\text{high-pass}}$$

$$= NL\left(1 + \frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^{k-1}}\right)$$

$$= NL\sum_{m=0}^{k-1}\frac{1}{2^m} = NL\left(\frac{\left(\frac{1}{2}\right)^k - 1}{\frac{1}{2} - 1}\right)$$

$$= NL \left(\frac{1-\left(\frac{1}{2}\right)^k}{1-\frac{1}{2}} \right) = 2NL \left(1-\left(\frac{1}{2}\right)^k \right) < 2LN$$

Now the punch line: *the wavelet transform is an* O(N) *operation!* Note that this is not only for performing one level wavelet decomposition, but for performing the full wavelet transformation. This is somewhat counter intuitive, we may expect a cost of O(Nlog2N), but the wavelet transform is even fast than that [6].

Example: A Daubechies wavelet applied to a finite signal

In this section we will use the (normalized) Daubechies filter

$$c=(c(0),c(1),c(2),c(3)) = \left(\frac{1+\sqrt{3}}{4\sqrt{2}}, \frac{3+\sqrt{3}}{4\sqrt{2}}, \frac{3-\sqrt{3}}{4\sqrt{2}}, \frac{1-\sqrt{3}}{4\sqrt{2}} \right)$$

as a low pass filter and

$$d=(d(0),d(1),d(2),d(3))= (c(3),-c(2),c(1),-c(0))$$

as a high pass filter to wavelet decompose a signal. The factor $\frac{1}{\sqrt{2}}$ is a normalization factor so that $\|c\|_2 = \|d\|_2 = 1$. Before using a filter that is given to you, make sure to find out whether it is normalized or not [6].

The two filters combined form a so-called *filter bank*. The idea is to use the following example as a test example for the code you will be asked to write and use later on in this lab. Since our filter now has length 4, we need to decide what to do when our filter encounters a boundary point. This example will periodically extend the signal [6].

Consider $f=(f(0), f(1),\ldots, f(7)) = (2,5,8,9,7,4,-1,1)$.

We wish to expand this signal in the basis defined by the filter bank above. In practice, we do this by performing the following steps:

Step 1:

Extend the signal periodically. We could also do an even extension of the signal or pad with some constant. What we choose to do at the boundary depends on the application and the signal. If we expect our signal to come from some ``periodic process'', a periodic extension makes sense [6].

We denote the extended signal with a tilde $\tilde{f}$

$$\tilde{f} = (f(6), f(7), f(0), f(1), f(2), f(3), f(4), f(5), f(6), f(7)) = (-1,1,2,5,8,9,7,4,-1,1)$$

Now move the low and high pass filters along this vector, two steps at a time (shifting by two steps comes from the fact that we are *down sampling* the signal):

$$f_1 = \begin{pmatrix} c(0)f(6) + c(1)f(7) + c(2)f(0) + c(3)f(1) \\ c(0)f(0) + c(1)f(1) + c(2)f(2) + c(3)f(3) \\ c(0)f(2) + c(1)f(3) + c(2)f(4) + c(3)f(5) \\ c(0)f(4) + c(1)f(5) + c(2)f(6) + c(3)f(7) \\ d(0)f(6) + d(1)f(7) + d(2)f(0) + d(3)f(1) \\ d(0)f(0) + d(1)f(1) + d(2)f(2) + d(3)f(3) \\ d(0)f(2) + d(1)f(3) + d(2)f(4) + d(3)f(5) \\ d(0)f(4) + d(1)f(5) + d(2)f(6) + d(3)f(7) \end{pmatrix}$$

$$= (0.155 , 5.78 , 12.4 , 6.37 , -0.837 , 0.966 , 0.871 , -3.12)$$

(For clarity, only three significant digits will be printed out in this example but the actual computations made used more digits.) [6]

Step 2:

For the next step, we keep the last half of the vector f1 fixed while we low- and high pass filter the first half of the vector. In order to do this, it is necessary to periodically extend the first half of the vector f1

$\tilde{f}_1 = (f_1(2), f_1(3), f_1(0), f_1(1), f_1(2), f_1(3), f_1(4), f_1(5), f_1(6), f_1(7))$

=(12.4,6.37,0.155,5.78,12.4,6.37,-0.837,0.966,0.871,-3.12).

Now move the low and high pass filters along the first six elements of this vector, two steps at a time:

$$f_2 = \begin{pmatrix} c(0)f_1(2) + c(1)f_1(3) + c(2)f_1(0) + c(3)f_1(1) \\ c(0)f_1(0) + c(1)f_1(1) + c(2)f_1(2) + c(3)f_1(3) \\ d(0)f_1(2) + d(1)f_1(3) + d(2)f_1(0) + d(3)f_1(1) \\ d(0)f_1(0) + d(1)f_1(1) + d(2)f_1(2) + d(3)f_1(3) \\ f_1(4) \\ f_1(5) \\ f_1(6) \\ f_1(7) \end{pmatrix}$$

=(10.6 , 6.87 , -5.70 , 6.02 , -0.837 , 0.966 , 0.871 , -3.12)

Note that we didn't touch the last four elements!

Step 3:

In the last step we only low- and high pass filter the first quarter of the vector f_2. First we extend the first two elements periodically: [6]

$\tilde{f}_2 = (f_2(0), f_2(1), f_2(0), f_2(1), f_2(2), f_2(3), f_2(4), f_2(5), f_2(6), f_2(7))$

=(10.6,6.87,10.6,6.87,-5.70,6.02,-0.837,0.966,0.871,-3.12).

As the final step we act with the filter on the first four elements of this vector:

$$f_3 = \begin{pmatrix} c(0)f_2(0) + c(1)f_2(1) + c(2)f_2(0) + c(3)f_2(1) \\ d(0)f_2(0) + d(1)f_2(1) + d(2)f_2(0) + d(3)f_2(1) \\ f_2(2) \\ f_2(3) \\ f_2(4) \\ f_2(5) \\ f_2(6) \\ f_2(7) \end{pmatrix}$$

= (12.4 , 2.66 , -5.70 , 6.02 , -0.837 , 0.966 , 0.871 , -3.12)

Note that we didn’t touch the last six elements!

3.1.3 Haar vs. Daubechies Transform

When I first started studying wavelets, one of the many questions I had was "How does one decide which wavelet algorithm to use?" There is no absolute answer to this question. The choice of the wavelet algorithm depends on the application. The Haar wavelet algorithm has the advantage of being simple to compute and easier to understand. The Daubechies algorithm has a slightly higher computational overhead and is conceptually more complex. As the matrix forms of the Daubechies algorithm above show, there is overlap between iterations in the Daubechies transform step. This overlap allows the Daubechies algorithm to pick up detail that is missed by the Haar wavelet algorithm [15].

The red line in the plot below shows a signal with large changes between even and odd elements. The pink line plots the largest band of Haar wavelet coefficients (e.g., the result of the Haar wavelet function). The green line plots the largest band of Daubechies wavelet coefficients. The coefficient bands contain information on the change in the signal at a particular resolution [15].

In this version of the Haar transform, the coefficients show the average change between odd and even elements of the signal. Since the large changes fall between even and odd elements in this sample, these changes are missed in this wavelet coefficient spectrum. These changes would be picked up by lower frequency (smaller) Haar wavelet coeffient bands. The overlapped coefficients of the Daubechies transform accurately pick up changes in all coefficient bands, including the band plotted here [16].

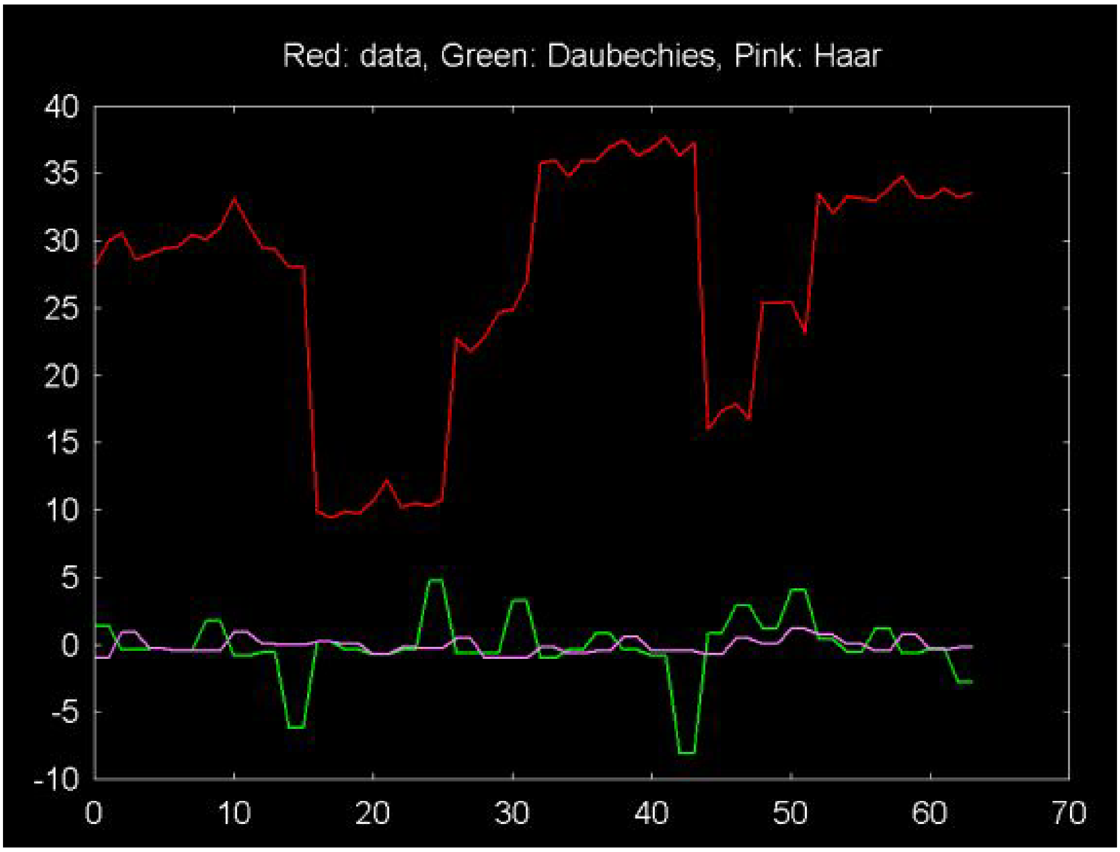

Figure 3.3 :The Discrete Wavelet Transform [26]

After investigation of two formantation, let's look at the differences in pictures of matlab environment.

In process below, decomposition was created in two levels by using Haar wavelet.

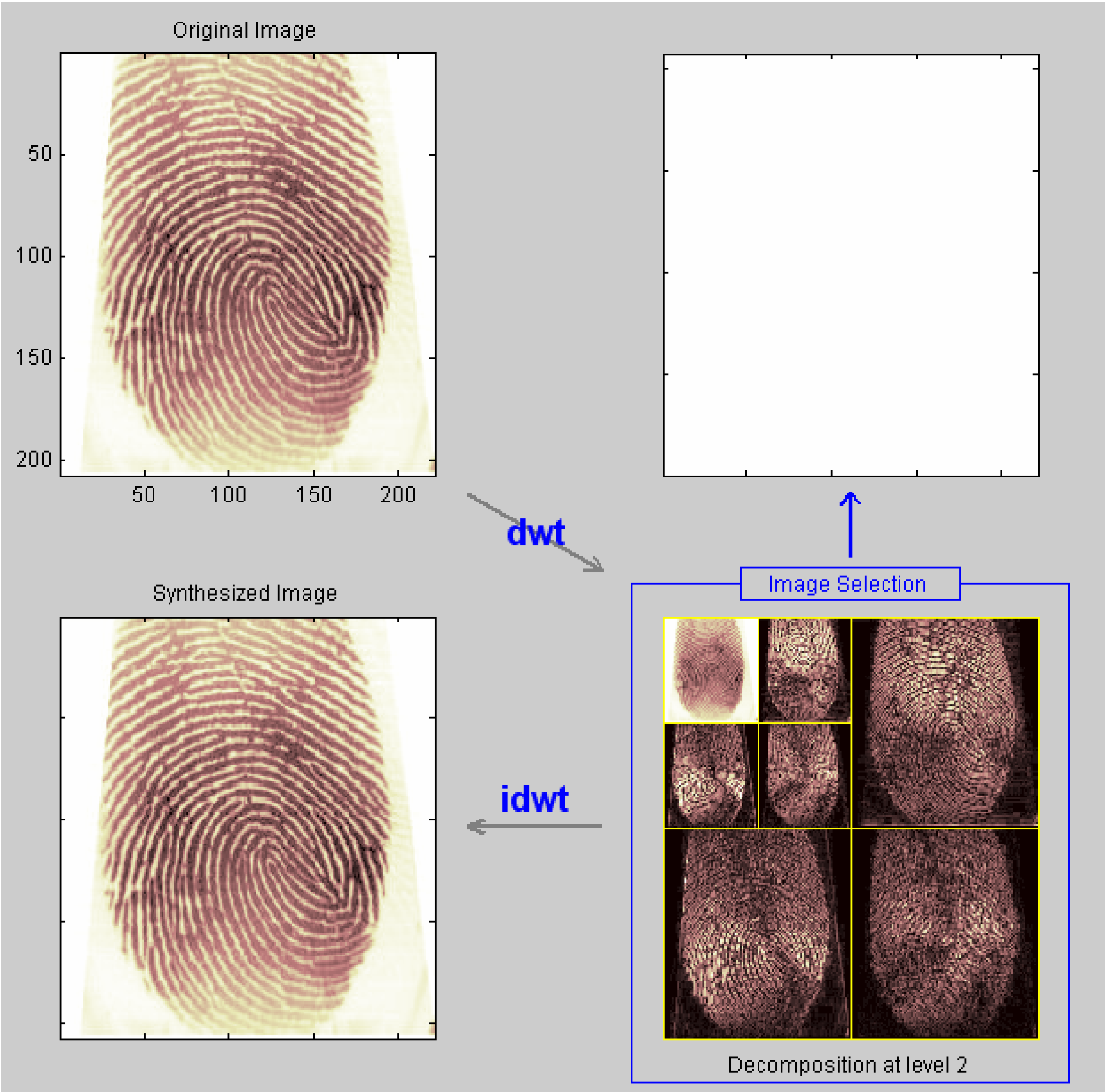

Figure 3.4 :Compressing of an original image

Decomposition was created in two levels by using Haar wavelet [16]

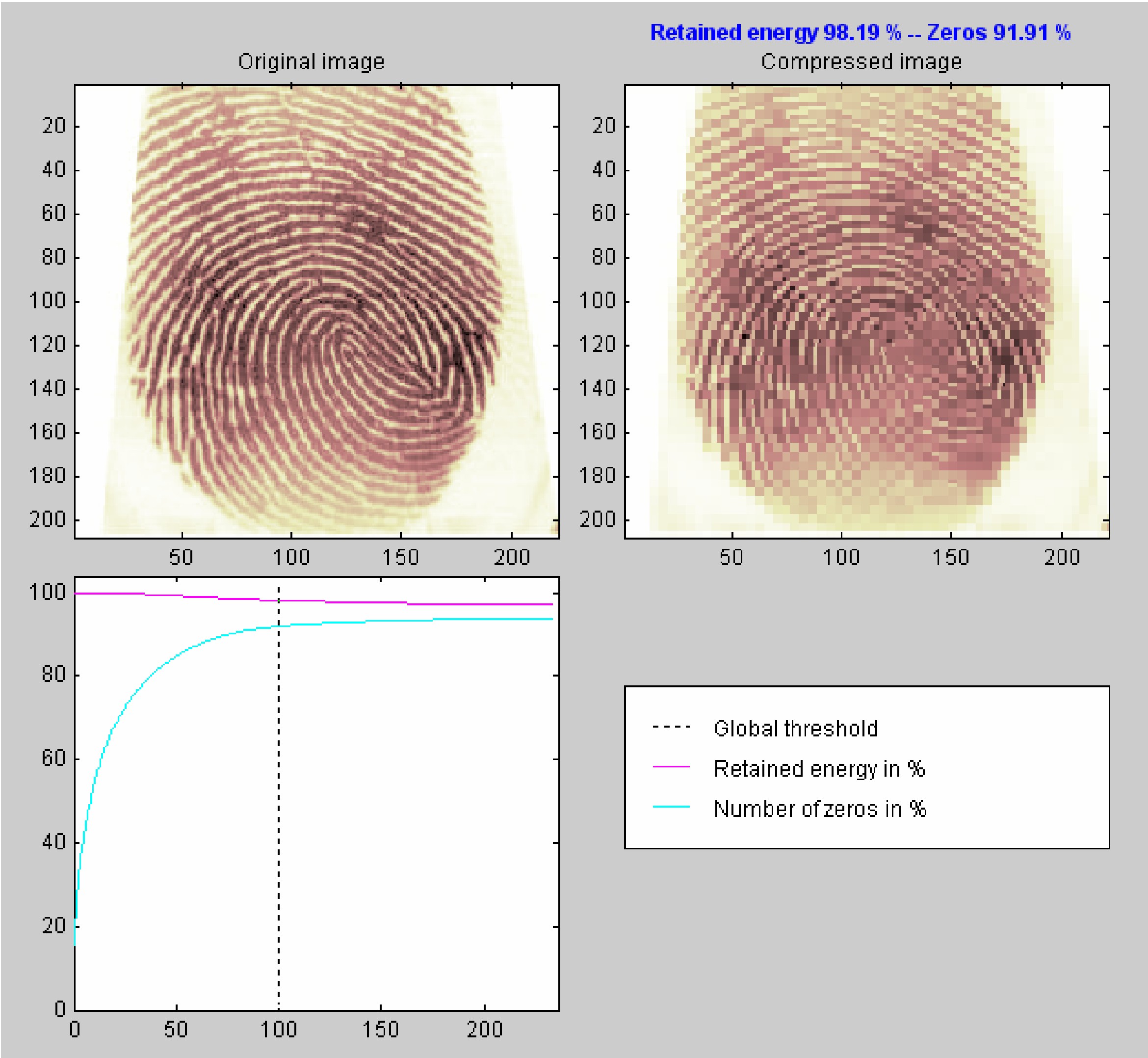

Figure 3.5 :Compressing of an original image

Decomposition was created in two levels by using Haar wavelet [16]

In process below, decomposition was created in two levels by using Daubechies wavelet.

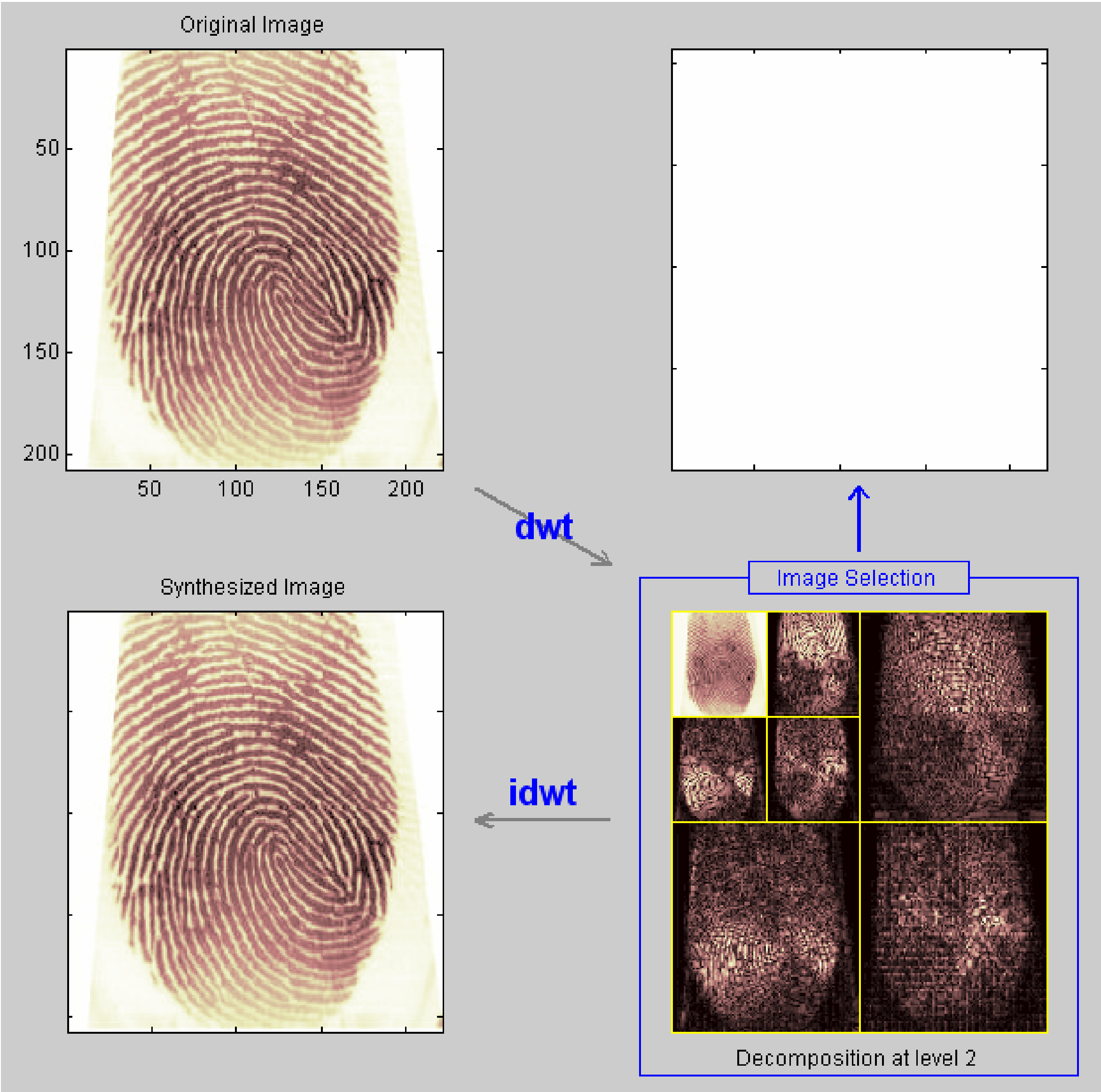

Figure 3.6 :Compressing of an original image

Decomposition was created in two levels by using Daubechıes wavelet [16]

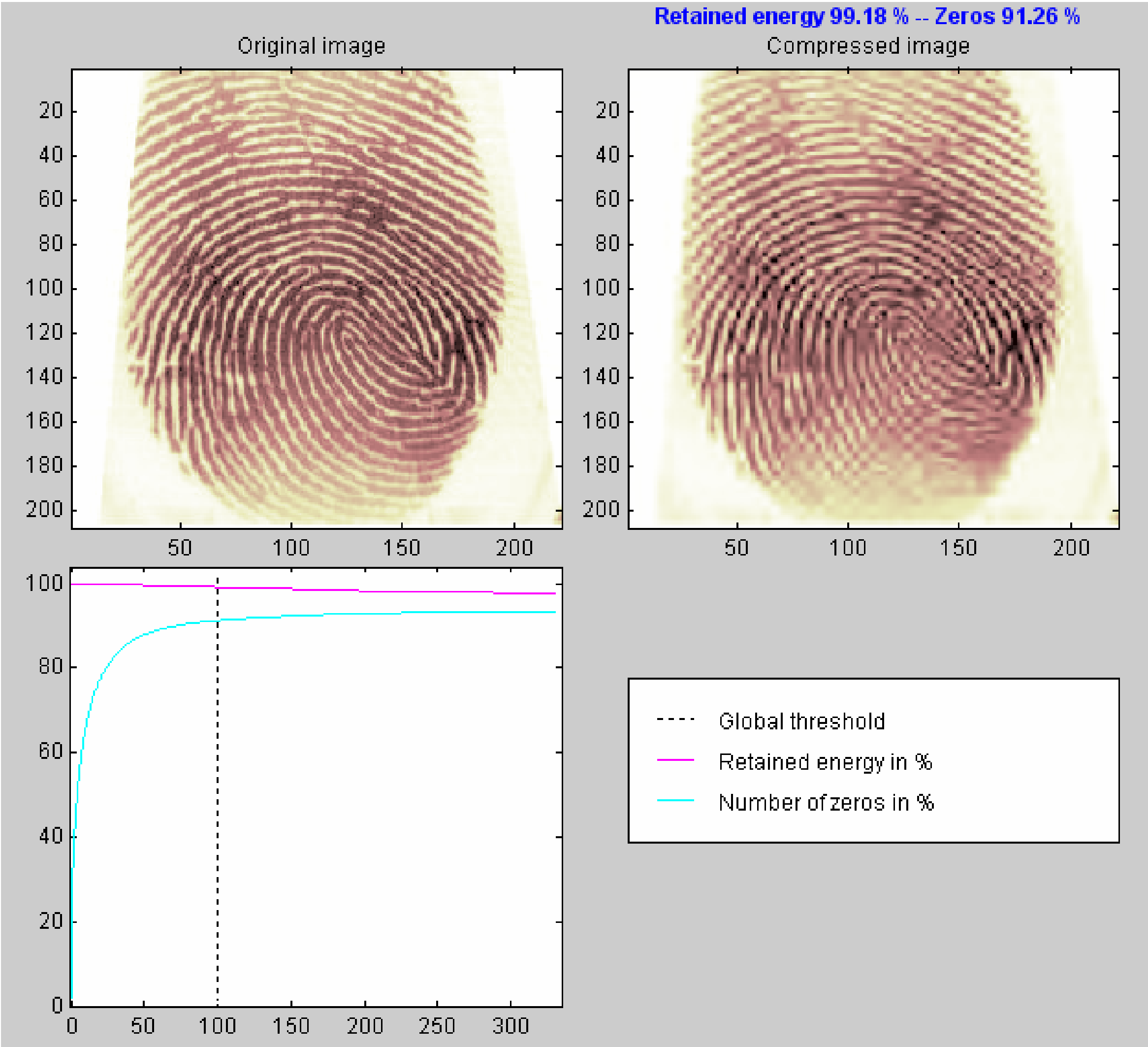

Figure 3.7 :Compressing of an original image.

Decomposition was created in two levels by using Daubechıes wavelet.

DWT is discrete wavelet transform, and IDWT is inverse discrete wavelet transform. As we see above images are being seperated by low and high pass filters. Then they're combining again. As a result of this filtering process (IDWT) there are damaged data in the image. Actually, the synthesize images in Figure 3.4 and Figure 3.6 are not the original of the image, there are some corruption. This is not a big loss. We can explain this with a simple example: when we send our images by e-mail, they pass from wavelet transfom, DWT is being used during sending process, IDWT is being used during receiving process.In fact, the images we send and received by opposed side are not similar but we couldn't understand this [17].

The compressed images in Figure 3.5 and Figure 3.7 are the filtered (DWT) states in second degree of the original image. And as it seems it is %99,18 corrupted. In this study we perform this technic to ease image training. So, we reduce data for a quick and clear result. As you appreciate that less level cause clear training results but hard to maintain [17].

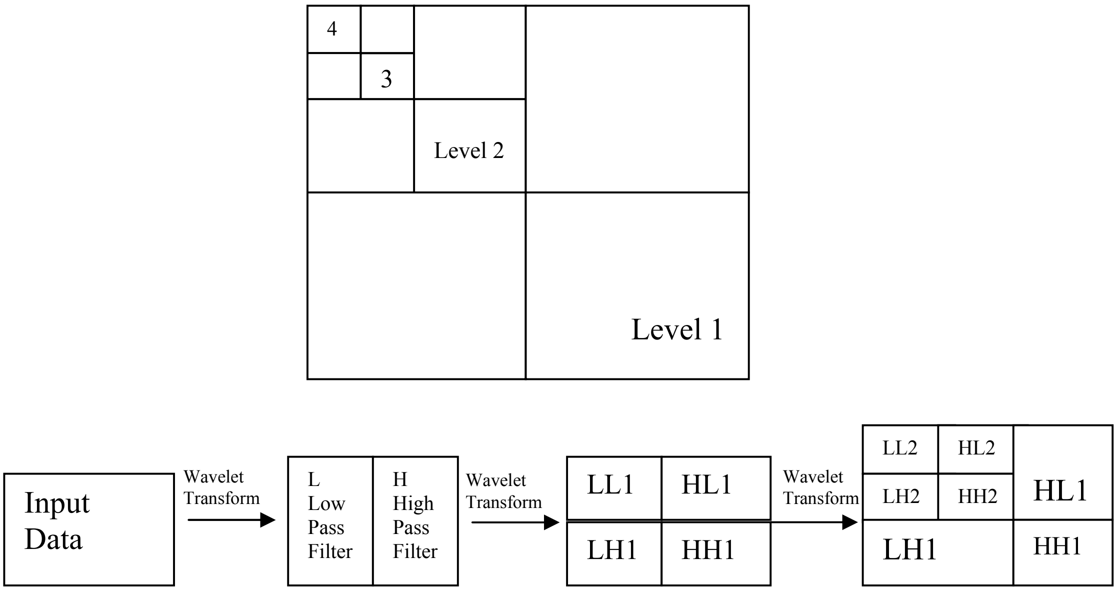

Figure 3.8 :Level of Wavelet

In the pictures above Figure 3.8 we have seen the processed samples of fingerprints in second level with Haar and Db filters. In the last picture (Figure 3.4 & Figure 3.6) we see the shrinking of the image while the levels progressing [17].

In above Figure 3.8, we see detail coefficient of wavelet transform at second level. This study is performed according to low transient filter. If decomposed image at second level is taken, we will get LL2 [17].

3.2 Support Vector Machine

The SVM proposed by Vapnik has been studied extensively for classification, regression and density estimation [7].

SVM maps the input patterns into a higher dimensional feature space through some nonlinear mapping chose a priori. A linear decision surface is then constructed in this high dimensional feature space. Thus, SVM is a linear classifier in the parameter space, but it becomes a nonlinear classifier as a result of the nonlinear mapping of the space of the input patterns into the high dimensional feature space. Training the SVM is a quadratic optimization problem. The construction of a hyper plane wT x + b = 0 (w is the vector of hyper plane coefficients, b is a bias term) so that the margin between the hyper plane and the nearest point is maximized and can be posed as the quadratic optimization problem. SVM has been shown to provide high generalization ability. A proper kernel function for a certain problem is dependent on the specific data and till now there is no good method on how to choose a kernel function . In this study, the choice of the kernel functions was studied empirically and optimal results were achieved using radial basis function (RBF) kernel function [7].

The SVM is a binary classifier which can be extended by fusing several of its kind into a multiclass classifier. In this study, SVM decisions were fused using the ECOC approach, adopted from the digital communication theory. In the ECOC approach, up to 2n−1 − 1 (where n is the number of classes) SVMs are trained, each of them aimed at separating a different combination of classes. For 3 classes (A, B, and C) 3 classifiers are necessary; one SVM classifies A from B and C, a second SVM classifies B from A and C and a third SVM classifies C from A and B. The multiclass classifier output code for a pattern is a combination of targets of all the separate SVMs. That is in the example, vectors from classes A, B, and C have codes (1,−1,−1), (−1, 1,−1), and (−1,−1, 1), respectively. If each of the separate SVMs classifies a pattern correctly, the multiclass classifier target code is met and the ECOC approach reports no error for that pattern. However, if at least one of the SVMs misclassifies the pattern, the class selected for this pattern is the one its target code closest in the Hamming distance sense to the actual output code and this may be an erroneous decision [25].

SVM sets can be classified as multi-modal and multi-probe sets. SVM multi-modal verification performance tests are compared with multi-probe tests of the individual subsystems. Appropriate selection of the parameters of the proposed quality-based scheme leads to a quality-based fusion scheme outperforming the raw fusion strategy without considering quality signals. In particular, a relative improvement of 18% is obtained for small SVM training set size by using only fingerprint quality labels.[31]

4-FINDINGS

The principle of Biometric Recognition System can be summarized as following:

First we have a registered image. This image converted to digital code. If it's necessary, this code can be encrypted according to the process and recorded to memory. Then user introduces himself to system with a device. Usually, even if it belongs to the same person, there is no similarity between the passed code and the recorded code. Many factors may influence this. Following are the most intensive factors:

- Lighting of environment

- The angle between the device and the organ.

- The purity degree and humidity of the organ and the device.

Due to this adverse effects, the entered code must be compared with recorded codes for a high certain percentage. When the required percentage binded, the person is recognized and the process is confirmed.

In this section, modeling stages are detailed. The training percentages of numeric values that calculated by wavelet in Matlab. The results will help us to understand which formula is more success .Also we will be available to see how much we close to success or away and on which filter.

In this study we will use SMO tool at Weka. In additioan that we will use Poly and RBF formulas which are most used kernel formulas.

SMO is a classification tool in Weka. Definition of SMO is Support Machines Optimization and we defined this in future levels as SVM (Support Vector Machine).

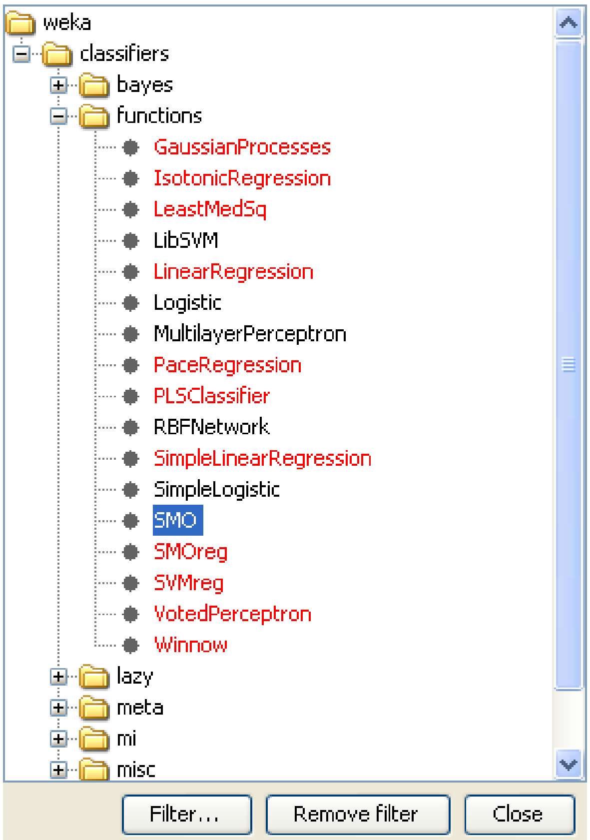

Figure 4.1 :(SMO tool at Weka)

Support Machines Optimization (SMO), which was defined as a kind of statistical learning system.

SVM merges different learning tasks and uses best methods to achieve the results. Sometimes we have to tune some parameters. It's fully automatic, no need to retouch parameters.

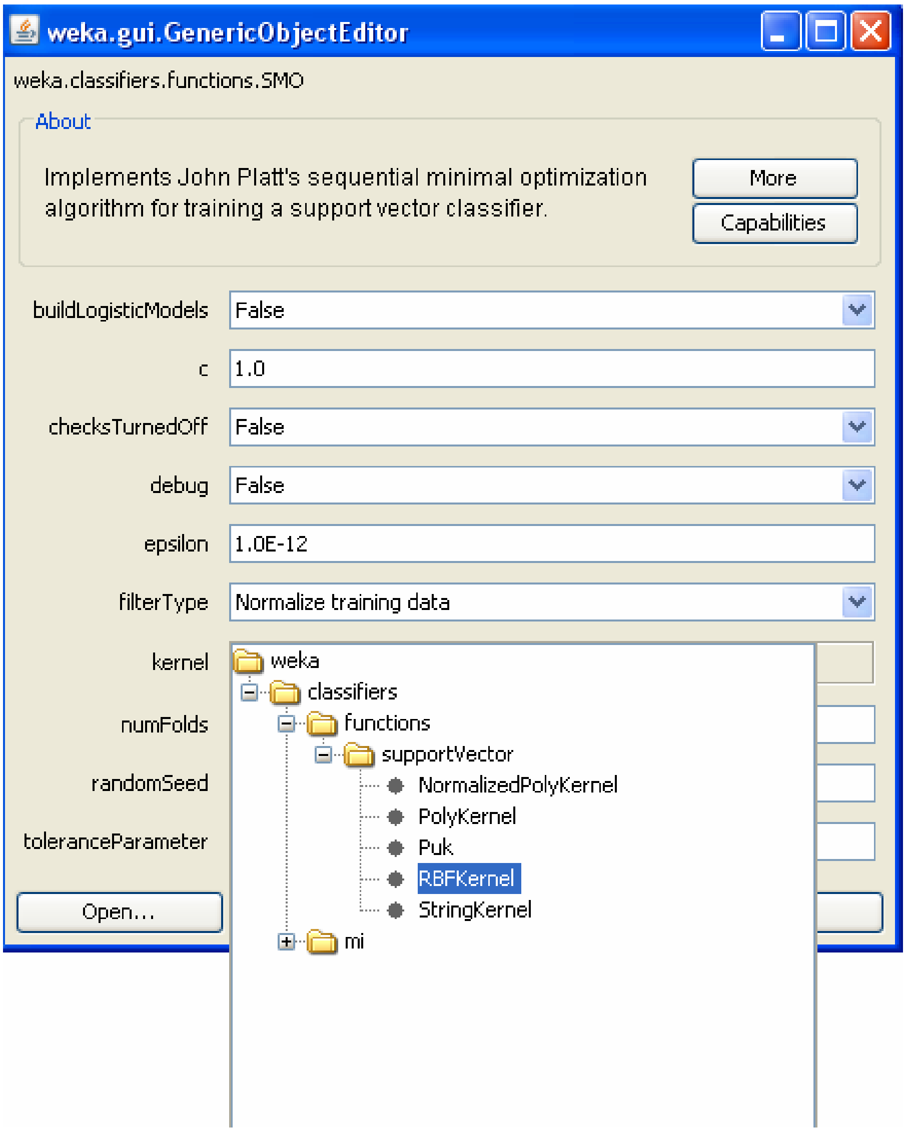

Figure-4.2 (SMO tool at Weka)

As we said before we used the db and haar filters in wavelet as filter. Those filters used in the 5.Level. But first full we got the approach results separately in Wavelet. We calculated approach results of minimum, maximum, average, standard division, variance and moment. In this case we calculated vertical, horizontal, diagonal and approximation factor values from the pictures. Also when we get maximum, minimum, standard division, variant and moment values, we had 4*6=24 for each image. First full, let's see those calculation and training success percentage. Also we mention that when we get those calculations, for each filter, we used cross validation and percentage split(75,25,50) on (haar,db) rbf and poly kernel methods…

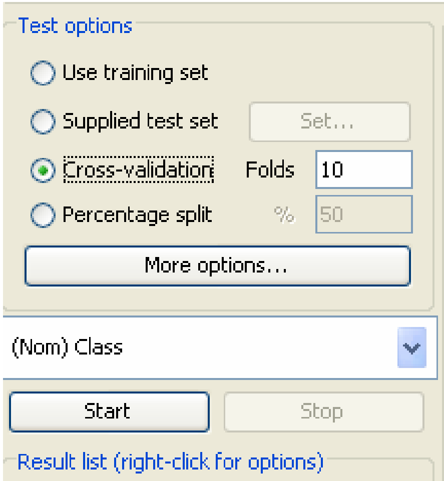

Figure 4.3 :(SMO tool at Weka)

We will see first example of training data results;

Table 4.1 Table of Classifier Output;

Methods	Correctly Classified Instances	Incorrectly Classified Instances	Total Number of Instances
Haar Filter ; (Poly-CrosValidation)	191 (%57.8788)	139 (%42.1212)	330
Haar Filter ; (Poly- Percantage Split %75)	48 (%57.8313)	35 (%42.1687)	83
Haar Filter ; (Poly- Percantage Split %25)	89 (%35.8871)	159 (%64.1129)	248
Haar Filter ; (Poly- Percantage Split %50)	73 (%44.2424)	92 (%55.7576)	165
Haar Filter ; (RBF-CrossValidation)	163 (%49.3939)	167 (%50.6061)	330
Haar Filter ; (RBF- Percantage Split %75)	31 (%37.3494)	52 (%62.6506)	83
Haar Filter ; (RBF- Percantage Split %25)	30 (%12.0968)	218 (%87.9032)	248
Haar Filter ; (RBF- Percantage Split %50)	31 (%18.7879)	134 (%81.2121)	165
DB Filter ; (Poly-CrossValidation)	191 (%57.8788)	139 (%42.1212)	330
DB Filter ; (Poly-Percantage Split %75)	48 (%57.8313)	35 (%42.1687)	83
DB Filter ; (Poly-Percantage Split %25)	89 (%35.8871)	159 (%64.1129)	248
DB Filter ; (Poly-Percantage Split %50)	73 (%44.2424)	92 (%55.7576)	165
DB Filter ; (RBF-CrossValidation)	163 (%49.3939)	167 (%50.6061)	330
DB Filter ; (RBF-Percantage Split %75)	31 (%37.3494)	52 (%62.6506)	83
DB Filter ; (RBF-Percantage Split %25)	30 (%12.0968)	218 (%87.9032)	248
DB Filter ; (RBF-Percantage Split %50)	31 (%18.7879)	134 (%81.2121)	165

Right now we will take a look data trainings that are calculated by image's approximation factors. This will give us more data then before. The reason of this, we at previous wavelet.

We are having all the factors of approximation. Actually also we could get a total value by taking all the values of horizontal, vertical, diagonal values, but we would have too many values. In this case, training would have been much difficult and too many details could make us to not recognize almost the same picture. That's why only all approximation factors are enough for training. We make the decision by considering natural trainings.

Table 4.2 :Table of Classifier Output;

Methods	Correctly Classified Instances	Incorrectly Classified Instances	Total Number of Instances
Haar Filter ; (Poly-CrosValidation)	266 (%80.6061)	64 (%19.3939)	330
Haar Filter ; (Poly- Percantage Split %75)	62 (%74.6988)	21 (%25.3012)	83
Haar Filter ; (Poly- Percantage Split %25)	114 (%45.9677)	134 (%54.0323)	248
Haar Filter ; (Poly- Percantage Split %50)	106 (%64.2424)	59 (%35.7576)	165
Haar Filter ; (RBF-CrossValidation)	161 (%48.7879)	169 (%51.2121)	330
Haar Filter ; (RBF- Percantage Split %75)	23 (%27.7108)	60 (%72.2892)	83
Haar Filter ; (RBF- Percantage Split %25)	21 (%8.4677)	227 (%91.5323)	248
Haar Filter ; (RBF- Percantage Split %50)	17 (%10.303)	148 (%89.697)	165
DB Filter ; (Poly-CrossValidation)	266 (%80.6061)	64 (%19.3939)	330
DB Filter ; (Poly-Percantage Split %75)	62 (%74.6988)	21 (%25.3012)	83
DB Filter ; (Poly-Percantage Split %25)	114 (%45.9677)	134 (%54.0323)	248
DB Filter ; (Poly-Percantage Split %50)	106 (%64.2424)	59 (%35.7576)	165
DB Filter ; (RBF-CrossValidation)	161 (%48.7879)	169 (%51.2121)	330
DB Filter ; (RBF-Percantage Split %75)	23 (%27.7108)	60 (%72.2892)	83
DB Filter ; (RBF-Percantage Split %25)	21 (%8.4677)	227 (%91.5323)	248
DB Filter ; (RBF-Percantage Split %50)	17 (%10.303)	148 (%89.697)	165

As we understand from the test results, the best success are DB Filter ; (Poly-CrossValidation) and Haar Filter ; (Poly-CrossValidation). At the SMO , we can not get the same results by using RBF model. We got the most success when the gama value was 5. But in a way we could not get the Poly Models success. Let's see that results below;

12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.RBFKernel -C 250007 -G 0.01"

Classifier output

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         123               37.2727 %
Incorrectly Classified Instances       207               62.7273 %
Kappa statistic                          0.3531
Mean absolute error                      0.0574
Root mean squared error                  0.1683
Relative absolute error                 97.7539 %
Root relative squared error             98.172  %
Total Number of Instances              330

=== Detailed Accuracy By Class ===
```

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.1	0.041	0.071	0.1	0.083	0.764	adem
1	0	1	1	1	1	ahmet
0.3	0.031	0.231	0.3	0.261	0.836	aise
0.5	0.041	0.278	0.5	0.357	0.94	akin
0.6	0.038	0.333	0.6	0.429	0.941	ali
0.2	0.013	0.333	0.2	0.25	0.937	asuman
0.3	0.041	0.188	0.3	0.231	0.844	ayse
0.7	0.028	0.438	0.7	0.538	0.974	aysekara
0.3	0.019	0.333	0.3	0.316	0.829	aysenur
0.9	0.022	0.563	0.9	0.692	0.984	ayseoksuz
0.2	0.016	0.286	0.2	0.235	0.856	berrak
0.4	0.016	0.444	0.4	0.421	0.95	cetin
0.8	0.044	0.364	0.8	0.5	0.953	deniz
0.3	0.019	0.333	0.3	0.316	0.864	elif

Figure 4.4 :(Haar Filter ; (RBF-CrossValidation);Gama=0.01)

At the figure above, we see the success ratio where the image values that has been converted to digital code with haar filter,using SMO with rbf formula and choosing 0,01 gama value.

```
-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.RBFKernel -C 250007 -G 5.0"

Classifier output
=== Summary ===

Correctly Classified Instances         218               66.0606 %
Incorrectly Classified Instances       112               33.9394 %
Kappa statistic                          0.65
Mean absolute error                      0.0571
Root mean squared error                  0.1675
Relative absolute error                 97.2124 %
Root relative squared error             97.736  %
Total Number of Instances              330

=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
  0.4       0.034      0.267     0.4       0.32       0.806    adem
  0.9       0          1         0.9       0.947      0.999    ahmet
  0.5       0.034      0.313     0.5       0.385      0.876    aise
  0.8       0.009      0.727     0.8       0.762      0.975    akin
  0.8       0          1         0.8       0.889      0.971    ali
  0.5       0.009      0.625     0.5       0.556      0.97     asuman
  0.2       0.019      0.25      0.2       0.222      0.833    ayse
  0.6       0.009      0.667     0.6       0.632      0.968    aysekara
  0.4       0.019      0.4       0.4       0.4        0.766    aysenur
  0.8       0          1         0.8       0.889      0.995    ayseoksuz
  0.6       0.003      0.857     0.6       0.706      0.852    berrak
  0.6       0.006      0.75      0.6       0.667      0.976    cetin
  0.9       0          1         0.9       0.947      0.999    deniz
  0.9       0.009      0.75      0.9       0.818      0.993    elif
  0.2       0.028      0.182     0.2       0.19       0.846    erol
  0.9       0.009      0.75      0.9       0.818      0.986    isa
```

Figure 4.5 :(Haar Filter ; (RBF-CrossValidation);Gama=5.0)

At the figure above, we see the success ratio where the image values that has been converted to digital code with haar filter,using SMO with rbf formula and choosing 5,0 gama value.

```
12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.RBFKernel -C 250007 -G 10.0"

Classifier output

Time taken to build model: 125.36 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         161               48.7879 %
Incorrectly Classified Instances       169               51.2121 %
Kappa statistic                          0.4719
Mean absolute error                      0.0576
Root mean squared error                  0.169
Relative absolute error                 98.0528 %
Root relative squared error             98.5956 %
Total Number of Instances              330

=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
  0.2       0.019      0.25      0.2       0.222      0.616    adem
  0.3       0          1         0.3       0.462      0.907    ahmet
  0.4       0.05       0.2       0.4       0.267      0.781    aise
  0.5       0.009      0.625     0.5       0.556      0.898    akin
  0.6       0          1         0.6       0.75       0.913    ali
  0.7       0          1         0.7       0.824      0.9      asuman
  0.2       0.028      0.182     0.2       0.19       0.608    ayse
  0.5       0.009      0.625     0.5       0.556      0.768    aysekara
  0.3       0.013      0.429     0.3       0.353      0.687    aysenur
```

Figure 4.6 :(Haar Filter ; (RBF-CrossValidation);Gama=10.0)

At the figure above, we see the success ratio where the image values that has been converted to digital code with haar filter,using SMO with rbf formula and choosing 10,0 gama value.

The success is getting down after the 5 gama value and after 10 , success is almost the same.

4.1 Analysis of The Output With A Simple Example

Confusion Matrix;

A confusion matrix is an easy way of describing the results of the experiment. The best way to describe it is by example. Returning to the birth weight example, the following table was generated.

Table 4.3 : Sample of Confusion Matrix

Confusion Matrix		Class	
123	7	a	a=normal
39	11	b	b=low

The columns represent the predictions, and the rows represent the actual class. It shows that 123 instances were correctly predicted as normal birth weights. These cases are also known as "True Positives". The table also shows that 11 instances were correctly predicted as low weights. These cases are also known as "True Negatives". Correct predictions always lie on the diagonal of the table.

On the other hand, it shows that 39 instances were predicted as normal birth weight when they were in fact low weight. These cases are also known as "False Positives". Lastly, it shows 7 instances that were incorrectly predicted as low weight. These cases are also known as "False Negatives". The negative positive terminology is only useful if the class variable has 2 levels, with one level designated positive and the other negative.

It can be seen here that the model was not very good at predicting low birth weight cases.

This is most likely a symptom of the fact that most of the cases were in fact normal weight so the tree was made more sensitive to this class. One should in general adjust misclassification costs and threshold levels so that sufficient accuracy and sensitivity in the desired class is obtained.

4.1.1 Detailed Accuracy by Class

This portion of the text output has been extracted and is displayed below and the results (from the birth weight example) are shown below:

Table 4.4 : Detailed Accuracy By Class

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0,946	0,78	0,759	0,946	0,842	normal
0,22	0,054	0,611	0,22	0,324	low

The first two columns are the TP Rate (True Positive Rate) and the FP Rate (False Positive Rate). For the first level where 'weight=low' TP Rate is the ratio of low weight cases predicted correctly cases to the total of positive cases. There were 123 instances correctly predicted as low weight, and 130 instances in all that were low weight. So the TP Rate = 123/130 = 0.946. The FP Rate is then the ratio normal weight cases of incorrectly predicted as low weight cases to the total of normal weight cases. 39 normal weight instances were predicted as low weight and there were 50 normal weight cases in all. So the FP Rate is 9/50=0.78

The next two columns are terms related to information retrieval theory. When one is conducting a search for relevant documents, it is often not possible to get to the relevant documents easily or directly. In many cases, a search will yield lots results many of which will be irrelevant. Under these circumstances, it is often impractical to get all results at once but only a portion of them at a time. In such cases, the terms recall and precision are important to consider. Recall is the ratio of relevant documents found in the search result to the total of all relevant documents. Thus, higher recall values imply that relevant documents are returned more quickly. A recall of 30% at 10% means that 30% of the relevant documents were found with only 10% of the results examined. Precision is the proportion of relevant documents in the results returned.[123/(123+39)] Thus a precision of 0.75 means that 75% of the returned documents were relevant. Lastly, the F-measure is a way of combining recall and precision scores into a single measure of performance. The formula for it is (2*recall*precision)/(recall+precision)

In the context described above, these measures are important elements to consider when studying the performance of a certain model in the domain of informational search and retrieval. In our birth weight example, such measures are not very applicable…the recall in this case just corresponds to the TP Rate, as we are always looking at 100% of test sample and precision is just the proportion of low and normal weight cases in the test sample.

4.1.2 Summary

The output below is the final segment of the text output produced by the explorer.

Table 4.5 : Summary

Correctly Classified Instances	**Incorrecty Classified Instances**	**Kappa Static**	**Mean Absolute Error**	**Root Mean Squard Error**	**Relative Absolute Error**	**Root Relative Squared Error**	**Total Number**
134 74,4444 %	46 25.5555 %	0,2069	0,3632	0,44	90,24%	98,24%	180

In our case since our class variable is nominal, the first 2 lines are most useful. The first line shows the number and percentage of cases that were correctly predicted. The second line shows the number and percentage of cases that the classifier predicted incorrectly.

The third line shows the kappa statistic, which measures the agreement of predictions with the actual class. This statistic is not very informative as it can have low values even when there are high levels of agreement as in the case above. In general, Kappa statistics are only appropriate for testing whether agreement exceeds chance levels, i.e. that predictions and actual classes are correlated. Since classifiers are designed and intended to be correct in their predictions, the Kappa statistic is not very helpful. It will usually find that predictions and actual classes are correlated and even a weak classifier will tend to show a correlation between the two.

4.2 Analyzing An Output Which Has A Higher Validation Percentage By Real Measurements

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         266               80.6061 %
Incorrectly Classified Instances        64               19.3939 %
Kappa statistic                          0.8
Mean absolute error                      0.057
Root mean squared error                  0.1672
Relative absolute error                 97.0597 %
Root relative squared error             97.5295 %
Total Number of Instances              330

=== Detailed Accuracy By Class ===
```

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.6	0.022	0.462	0.6	0.522	0.88	adem
1	0	1	1	1	1	ahmet
0.6	0.022	0.462	0.6	0.522	0.944	aise
0.8	0.003	0.889	0.8	0.842	0.998	akin
0.9	0.003	0.9	0.9	0.9	0.999	ali
0.9	0.006	0.818	0.9	0.857	0.992	asuman
0.8	0.009	0.727	0.8	0.762	0.964	ayse
0.8	0.006	0.8	0.8	0.8	0.987	aysekara
0.7	0.006	0.778	0.7	0.737	0.869	aysenur
0.8	0	1	0.8	0.889	0.989	ayseoksuz
0.6	0.006	0.75	0.6	0.667	0.932	berrak
0.9	0.013	0.692	0.9	0.783	0.994	cetin
1	0.006	0.833	1	0.909	0.998	deniz
1	0.003	0.909	1	0.952	1	elif

0.6	0.013	0.6	0.6	0.6	0.959	erol
0.9	0.009	0.75	0.9	0.818	0.992	isa
0.7	0.006	0.778	0.7	0.737	0.955	kader
0.5	0	1	0.5	0.667	0.856	mehmetaca
0.9	0.016	0.643	0.9	0.75	0.985	mehmeteser
1	0	1	1	1	1	merve
0.9	0.003	0.9	0.9	0.9	0.984	mesut
0.4	0	1	0.4	0.571	0.932	murat
0.9	0.003	0.9	0.9	0.9	0.998	rahmi
0.9	0	1	0.9	0.947	1	ramazan
0.9	0.006	0.818	0.9	0.857	0.985	said
0.7	0	1	0.7	0.824	0.957	selin
0.7	0	1	0.7	0.824	0.999	sevilay
0.9	0.006	0.818	0.9	0.857	0.994	sibel
0.8	0.009	0.727	0.8	0.762	0.984	sumeyra
1	0	1	1	1	1	umit
0.9	0.003	0.9	0.9	0.9	0.996	yakup
0.8	0.013	0.667	0.8	0.727	0.906	yasemin
0.8	0.006	0.8	0.8	0.8	0.985	zafer

Figure 4.7 :(Detailed Accuary By Class of Perfect Example)

=== Confusion Matrix ===

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	aa	ab	ac	ad	ae	af	ag	<-- classified as
6	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	a = adem
0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	b = ahmet
0	0	6	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	c = aise
1	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	d = akin
0	0	0	0	9	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	e = ali
0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	f = asuman
0	0	0	0	0	1	8	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	g = ayse
0	0	0	0	1	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	h = aysekara
1	0	1	0	0	0	0	0	7	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	i = aysenur
1	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	j = ayseoksuz
1	0	1	0	0	0	0	1	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	k = berrak
0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	l = cetin
0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	m = deniz
0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	n = elif
0	0	0	0	0	1	1	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	o = erol
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	p = isa
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	q = kader
0	0	1	0	0	0	1	0	0	0	0	1	2	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r = mehmetaca
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	s = mehmeteser
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	t = merve
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	u = mesut
0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	2	0	0	4	0	0	1	0	0	0	0	0	0	1	0	v = murat

```
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  9  0  0  0  0  0  0  0  0  0  0 |   w = rahmi
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  9  0  0  0  0  0  0  0  0  0 |   x = ramazan
 0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0 |   y = said
 1  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  7  0  1  0  0  0  0  0 |   z = selin
 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  1  0  7  0  0  0  0  0  0 |  aa = sevilay
 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  9  0  0  0  0  0 |  ab = sibel
 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  8  0  0  0  0 |  ac = sumeyra
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 10  0  0  0 |  ad = umit
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  9  0  0 |  ae = yakup
 0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  8  0 |  af = yasemin
 0  0  1  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  8 |  ag = zafer
```

Figure 4.8 :(Confusion Matrix By Class of Perfect Example)

The results in the Figure 4.7 and 4.8 have the highest success rate with the real inputs (The digital data converted from images).

The processes below are the classes' attributes we gained by taking values from Figure 4.8 of confusion matrix.

Table 4.6 : The classes' attributes of confusion matrix.

Class	**TP**	**FP**	**Precision**	**Recall**	**F-Measure**
Adem	6/10=**0.6**	7/320-4=7/316=**0.022**	6/13=**0.462**	TP=**0.6**	(2*(0.462*0.6))/ (0.462+0.6)=**0.522**
Ahmet	10/10=**1**	0/320=**0**	10/10=**1**	TP=**1**	(2*(1*1))/ (1+1)=**1**
Aise	6/10=**0.6**	7/320-4=7/316=**0.022**	6/13=**0.462**	TP=**0.6**	(2*(0.462*0.6))/ (0.462+0.6)=**0.522**
Akin	8/10=**0.8**	1/320-2=1/318=**0.003**	8/9=**0.889**	TP=**0.8**	(2*(0.889*0.8))/ (0.889+0.8)=**0.842**
Ali	9/10=**0.9**	1/320-1=1/319=**0.003**	9/10=**0.9**	TP=**0.9**	(2*(0.9*0.9))/ (0.9+0.9)=**0.9**
Asuman	9/10=**0.9**	2/320-1=2/319=**0.006**	9/11=**0.818**	TP=**0.9**	(2*(0.818*0.9))/ (0.818+0.9)=**0.857**
Ayse	8/10=**0.8**	3/320-2=3/318=**0.009**	8/11=**0.727**	TP=**0.8**	(2*(0.727*0.8))/ (0.727+0.8)=**0.762**
Aysekara	8/10=**0.8**	2/320-2=3/318=**0.006**	8/10=**0.8**	TP=**0.8**	(2*(0.8*0.8))/ (0.8+0.8)=**0.8**
Aysenur	7/10=**0.7**	2/320-3=3/317=**0.006**	7/9=**0.778**	TP=**0.7**	(2*(0.778*0.7))/ (0.778+0.7)=**0.737**
Ayseoksuz	8/10=**0.8**	0/320-2=0/318=**0**	8/8=**1**	TP=**0.8**	(2*(1*0.8))/ (1+0.8)=**0.889**
Berrak	6/10=**0.6**	2/320-4=2/316=**0.006**	6/8=**0.75**	TP=**0.6**	(2*(0.75*0.6))/ (0.75+0.6)=**0.667**
Cetin	9/10=**0.9**	4/320-1=4/319=**0.013**	9/13=**0.692**	TP=**0.9**	(2*(0.692*0.9))/ (0.692+0.9)=**0.783**
Deniz	10/10=**1**	2/320-0=2/320=**0.006**	10/12=**0.883**	TP=**1**	(2*(0.883*1))/ (0.883+1)=**0.909**
Elif	10/10=**1**	1/320-0=1/320=**0.003**	10/11=**0.909**	TP=**1**	(2*(0.909*1))/ (0.909+1)=**0.952**
Erol	6/10=**0.6**	4/320-4=4/316=**0.013**	6/10=**0.6**	TP=**0.6**	(2*(0.6*0.6))/ (0.6+0.6)=**0.6**
Isa	9/10=**0.9**	3/320-1=3/319=**0.009**	9/12=**0.75**	TP=**0.9**	(2*(0.75*0.9))/ (0.75+0.9)=**0.818**
Kader	7/10=**0.7**	2/320-3=2/317=**0.006**	7/9=**0.778**	TP=**0.7**	(2*(0.778*0.7))/ (0.778+0.7)=**0.737**
Mehmetaca	5/10=**0.5**	0/320-5=0/315=**0**	5/5=**1**	TP=**0.5**	(2*(1*0.5))/ (1+0.5)=**0.667**
Mehmeteser	9/10=**0.9**	5/320-1=5/319=**0.016**	9/14=**0.643**	TP=**0.9**	(2*(0.643*0.9))/ (0.643+0.9)=**0.75**
Merve	10/10=**1**	0/320-0=0/320=**0**	10/10=**1**	TP=**1**	(2*(1*1))/ (1+1)=**1**
Mesut	9/10=**0.9**	1/320-1=1/329=**0.003**	9/10=**0.9**	TP=**0.9**	(2*(0.9*0.9))/ (0.9+0.9)=**0.9**
Murat	4/10=**0.4**	0/320-6=0/314=**0**	4/4=**1**	TP=**0.4**	(2*(1*0.4))/ (1+0.4)=**0.571**
Rahmi	9/10=**0.9**	1/320-1=1/319=**0.003**	9/10=**0.9**	TP=**0.9**	(2*(0.9*0.9))/ (0.9+0.9)=**0.9**
Ramazan	9/10=**0.9**	0/320-1=0/319=**0**	9/9=**1**	TP=**0.9**	(2*(1*0.9))/ (1+0.9)=**0.947**

Said	9/10=**0.9**	2/320-1=2/319=**0.006**	9/11=**0.818**	TP=**0.9**	(2*(0.818*0.9))/ (0.818+0.9)=**0.857**
Selin	7/10=**0.7**	0/320-3=0/317=**0**	7/7=**1**	TP=**0.7**	(2*(1*0.7))/ (1+0.7)=**0.824**
Sevilay	7/10=**0.7**	0/320-3=0/317=**0**	7/7=**1**	TP=**0.7**	(2*(1*0.7))/ (1+0.7)=**0.824**
Sibel	9/10=**0.9**	2/320-1=2/319=**0.006**	9/11=**0.818**	TP=**0.9**	(2*(0.818*0.9))/ (0.818+0.9)=**0.857**
Sumeyra	8/10=**0.8**	3/320-2=3/318=**0.009**	8/11=**0.727**	TP=**0.8**	(2*(0.727*0.8))/ (0.727+0.8)=**0.762**
Umit	10/10=**1**	0/320-0=0/320=**0**	10/10=**1**	TP=**1**	(2*(1*1))/ (1+1)=**1**
Yakup	9/10=**0.9**	1/320-1=1/319=**0.003**	9/10=**0.9**	TP=**0.9**	(2*(0.9*0.9))/ (0.9+0.9)=**0.9**
Yasemin	8/10=**0.8**	4/320-2=4/318=**0.013**	8/12=**0.667**	TP=**0.8**	(2*(0.667*0.8))/ (0.667+0.8)=**0.727**
Zafer	8/10=**0.8**	2/320-2=2/318=**0.006**	8/10=**0.8**	TP=**0.8**	(2*(0.8*0.8))/ (0.8+0.8)=**0.8**

4.3 Roc Plot By Real Measurements

The ROC graph is a comprised of a set of (TPR,FPR) points, one for each unique threshold level in the similarity data.

Sweep threshold and plot

- TPR vs. FPR
- Sensitivitiy vs. 1- Specificity
- P(true|true) vs. P(true|false)
- Sensitivity=a/a+b=Recall=Lift numarato
- 1- Specificity = 1-d/(c+d)

Real Measurements;

In example by using Figure 4.8 (Confusion Matrix By Class of Perfect Example) for adem class we calculated Sensitivitiy and Specificity values as we see following.

a=6

b=4

a/a+b=6/6+4=0,6

c=7

d=10+12+9+10+11+11+10+9+8+8+13+12+11+10+12+9+5+14+10+10+4+10+9+11+7+7+11+11+10+10+12+10=316

d/d+c=316/323=0,978

Table 4.7 :(The result of SVM classifier for each class)

Class	**Sensivitiy**	**Specificty**	**1-Specificty**	**TPR**	**FPR**
adem	0,6	0,978	0,022	0,6	0,022
ahmet	1	1	0	1	0
aise	0,6	0,978	0,022	0,6	0,022
akin	0,8	0,996	0,004	0,8	0,004
ali	0,9	0,996	0,004	0,9	0,004
asuman	0,9	0,993	0,007	0,9	0,007
ayse	0,8	0,99	0,01	0,8	0,01
aysekara	0,8	0,993	0,007	0,8	0,007
aysenur	0,7	0,993	0,007	0,7	0,007
ayseoksuz	0,8	1	0	0,8	0
berrak	0,6	0,993	0,007	0,6	0,007
cetin	0,9	0,987	0,013	0,9	0,013
deniz	1	0,993	0,007	1	0,007
elif	1	0,996	0,004	1	0,004
erol	0,6	0,987	0,013	0,6	0,013
isa	0,9	0,99	0,01	0,9	0,01
kader	0,7	0,993	0,007	0,7	0,007
mehmetaca	0,5	1	0	0,5	0
mehmeteser	0,9	0,984	0,016	0,9	0,016
merve	1	1	0	1	0
mesut	0,9	0,996	0,004	0,9	0,004
murat	0,4	1	0	0,4	0
rahmi	0,9	0,996	0,004	0,9	0,004
ramazan	0,9	1	0	0,9	0
said	0,9	0,993	0,007	0,9	0,007
selin	0,7	1	0	0,7	0
sevilay	0,7	1	0	0,7	0
sibel	0,9	0,993	0,007	0,9	0,007
sumeyra	0,8	0,99	0,01	0,8	0,01
umit	1	1	0	1	0
yakup	0,9	0,996	0,004	0,9	0,004
yasemin	0,8	0,987	0,013	0,8	0,013
zafer	0,8	0,993	0,007	0,8	0,007

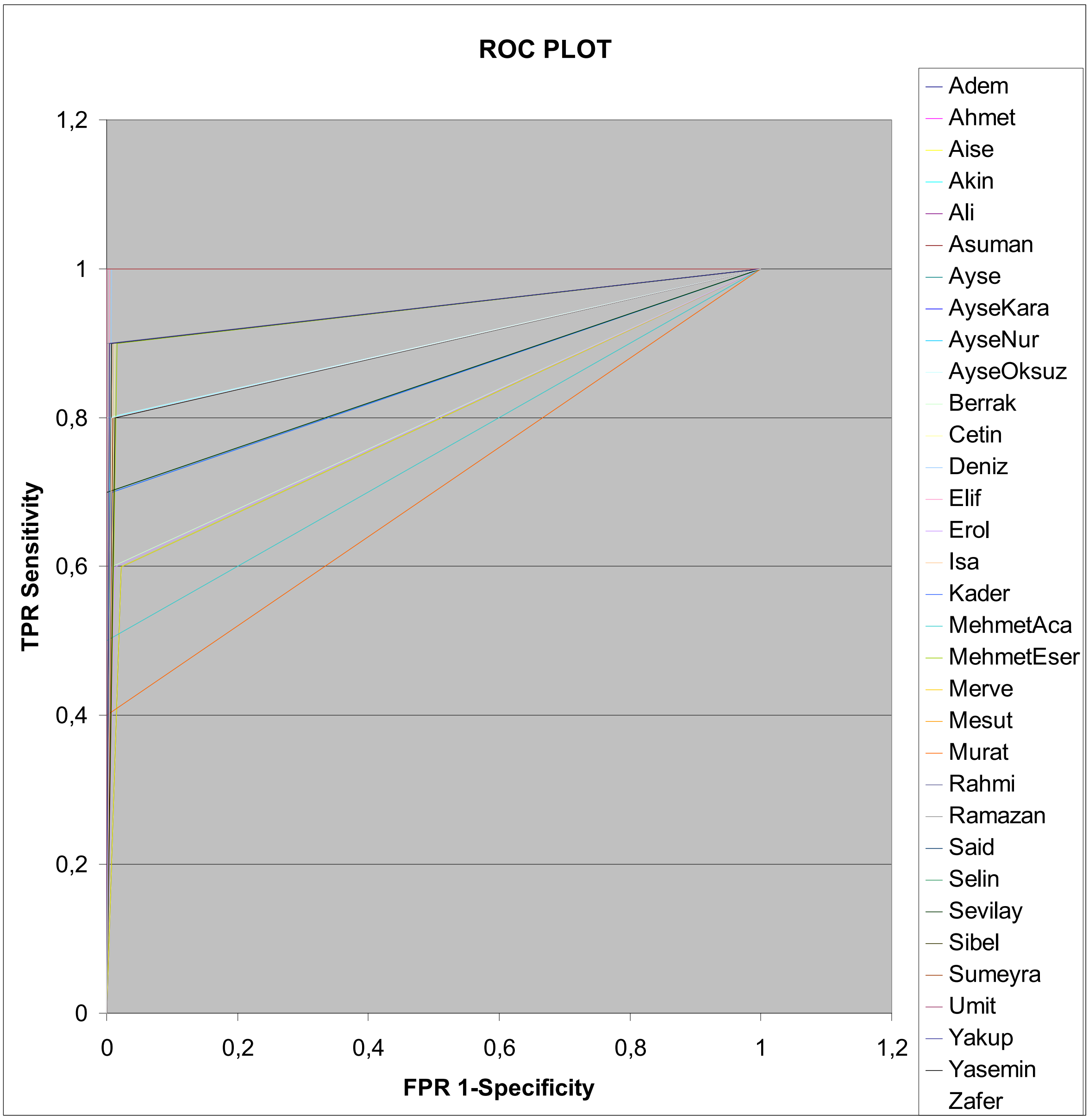

Figure 4.9 :(ROC Curve for each classes with SVM classifier)

This graphic (Figure 4.9) is gained by the intersection of Sensitivity (TPR) and Specificity (FPR) values with 1 point in the table 4.6. As you see in Figure 4.9 we draw the roc pilot for every class. And in Figure 4.10 an average roc pilot is shown for the data set containing 330 images and 33 classes. Average of TPR sensitivity is 0,806 and average of FPR 1-specificity is 0,0065. Also the average roc plot in Figure 4.10 consists of those values.

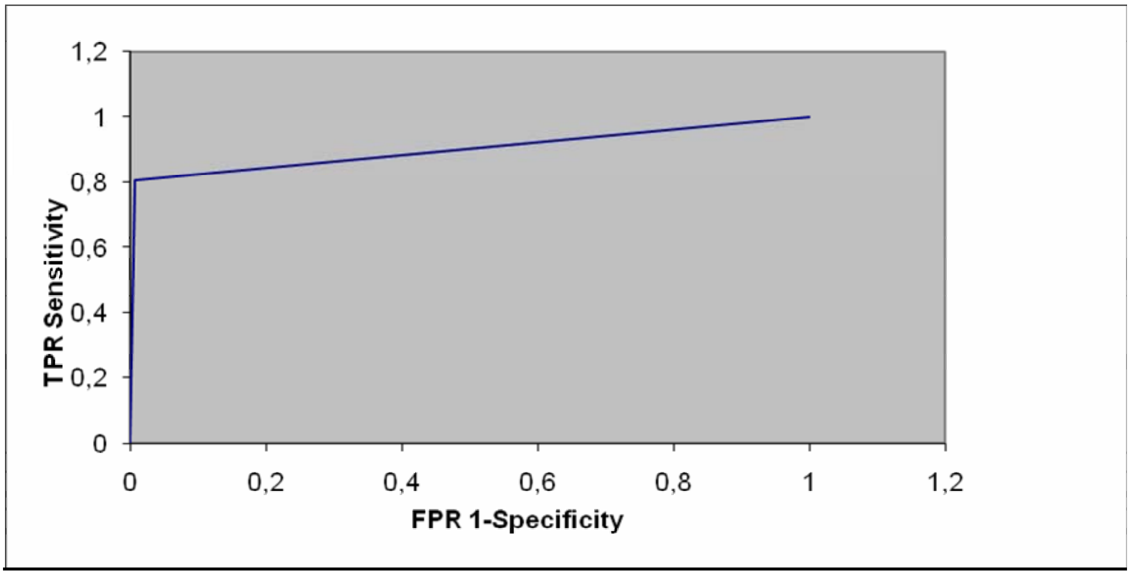

Figure 4.10 :(ROC Curve for average all of classes with SVM classifier)

5-CONCLUSION

We have trained the digital values that the images we passed through on wavelet. During thus process have been going on, we passed the values to the related classes which we gained from wavelet and we used some filters from weka. We found the formula that the most successful. Our intention was gaining %75 of success rate.

In this thesis in weka we used SMO. We used POLY and RBF kernel formulas which SMO determined. For filtering, we used DB and HAAR filters in fifth scale. But we first have the approach values respectively in wavelet.

We have the min, max, average, standard deviation, variance and moment values. Primarily we obtained the coefficient's of vertical, horizontal, diagonal and approximation values from images. There were 24 values for each of images when we got min, max, avrg., variance and moment values respectively. With the coefficients which we got, we figure outed success rate of the training in Weka. No wonder it's necessary to explain that; while this process, for each fitler in RBF and POLY kernel methods, cross validation and percentage split(75,25,50) have used. The success rates in here didn't give us the results we expected.

After this we have only trained the datas that we got the approximation coefficient's. This gave us more data than we explained overhead. Because before we limited the values with six parameters(like max., min., stand., var. etc…) and the percentages were nominal.

We got the all coefficient values of approximation. In fact as well as this, we could have obtained aggregate of the image's vertical, horizontal and diagonal values. But there would be many values and the training would be harder than before because of the detailed values. Therefore only approximation coefficients are enough for this training. We resolved the adequation of this from our success results of training.

It appears by test results that the best success achieved by db, haar filter. We didn't have the same results when we used SVM so RBF model in SMO. When we started to increasing the gama values by 0.01, we gained the best success when the gama value was

5. But after all we couldn’t have the success of poly model. By our observations the success rate starts to falling after we pass to gama more than 5. And the treshold value for gama is 10. After 10 the gama value settles down.

Our most succesfull achievement is obtained using DB Filter ; (Poly-CrossValidation) and Haar Filter ; (Poly-CrossValidation) while exponent is set to 2 and we got success rate of %80.60

REFERENCES

[1]-Catten Ely , Oct20,2004.”Crime Stories”, Master Thesis at University of Houston.

[2]-Elena Battini SÖNMEZ, Nilay Özge ÖZBEK, Önder ÖZBEK, 2008. ” A Multi Modal Biometric System Based On Wavelet Transformation”, Akademik Bilişim - Çanakkale Onsekiz Mart University

[3]-Jean-François Mainguet, 2004-2007. “Fingerprint Algorithms”, School of Computing and Mathematical Sciences - Liverpool John Moores University – UK

[4]-The MathWorks ,Technical Solutions, Feb23, 2007. Official MATLAB and SIMULINK Products’ site

[5]-Kristian Sandberg, Apr01, 2000 .“The Haar Wavelet Transform”, University of Otago New Zealand

[6]-Kristian Sandberg, Apr14, 2000. “The Daubechies Wavelet Transform”, University of Otago New Zealand

[7]-Elif Derya Übeyli, Sept12,2007. ” Multiclass support vector machines” , TOBB Economy and Technology University

[8]-Davide Maltoni, Anil K. Jain, Salil Prabhakar, June 2003. ”Handbook of Fingerprint Recognition”, Springer - ISBN 0387954317, p1-p348

[9]-Nalini Rahta- Ruud Bolle,Sept12,2003. ”Automatic Fingerprint Recognition Systems”, Springer - ISBN 0387955933, p1-p458

[10]-Henry J. Lee, Robert E. Gaenslen, Jun 16 2001. ”Advances in Fingerprint Technology”, CRC Press - ISBN 0849309239, p1-p444

[11]-Simon A. Cole , July 1 2004. "History of Fingerprint Pattern Recognition", Web access, ncstl.org

[12] A.R Calderbank,Ingrid Daubeshies, Wim Sweldens, and Boon Lock Yeo, August 1996. "Wavelet transforms that map integers to integers" , Appl. Comput. Harmon. Anal, ISSN-1063-5203, p332-p369

[13]-Zbigniew R. Struzik and Arno Siebes, March 1999. "The Haar Wavelet Transform in Time Series Similarity Paradigm", Lecture Notes In Computer Science, ACM, ISBN:3-540-66490-4, p12-22

[14]-Agostino Abbate, Casime DeCusatis, Pankaj K. Das , Oct 2001. “Wavelets and Subbands”, Springer, ISBN 081764136X, p1-p551

[15]-Hakob Sarukhanyan and Arthur Petrosian, April 19, 2005. “Construction and Application of Hybrid Wavelet and Other Parametric Orthogonal Transforms”, National Armenian Academy of Sciences, Institute for Informatics and Automation Problems, Yerevan, Armenia. ISSN:0924-9907

[16]-Hae-Gil Hwang,Hyung-Ju Choi,Byeong-II Lee, Hye-Kyoung Yoon, Dec 7 2004. “Multi-resolution wavelet-transformed image analysis of histological sections of breast carcinomas”, Cellular Oncology Publisher, IOS Press, ISSN-1570-5870, p237-p244

[17]-Claypoole, R.L., Jr. Baranjuk, R.G Nowak, R.D Dept. Of Elctr & Comput. Eng.,Rice Univ.,Houston, TX, 2002-08-06.“Lifting construction of non-linear wavelet transforms”, Time-Frequency and Time-Scale Analysis, 1998. Proceedings of the IEEE-SP International Symposium, p49-p52

[18]-Andre Augusto Spadotto, Jao Paulo Papa, Ana Rita Gatto,Paula Cristina Cola, Jose Carlos Pereira, Rodrigo Capobianco Guido, Arthur Oscar Schelp, November 2007. “Denoisint Swallowing Sound to İmprove to İmprove the Evaluartor’s Qualitative Analysis”, Computers and Electrical Engineering archive, ACM, ISSN:0045-7906, p148-p153

[19]-Addisson, P. S.; Ng, J.; Kingsbury, N. G. , 2004. “The Illustrated Wavelet Transform Handbook”, CRC Press, ISBN 0750306920, p1p353

[20]-Ufuk Adıgüzel, Münevver Yeşilli, Ayça Sarı, Tülin Kervancı,Özay Öz , Jun 6 2001. “Effect of Pterygium surgery on corneal regular and irregular a stigmatism: Evalution with fourier analysis” , Medical faculty at Mersin University

[21]-Hans G. Feichtinger and Ferenc Weisz, Aug 6 1995. “Inversion Formulas for the Short-Time Fourier Transform”, ISSN - 1050-6926, p507-p521

[22]-Bruno Buchberger -“Artificial intelligence and symbolic computation” Sept 6 2001, Artificial intelligence and symbolic computation: 7th international conference, AISC 2004, Linz, Austria

[23]-H.F. Ates, M.T. Orchard, 15 July 2001. “Spherical Coding Algorithm for Wavelet Image Compression”, Image Processing, IEEE Transactions on In Image Processing, IEEE Transactions on, Vol. 18, No. 5. (2009), pp. 1015-1024.

[24]-Wornell, G.W. , Oct 1993. “Wavelet-based representations for the 1/f family offractal processes”, Proceedings of the IEEE, ISSN: 0018-9219, p1428-p1450

[25]-Kulik, Chen-Lin C. Kulik, James A. ,1991. “Effectiveness of computer-based instruction”, The educational effectiveness of computer-based instruction Source Computers & Geosciences archive, ACM, ISSN:0098-3004, p677-p682

[26]-Jensen, Arne; Jensen, A.; Cour-Harbo, Anders La, January 2001. "Ripples in Mathematics: The Discrete Wavelet Transform", ISBN: 3540416625, p1-p246

[27]-Henry C. Lee and R. E. Gaensslen ,2001. “Advances in Fingerprint Technology”, Advances in Fingerprint Technology, CRC Press, ISBN 0849309239, p1-p444

[28]-Jain, A.K,Lin Hong,Pnakanti, S.Bolle A. Moenssens, 2002. “An identity-authentication system using fingerprints”, Proceedings of the IEEE, p1365 – 1388

[29]-Simon A. Cole, 2001. “Suspect Identities”, Law and Society at University of California, Irvine. Harvard edition World, p1-p384

[30]-Yuan Yao, Paolo Frasconi and Massimiliano Pontil , January , 2001. “Audio- and Video-Based Biometric Person Authentication”, Book Series Lecture Notes in Computer Science, Springer, Berlin. ISSN-0302-9743

[31]-J. Fierrez-Aguilara, J. Ortega-Garciaa, J. Gonzalez-Rodrigueza and Josef Bigunb, 2002. “Kernel-based multimodal biometric verification using quality signals”, Proceedings of the SPIE, Volume 5404, p544-p554

[32]-Shah, S.,Sastry, P.S. , 2004. “Fingerprint classification using a feedback-based line detector”, IEEE Transactions, p85 - p94

[33]-Tokiwadai, Hodogaya, Yokohama, 2002. “Impact of Artificial "Gummy" Fingers on Fingerprint Systems ” , Graduate School of Environment and Information Sciences Yokohama National University, 240-8501, Japan