BREAST IMAGE CLASSIFICATION USING MACHINE LEARNING TECHNIQUES

Jonathan Wu

Bachelor of Engineering
Computer Engineering

Department of Electronic Engineering
Macquarie University

November 7, 2016

Dr. Yinan Kong

ACKNOWLEDGMENTS

I would like to acknowledge the help from my supervisor Dr Yinan Kong and co-supervisor Abdullah-Al Nahid for their guidance and understanding throughout my thesis project at Macquarie University.

STATEMENT OF CANDIDATE

I, Jonathan Wu, declare that this report, submitted as part of the requirement for the award of Bachelor of Engineering in the Department of Engineering, Macquarie University, is entirely my own work unless otherwise referenced or acknowledged. This document has not been submitted for qualification or assessment in any academic institution.

Student's Name: Jonathan Wu

Student's Signature:

Date: 7 November, 2016

ABSTRACT

Among all the common forms of cancer, breast cancer is the most prevalent one. Statistics shows that breast cancer causes the second most mortality in women worldwide. Accurate classification of the breast cancer is of high importance for the proper diagnosis. The survival rate of the breast cancer affected people is greatly increased if the cancer is early detected. Ultrasound, MRI or CT imaging techniques are used to capture the current condition of the breast. Digital signal processing along with the modern computer aided techniques are used to analyse these images, and proper diagnosis of the images can save the diagnoses time of the doctors. For the correct classification of the images, feature detection and extraction of the features play a vital role. Various feature detection techniques are available, but for this thesis, we will use the wavelet transform to perform feature detection. The features extracted from the images using the wavelet transform will be utilised for the classification purposes. For the classification task, we will use support vector machine.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Breast cancer is the most common form of cancer for women [12]. Early detection of cancer will significantly increase the survival rate of cancer affected people. To help doctors with detecting and diagnosing cancer affected people, artificial intelligence and machine learning techniques are being used to increase the accuracy and speed in diagnosing breast cancer patients. With machine learning, the breast mammogram will need to be examined to extract features of the mammogram to allow the machine to understand the image.

The Wavelet Transform is a method used in some different applications, including the ability to extract features from images, in this case mammograms. The features extracted on the mammograms will be utilised as training data for the classifier. The classifier in question is the Support Vector Machine (SVM), where is has been used in some different applications and has been reported as an accurate classifier. This thesis will use MatLAB to program and implement both techniques to classify breast cancer and the MatLAB toolbox ScatNet for the wavelet scattering network.

1.1 Techniques Used

1.1.1 Wavelet and Wavelet Transform

The Fourier theory is that a signal can be represented as a possibly infinite series of sine and cosine waves. The Fourier Transform is, therefore, a useful tool to analyse the frequency components of a signal and has been used by engineers extensively. However, the Fourier theory has some significant disadvantages. While the Fourier transform can represent a signal into a series of sine and cosines, it cannot determine at what instance a particular frequency arises [7, 17].

There have been several solutions to overcome the problem of getting the time and frequency domain at the same time. One solution was to cut the signal into separate parts and to analyse the individual parts separately the Short-time Fourier transform. Doing this method can give the time and frequency domains of each part. However, the method is very limiting in that cutting the signal will introduce convolution.

The main reason that the Fourier transform has these issues is that it is subjected to

the Heisenbergs uncertainty principle. The Heisenbergs uncertainty principle for signal processing shows that it is impossible to know both the exact frequency and the exact time of a frequency within a signal [7, 17]. The Wavelet Transform is a solution that overcomes the problems with the Fourier transform.

The wavelet transform (or wavelet analysis) uses a scalable modulated window to cut the signal. The modulated window is repeatedly shifted along the signal and would be slightly shorter with every new cycle. The result of the wavelet transform gives a collection of time-frequency windows where each window will have different resolutions. The difference between the wavelet transform and the Fourier transform is that the wavelet transform is not subjected to the Heisenberg principle, where the wavelet transform can receive both time and scale of a signal [17].

The wavelet transform can be extended onto two-dimensional signals such as images. This is important for this thesis as we are using the wavelet transforms properties on 2D signals such as images to perform a wavelet transform on the breast cancer mammograms for feature extraction, where the features are fed into the support vector machine classifier. The wavelet transform in two dimension works in much the same way as in one dimension, where a 2D signal is transformed in multiple orientations.

The wavelet and wavelet transform is explored in chapter 3.

1.1.2 Scattering Wavelet Transform

Furthermore, the wavelet transform can be extended to form the wavelet scattering network, where the wavelet transform can compute a translation invariant representation of a one or two-dimensional signal to be used for classification tasks.

This wavelet scattering network is a method that computes a translation invariant representation of an image by computing a cascade of convolution network using the wavelet transform. The advantageous properties of the wavelet transform are kept, but the scattering network also deals with a disadvantage of the wavelet transform - that it is covariant to translations.

The scattering convolution network has been used on texture images, facial recognition and also biomedical images where it has achieved highly accurate classification results. The mathematical and algorithmic properties of the scattering convolution network is explained in chapter 4.

1.1.3 Support Vector Machine

A Support Vector Machine (SVM) is a machine learning algorithm model used in classification. The key ability of the SVM is that is can learn from data to discern between a set of data and classify or predict any new unclassified data it reads. SVMs have an empirically good performance and has been successfully used in many fields [3].

An SVM is a binary classifier where labelled data is fed into the SVM. The data fed into the SVM is used to train the SVM to discern the differences between the different sets of data. The SVM then forms a hyperplane between the different sets of data. This

hyperplane can be viewed as a plane that has the largest margin between the sets of data, and if the data trained into the SVM causes the line to be non-linear, some kernel-tricks that can be applied to the SVM to allow the hyperplane to work in a higher dimension to allow non-linear classification [6].

1.2 Project Overview

This section is a brief overview of the thesis project. This project is supervised by Dr Yinan Kong and co-supervised by Abdullah-Al Nihad from the Department of Engineering, Macquarie University. Regular meetings with Abdullah-Al Nihad are arranged weekly for progress and advice during the thesis project.

Chapter 2 is the literature review as well as the background information on the thesis project. These will include some theoretical background on the Wavelet Transform (WT) as well as Support Vector Machines (SVM).

Chapter 3 will be the process of on how to utilise the Wavelet transform onto the breast cancer images for feature extraction and how it will be utilised by the SVM.

Chapter 4 involves the use of the Scattering Transform (ST) on the breast cancer images and how the accuracy performance compares to the regular Wavelet transform when using SVMs for classification purposes.

Chapter 5 will discuss the results from the SVM of both the wavelet transform feature extraction and the scattering transform.

Project Objectives

1. Familiarise with WT, ST and SVM theories
2. Familiarise with MatLAB programming environment
3. Program/modify existing WT code for the breast cancer images
4. Program/modify existing SVM code for the breast cancer images
5. Program/modify existing ST code for the breast cancer images
6. Analyse and compare the results of WT and ST using SVMs

1.2.1 Breast Cancer Images

The breast cancer images are provided by Abdullah-Al Nihad and Dr, Yinan Kong. There is a total number of 9527 images for this project to work with. It consists of the set of 3291 benign, 3520 malignant and 2716 normal images. The entire image set will be used to be analysed for the WT and SVM classification.

Estimated work	98 days
Realised work	81 days
Completion	83%

Table 1.1: Time budget review summary

1.2.2 Baseline Review

The thesis project ran from the 1st August to 7th November 2016. The plan for this thesis project is to work throughout all days in the week during the allocated time including weekends and the mid-semester break.

Time Budget Review

Table 1.1 summarises the number of days spent on the project. It also indicates the percentage of the project has been completed so far. At the moment, the allocated time for each project activities has been followed closely. Overall the progress of the project is in alignment with the project schedule planned at the beginning of the semester. Hence, there is no amendment to the project at this stage.

Financial Budget Review

The project came with $300 for the financial budget. The project only requires the software MatLAB, with a free student license obtained through Macquarie University. MatLAB has been installed on computers to do the research thesis. There is no need for any additional purchases.

Chapter 2

Background and Related Work

2.1 Wavelet Transform

Wavelet transform has been used in signal processing since the 1980 and has been used by engineers for a variety of applications. The wavelet transform is used because it has an advantage over the Fourier transform where it can receive a signal's frequency and time scales whereas the Fourier transform cannot get both at the same time due to the Heisenberg uncertainty principle.

The wavelet transform has been used for signal and image compression, pattern recognition, noise removal and texture analysis. This thesis will use the wavelet transform for feature extraction on breast cancer images.

2.2 Wavelet Scattering Network

The Wavelet Scattering Network (WST) is a process that computes a translated invariant image representation which is stable to deformations and preserves high-frequency information for classification. The WST uses Scattering Wavelets (SW) to produce a Scattering Convolution Network (SCN) where the properties of the SCN enables state-of-the-art classification results on handwritten digits and texture discrimination using a Gaussian kernel SVM and a Generative PCA classifier [5].

For image classification, a major difficulty is the many variations of an image within an image class. The amount of variation that can be in an image class means that there needs to be a translation invariant representation of the image that is stable to deformations. Small deformations need to be handled effectively with linear classifiers, which means that the representation is Lipschitz continuous to deformations.

The Fourier Transform can be used to construct a translation invariant representation, but it is not stable to deformations. The Wavelet Transform is stable to deformations, but it is covariant to translation. Building upon the WT by introducing non-linear operators leads to a convolution network that will result in a translation invariant representation that is stable to deformations.

The scattering transform computes a translation invariant representation by cascading wavelet transforms and modulus pooling operations, which average the amplitude of iterated wavelet coefficients. The scattering network is Lipschitz continuous to deformations while preserving the signal energy [5]. The scattering representation is able to give information on higher orders moments when compared to the Fourier power spectrum, thereby it is able to discriminate non-Gaussian textures that have the same power spectrum. This makes the scattering network effective for textures discrimination when using Gaussian kernel SVM and affine space models.

2.3 Support Vector Machines

The SVM is a machine learning model where it uses algorithms to classify and analyse data. SVM has the properties of a highly accurate machine learning model, and the ability to compute with high-dimensional data with the flexibility to be moulded with diverse data sources [4]. SVMs are a supervised learning algorithm, which requires some input from the user for the SVM to begin to learn. An SVM learns to classify data by first being fed with initial training data.

As SVMs are a binary classifier, the data used to train the SVM must be labelled. Using this training data, the SVM calculates the best way to discern between the two sets of data by creating a hyperplane. The hyperplane is a line that separates the two sets of data points with the largest possible in between them. It does this by first determining the support vectors within the data pool. Support vectors are the data points that are closest to their opposite sides, and the SVM will use these support vectors to calculate the best possible hyperplane.

Suppose that the training data fed into the SVM is given in the set $\{x_1, x_n\}$ for some space $x \subseteq \mathbb{R}^d$ and their labels $\{y_1, y_n\}$ where $y_i \subseteq \{-1, 1\}$. To calculate the hyperplane requires support vectors whose points are such that

$$\mathbf{x}_i \cdot \mathbf{w} + b \geqslant 1, y_i = 1 \tag{2.1}$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq 1, y_i = -1 \tag{2.2}$$

or compacted into

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geqslant 1 \tag{2.3}$$

Where w is normal to the hyperplane. The points for the equation 2.1 hold that lies on the hyperplane $H_1 : x_i \cdot w + b = 1$ with normal w and perpendicular distance from the origin $|-1-b|/||w||$. Similarly, the points for equation 2.2 hold that lie on the hyperplane $H_2 : x_i \cdot w + b = -1$ with normal w and perpendicular distance from the origin $|-1-b|/||w||$. Hence $d_+ = d = 1/||w||$ and the margin is simple $2/||w||$ [6].

Note that that H_1 and H_2 are parallel and that no training points lie between them. We can then find the pairs of hyperplanes which give the maximum margin by minimizing $||w||^2$, subjected to the constraints 2.3 [1].

With SVM there are cases where a linear hyperplane cannot sufficiently separate the two sets of data as the data points are scattered in a way that the hyperplane causes data from both sides to lie on the same side of the hyperplane. In these cases, the SVM can work in a different space to allow a non-linear hyperplane to develop. This is done with a kernel-trick, used to map the data into a higher dimension. A function ϕ is mapped to the higher space, and there will be a function $\phi(x_i) \cdot \phi(x_j)$. But if there is a kernel function K such that $k(x_i, x_j) = \phi(x_i) \cdot \phi(X_j)$, then knowing K is irrelevant [1, 4].

2.4 Previous Research

With breast cancer research, there have been many papers published that have focused on the same topic as this thesis project. Many have used wavelets to perform feature extraction on breast cancer images and other types of medical conditions. These papers have also used or at least acknowledge SVM in use for classification.

Y. Ireaneus et al. [14] have published a paper where the use of the DWT is applied to a mammogram and an SVM used as the classifier. In their paper, their proposed method was to use a series of image enhancement techniques of a Gaussian smoothing filter to remove noise, top hat filtering to correct uneven illumination and a DWT. Feature extraction was done by having the DWT'd image to measure properties of the image and used as training data for the SVM. Furthermore, the SVM had the Radial Basis Function (RBF) kernel applied to map the training data in a higher dimension.

Perlin Gorgel et al. [13] has done a similar method to Y.Ireneus et al. where they used the FWT instead of the DWT to process the image and used SVM were used alongside the Fuzzy subtractive clustering as a method of classification.

Andy Tirtajaya et al. [16] also use the same approach of WT to extract features and SVM to classify the mammograms. For the WT, the DWT was not used. Instead, the Dual-Tree Complex Wavelet was utilised for the paper. This resulted in achieving an 88.64% overall accuracy. Jeyashree.K et al. [9] used the CWT for the WT. Issac Niwas S et al. [11] opted for the complex Biorthogonal wavelets but used the k-nn classifier for classification instead of SVM.

Jiuqing Wang et al. [18] uses the WT on ultrasound liver images. The paper uses the Wavelet Packet Transform (WPT) rather than the standard WT. The difference between WT and WPT is that the WPT decomposes the three detail coefficients on top of the approximation coefficients. The reasoning is that much of the information needed for feature extraction are located in the middle-high frequencies, which the WPT are well suited for. The SVM classifier is used for classification where it resulted in a correct ratio of 71.82-85.79% with the WPT, higher than the correct ration of 68.52-77.65% for the WT.

Henrik Nicolay Finsberg [8] worked with Windowed Scattering Transform (WST) on ultrasound images of finger-joints. The WST is done with the software ScatNet, where a scattering transform is applied to the ultrasound images to generate texture discrimination results, by computing invariants to translation, rotations, scaling and deformations.

The work by Joan Bruna and Stéphane Mallat in [5] shows that the wavelet scattering network is able to achieve state-of-the-art results on handwritten digits and texture discrimination. They are able to achieve a classification error rate of 4.7 to 0.43 percent error rate on the MINIST hand-written digits database and a 0.2 classification error rate with the CURet texture database.

Chapter 3

Wavelet Transform

This chapter concerns the wavelet theory where the wavelet transform is explored in one and two dimensions as well as the continuous and discrete wavelet transforms. It will later in the chapter focus on how the discrete wavelet transform in two dimensions have been utilised for feature extraction on the breast images and how the features extracted will be used by the SVM for classification purposes. The wavelet theory will also be used to be built upon for the scattering wavelets that will be expanded upon in chapter **??** where wavelets will be used to create a convolution network that of an image representation that is invariant to translation and stable to deformations on images.

3.1 The Continuous Wavelet Transform

The Continuous Wavelet Transform (CWT) is one way that wavelets are utilised in signal processing and will be used as to show the properties of wavelets when used with the CWT.

The CWT is written as:

$$\gamma(s,u) = \int f(t)\psi^*_{s,u}(x)dx, \tag{3.1}$$

where * is the complex conjugate. Equation 3.1 also shows that the function$f(t)$ is decomposed into a set of basis functions, or wavelets $\psi_{s,u}(x)$ where s is the scale factor and u is the translation factor after the wavelet transform.

Wavelets are generated from this basic wavelet called the mother wavelet where the generated wavelets are created by scaling and translation the mother wavelet. The mother wavelets are written as

$$\psi_{s,u}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right), (s,u) \in \mathbb{R}\times\mathbb{R}^+, \tag{3.2}$$

Again, in 3.2 s is the scale factor and u is the translation factor and the factor $\sqrt{s}$ is for energy normalization across the different scales [8, 17]. One thing to note is that the

wavelet function is not specified compared to the Fourier transform. Instead, the wavelet function only needs to conform to a set of general properties to be used for wavelet analysis.

3.2 Properties of Wavelets

The most important properties of wavelets are the admissibility and regularity conditions. The square integrable functions $\psi(t)$ satisfy the admissibility condition,

$$\int \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < +\infty, \tag{3.3}$$

where $\Psi(\omega)$ is the Fourier transform of $\psi(t)$, so the signal can be first analysed and reconstructed without loss of information. Admissibility condition means that $\psi(t)$ vanishes at zero frequency

$$|\Psi(\omega)|^2|_{\omega=0} = 0 \tag{3.4}$$

A zero at zero frequency means a wavelet is then described as a function where the average is zero in the time domain and oscillatory

$$\int_{\mathbb{R}^2} \psi(t) dx = 0. \tag{3.5}$$

Using 3.1, the wavelet transform of a one-dimensional function is two-dimensional, and the wavelet transform of a two-dimensional is four-dimensional. The time-bandwidth product of the wavelet transform is the square of the input signal which is not a desirable property.

Therefore some additional conditions are imposed on the wavelet functions to have the wavelet transform decrease quickly with a decreasing scale of s. These are regularity conditions which state that wavelets should have some smoothness and concentration in time and frequency domains, which can be explained using vanishing moments [17].

If the wavelet transform in 3.1 is expanded into the Taylor series at $t = 0$ until order n with $u = 0$,

$$\gamma(s, 0) = \sum_{p=0}^{n} f^p(0) \int \frac{t^p}{p!} \psi\left(\frac{t}{s}\right) dt + O(n+1), \tag{3.6}$$

moreover, moments in wavelets are defined by M_p,

$$M_p = \int t^p \psi(t) dt, \tag{3.7}$$

where f^p is the p^{th} derivative of f and $O(n+1)$ is the rest of the expansion, then you can rewrite into 3.8

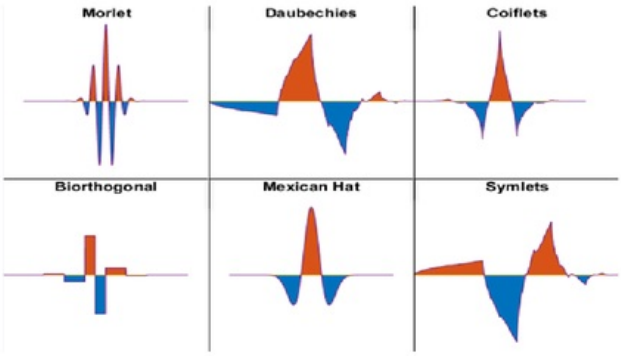

Figure 3.1: Examples of types of wavelets

$$\gamma(s,0) = \frac{1}{\sqrt{s}}\left[f(0)M_0 s + \frac{f^{(1)}(0)}{1!}M_1 s^2 + \frac{f^{(2)}(0)}{2!}M_2 s^3 + \cdots + \frac{f^{(n)}(0)}{n!}M_N s^{n+1} + Os\left(s^{n+2}\right)\right], \tag{3.8}$$

From the admissibility condition, the 0^{th} moment $M_0 = 0$, and make the other moments up to $M_n = 0$ as well, then the wavelet coefficients $\gamma(s,u)$ will decay as fast as s^{n+2} for a smooth signal $f(t)$. These are the vanishing moments.

3.2.1 Wavelet Families

There exists a possible infinite type of wavelets. Some well-known wavelets are the Haar wavelet which is the most simple wavelet being a square wave, the Daubechies family of wavelets, the Symlets family of wavelets - a modification of Daubechies, the Coiflets family, the BiorSplines family, Reverse BiorsSplines, Meyer, Gaussian, Mexican hat, Shannon and Fejer-Korovkin.

3.3 Discrete Wavelets

All of the above apply to the Continuous Wavelet Transform (CWT). There is also the Discrete Wavelet Transform (DWT). The DWT is a more practical use of the wavelet transform when compared to the CWT due to the excessive number of wavelets resulted through the CWT. In 3.1, the wavelet transform is calculated by shifting the scaled functions of over the signal continuously and calculating the correlation between the two. The CWT will not result in an orthogonal basis, and there will be wavelet coefficients that become redundant. [17]

Discrete Wavelets (DW) are used to transform a continuous signal into a useful series of wavelet coefficients. DW are not continuously scalable and translatable but are done in discrete steps. The DW is done with a modified mother wavelet

$$\psi_{j,k}(t) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{t - k2^j}{2^j}\right), \tag{3.9}$$

where j and k are integers. Now the time-scale space is sampled at discrete intervals. When discrete wavelets are used to transform a continuous signal, they will result in wavelet coefficients. Reconstruction of a signal is also possible, by having the wavelet coefficients lie between two positive bounds,

$$A||f||^2 \leq \sum_{j,k} |\langle f, \psi_{j,k}\rangle|^2 \leq B||f||^2, \tag{3.10}$$

where $||f||^2$ is the energy of $f(t), A > 0, b < \text{inf}$ and A, B are independent of $f(t)$.

DW can be made orthogonal to their dilations and translations by their mother wavelet,

$$\int \psi_{j,k}\psi^*_{m,n}(t)dt = \begin{cases} 1 & \text{if j = m and k =n} \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

Any signal can be reconstructed by summing the orthogonal wavelet basis function, weighted by the wavelet transform coefficients

$$f(t) = \sum_{j,k} \gamma(j,k)\psi_{j,k}(t), \tag{3.12}$$

however, the disadvantage of the discrete wavelets then is the resulting wavelet transform is no longer shift invariant.

To further limit the number of scaling and translations, DW takes advantage of the band-pass filter. In 3.4, wavelets have a band-pass like spectrum, so when a time compression of the wavelet is done by a factor of 2, it will also stretch the frequency of the wavelet by 2 and shift all frequency components by two as well. Doing this will cover the limited spectrum of a signal with the spectra of dilating wavelets.

Next, DW takes advantage of the scaling function. This is to prevent covering the spectrum all the way down to zero, for doing so will result in an infinite number of wavelets. A scaling function will limit the coverage of the spectrum when the wavelet spectra are small enough. The scaling function can be expressed as:

$$\varphi(t) = \sum_{j,k} \gamma(j,k)\phi_{j,k}(t) \tag{3.13}$$

The scaling function will cover all of the wavelets up the scale j, while the rest of the spectrum will be handled by wavelets. This sets a lower bound to wavelet, but doing so will result in lost information. The width of the scaling function would be the parameter of how many wavelets coefficients wanted [17].

Finally, DW uses subband coding to calculate the wavelet transform. Represent the wavelet transform as a filter bank, then a signal that goes through the wavelet transform

will be filtered. The output of the wavelet transform will result in the wavelet and scaling coefficients. Subband filtering can be done by building multiple band-pass filters to split the spectrum into frequency bands. The other method is to split the signal through a high-pass filter and a low-pass filter. The high-pass filter will contain the details of the signal that contain the wavelet coefficients, and the low-pass filter will contain the approximation of the signal. An approximation from the low-pass filter can be further filtered again to get further details [17].

3.4 The Discrete Wavelet Transform

With Discrete Wavelet Transform (DWT), band-pass filtering, scaling functions and sub-band coding are in effect when performing a DWT on a signal. When a wavelet spectrum is added to the scaling function spectrum, the resulting spectrum will be twice as wide as the first wavelet; the first scaling function can be expressed as the second scaling function. This the is the multi-resolution formulation where the scaling function can be expressed as the translated scaling functions at the next smaller scale:

$$\varphi(2^j t) = \sum_k h_{j+1}(k)\varphi(2^{j+1}t - k). \tag{3.14}$$

The first scaling function is replaced by a set of wavelets, which is expressed as translated scaling functions at the next scale. These sets of wavelets at level j can be written as:

$$\psi(2^j t) = \sum_k g_{j+1}(k)\varphi(2^{j+1}t - k). \tag{3.15}$$

A signal can be expressed in terms of dilated and translated wavelets up to a scale $j-1$, then a signal $f(t)$ can be expressed as:

$$f(t) = \sum_k \lambda_j(k)\varphi(2^j t - k) \tag{3.16}$$

or, more formally:

$$f(t) = \sum_k \lambda_j(k)\phi(2^j t - k) + \sum_k \gamma_j(k)\psi(2^{j-1}t - k). \tag{3.17}$$

If the scaling function and the wavelets are orthonormal, then the coefficients $\lambda_{j-1}(k)$, $\gamma_{j-1}(k)$ are found by taking the inner products

$$\lambda_{j-1}(k) = \langle f(t), \varphi_{j,k}(t)\rangle \tag{3.18}$$

$$\gamma_{j-1}(k) = \langle f(t), \psi_{j,k}(t)\rangle \tag{3.19}$$

Replacing the $\varphi_{j,k}(t), \psi_{j,k}(t)$ with 3.14 and 3.15 we get

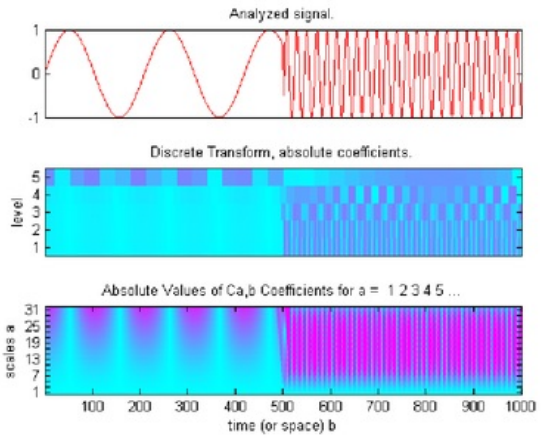

Figure 3.2: A one-dimensional signal undergoing discrete wavelet transformation for five levels

$$\lambda_{j-1}(k) = \sum_m h(m-2k)\lambda_j(m); \tag{3.20}$$

$$\gamma_{j-1}(k) = \sum_m g(m-2k)\gamma_j(m). \tag{3.21}$$

Wavelet and scaling coefficients on different scales can be calculated from the previous scale. Scaling coefficients come from the low-pass filter, and subband coding is done by splitting a low-pass spectrum through a low-pass and high-pass filter.

With the DWT then, a signal can be divided through subband coding with a high-pass filter containing the signal's λ_k coefficients and the low pass filter containing the γ_k coefficients. We call the high-pass filter $g(k)$ the wavelet filter, and the low-pass filter $h(k)$ the scaling filter.

As a signal is passed through these filter, $h(k)$ can be filtered again to receive to receive the $h(k)$ and $g(k)$ coefficients of the next level. This filtering can be done multiple times on subsequent $h(k)$ until to the desired level, or until one of the coefficients become smaller than the one of the filters. [17]

Figure 3.2 shows a signal that has undergone five levels of filtering using DWT.

3.5 The Discrete Wavelet Transform in Two-Dimensions

Wavelets in two dimensions are also defined as a function of a zero average.

$$\int_{\mathbb{R}^2} \psi(x)dx = 0. \tag{3.22}$$

In two-dimensions, the DWT is similar to DWT in one-dimension. There is still a wavelet and scale filter to receive the $g(k)$ and $h(k)$ coefficients except that it is now working in two dimensions:

$$\phi_{j,m,n}(x,y) = 2^{j/2}\phi(2^j x - m, 2^j y - n), \tag{3.23}$$

$$\psi^i_{j,m,n}(x,y) = 2^{j/2}\psi(2^j x - m, 2^j y - n), i = \{H,V,D\}. \tag{3.24}$$

The wavelet filter in two dimensions are actually three wavelet filters: $\psi^H_{j,m,n}(x,y)$, $\psi^V_{j,m,n}(x,y)$, $\psi^D_{j,m,n}(x,y)$. In two-dimensions, the scaling filter is still conceptually the same, being the result of the coefficients from the $h(k)$ filter, but the wavelet function is done in the horizontal, vertical and diagonal directions of the two-dimensional signal [10].

$$\phi(x,y) = \phi(x)\phi(y), \tag{3.25}$$

$$\psi^H(x,y) = \psi(x)\phi(y), \tag{3.26}$$

$$\psi^V(x,y) = \phi(x)\psi(y), \tag{3.27}$$

$$\psi^D(x,y) = \psi(x)\psi(y). \tag{3.28}$$

The two-dimensional signal can still be reconstructed by the wavelet and scaling coefficients:

$$W_\phi(j_0,m,n) = \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\phi_{j_0,m,n}(x,y), \tag{3.29}$$

$$W^i_\psi(j,m,n) = \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\phi^i_{j,m,n}(x,y), i = \{H,V,D\} \tag{3.30}$$

$$\begin{aligned} f(x,y) &= \frac{1}{\sqrt{MN}}\sum_m\sum_n W_\phi(j_0,m,n)\phi_{j_0,m,n}(x,y) \\ &+ \frac{1}{\sqrt{MN}}\sum_{i=H,V,D}\sum_{j=j_0}^{\infty}\sum_m\sum_n W^i_\psi(j,m,n)\phi^i_{j,m,n}(x,y) \end{aligned} \tag{3.31}$$

A DWT on a two-dimensional signal is typically done on images, where the wavelet coefficients consist of the horizontal, vertical and diagonal details of the image and the scaling coefficients will be the approximation of the image. [10]

Figure 3.3 shows an image that has undergone DWT in two dimensions in one level, showing it is approximation and detail coefficients. The top left image is the approximation coefficient $\phi(x,y)$ of the image, the top right is the horizontal coefficients $\psi^H(x,y)$, the bottom left is the vertical coefficients $\psi^V(x,y)$, and the bottom right is the diagonal coefficients $\psi^D(x,y)$.

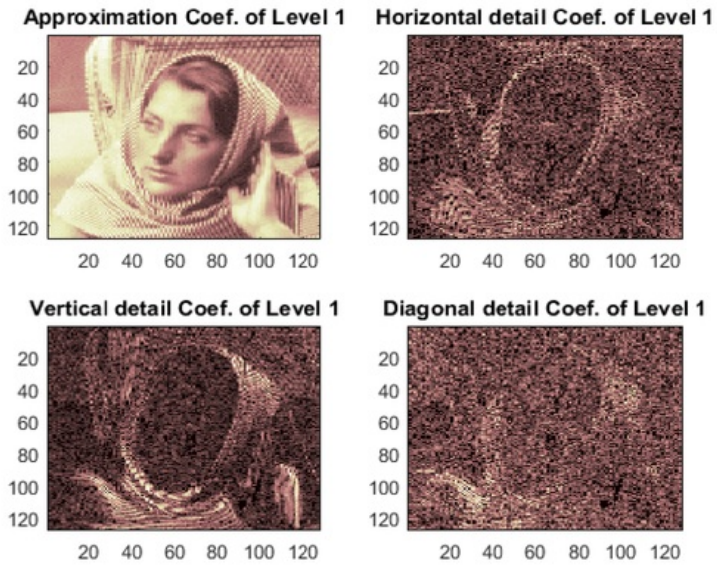

Figure 3.3: An image undergoing one level of discrete wavelet transformation

3.6 The Continuous Wavelet Transform in Two-Dimensions

While not directly used in this thesis, the theoretical analysis of the CWT in two dimensions is briefly examined here. The CWT in two-dimensions still use the wavelets in two dimensions where the function is a zero average. Contrast to the DWT in two dimensions, the CWT is done by dilating, rotating and translating the wavelet

$$\psi_{u,s,\theta}(x) = s^{-1}\psi\left(r_\theta^{-1}\left(\frac{x-u}{s}\right)\right), (s,u,r_\theta) \in \mathbb{R}_+ \times \mathbb{R}^2 \times SO(2), \tag{3.32}$$

with

$$SO(2) = \left\{ r_\theta = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta)) & cos(\theta)) \end{pmatrix} : \theta \in 0, 2\pi \right\} \tag{3.33}$$

The CWT at a position u, scale s and orientation θ are

$$Wf(u,s,\theta) = \langle f, \psi_{u,s,\theta} \rangle = \int_{\mathbb{R}^2} f(x) s^{-1} \psi^* \left(r_\theta^{-1}\left(\frac{x-u}{s}\right)\right) dx, \tag{3.34}$$

moreover, the admissibility condition is

$$C_\psi = \int_{\mathbb{R}^2} \frac{|\hat{\psi}(\omega)|^2}{|\omega|^2} d\omega < +\infty. \tag{3.35}$$

The admissibility conditions ensure that the signal can be reconstructed from the wavelet function, and the reconstruction formula is

$$f(x) = \frac{1}{C_\psi} \int_{\mathbb{R}_+ \times \mathbb{R}^2 \times SO(2)} \psi_{u,s,\theta}(x) W f(u,s,\theta) du \frac{ds}{s^3} d\theta \tag{3.36}$$

Combining the reconstruction formula to the wavelet transform 3.34 will give the reproduction property

$$W f(u_0, s_0, \theta_0) = \int_{\mathbb{R}_+ \times \mathbb{R}^2 \times SO(2)} K(u, u_0, s, s_0.\theta, \theta_0)) W f(u,s,\theta) du \frac{ds}{s^3} d\theta. \tag{3.37}$$

Kernel $K(u, u_0, s, s_0, \theta, \theta_0) = c_\psi^{-1} \langle \psi_{u,s,theta}, \psi_{u_0,s_0.\theta_0} \rangle$ is the reproducing kernel which means that the CWT is redundant representation of the signal. If given the points us, θ, then the wavelet transform in the neighbouring points u_0, s_0, θ_0 can be found with $K(u, u_0, s, s_0, \theta, \theta_0) \neq 0$. Restricting the wavelet transform to a subset of discrete points is possible without loss of information [8]. The CWT is important for the wavelet scattering network that will be discussed in section 2.2. [8]

3.7 Feature Extraction

Image processing uses feature extraction to describe an image's information used in many different applications such as image classification. It aims to obtain the most relevant information of an image and present the information in a suitable space in a machine-readable format. The choice of features to extract from an image is to be of interest to the task and would be able to distinguish between different classes. Ideally, the feature space should be enough for to maximise classification recognition rate with the lease amount of features in the feature space. In some cases, having an extended feature vector can least to worse classification performance rather than increase it.

It needs to be small enough to discriminate between classes efficiently, but produce similar feature vectors for each class. Features are either local features or global features. Local features usually describe geometric while global features are generally topological. There are three major methods for feature extraction statistical features, global transformation and series expansion features, and geometric and topological features.

The statistical features are derived from the statistical distribution of points. Global transformation and series expansion features are representing a signal in a series of functions where the coefficient of a linear combination provides compact encoding. Geometrical and topological features are the global and local properties of characters.

Feature extraction for this project was done in two methods. The first method was done by applying the wavelet transform to the images and using the approximation of the image to extract the features of the image. The approximation coefficient contains most of the information of the original image which is why it is used for the feature extraction. The wavelet transform is done up to three levels, and the features of the resulting three

levels of approximation are extracted. The features that will be extracted are the mean, median, standard deviation, peak-signal-to-noise ratio, signal-to-noise ratio, entropy and skewness of the image approximation.

The second method was to apply the wavelet transform to the images and using the detail coefficients to obtain the mean energy of each orientation. The detail coefficients contain information of an images energy in the horizontal, vertical and diagonal details. Images of the same class would have similar mean energy in the three directions of wavelet coefficients. This is done on three levels of wavelet coefficients where the feature vector contains the average energy of the first level of coefficients, then the second level and finally the third level.

This is done using the wavelet MatLAB toolbox, which performs the wavelet transform onto the images with a few lines of code. The wavelet toolbox on performs multilevel wavelet decomposition on a two-dimensional signal using the wavedec2 function.

The wavedec2 function applies a discrete wavelet transform onto a two-dimensional signal (in this case an image) and stores the wavelet coefficients into a row vector C and a bookkeeping matrix S. This row vector C stores the approximation, horizontal, vertical and diagonal coefficient of the image transform up to N levels. It stores the coefficients as:

$$A(N), H(N), V(N), D(N), H(N-1), V(N-1), D(N-1), \ldots, H(1), V(1)D(1) \quad (3.38)$$

where A, H, V and D are row vectors where A contains the approximation coefficients, H the horizontal detail coefficients, V the vertical detail coefficients and D the diagonal detail coefficients. The bookkeeping matrix S details the size the approximation coefficients and all the detail coefficients to know how the two-dimensional coefficient sizes are.

The function wavedec2 syntax is

```
[C,S] = wavedec2(X,N, 'wname')
```

C and S are already explained previously. X is the two-dimensional signal to undergo wavelet transform; N is the level to transform the two-dimensional signal, and 'wname' is the wavelet that will be used to transform the image. The wavelets that can be used for the function are orthogonal or biorthogonal wavelets. This means that the function can only accept the Daubechies, Coiflets, Symlets, Fejer-Korovkin, discrete Myer, Biorthogonal and reverse biorthogonal wavelets.

The function wavedec2 can alternatively have the syntax

```
[C,S] = wavedec2(X,N,Lo_D,Hi_D)
```

where 'wname' is replaced with Lo_D and Hi_D. These are low-pass and high-pass decomposition filters if the user chooses to use their own wavelet filters. This is because as mentioned before, there are potentially an infinite number of wavelets, and the user can specify their own wavelet for their needs. This thesis will only use the included wavelets in the wavelet toolbox.

Once the image has undergone the wavelet transformation, the next step is to extract the coefficients from the vector C. For the approximation coefficients; the function appcoef2 extracts the approximation coefficients from C. Detail coefficients are extracted using the detcoef2 function. The syntax of the appcoef2 function is:

```
A = appcoef2(C,S,'wname',N).
```

This function computes the approximation coefficients at level N, from the wavelet decomposition structure [C,S] using the wavelet at 'wname'. A is the resulting approximation coefficient of a signal X with the size determined from S.

The function for obtaining the detail coefficients uses the syntax:

```
[H, V, D] = detcoef2('all',C,S,lvl);
```

where H, V, D are the horizontal, vertical and diagonal coefficients respectfully.

Feature extraction will be applied to these approximation coefficients, using the mean2, median, std2, pnsr, entropy and skewness functions to gather the features. These will be explained in the next section. These features are stored into a numeric matrix to be used as training data to feed into the SVM or to be used as testing data to test the SVM's accuracy.

Feature extraction for the detail coefficients is done using the mean2 function on each orientation and for three levels. These would be used to train and test a separate SVM with the results compared to the other methods.

3.8 Method for Wavelet Transform Feature Extraction

In this thesis project, MatLAB is chosen to program the wavelet transform and the SVM. The process is to have the mammograms be cropped so only the breast is shown. These images would be then be transformed a DWT and then undergo feature extraction from the approximation of each image. The wavelet transform allows the image to be denoised, and provide the approximation of the image to get the intensity of the images for feature extraction.

The SVM would be fed with the training data where the benign, cancerous and normal mammograms will be labelled as three separate labels to compute the SVM classifier between three sets of data. SVMs is a binary classifier, so the images are grouped into three groups of images for binary classification. This will be expanded on section 3.9. Figure 3.4 is the overall method of the thesis project

3.9 MatLAB Method

The images originally include a lot of blank spaces and also medical labels, so the images are first cropped from the original image, to only feature the breast as best as possible.

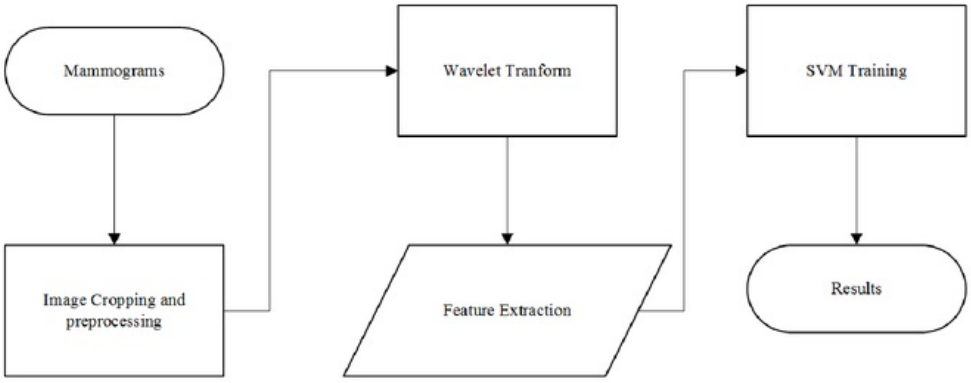

Figure 3.4: Flow diagram of method

Algorithm 1 Image cropping

```
for all Images do
    Crop image
    Save image
end for
```

The resulting cropped images will then undergo the wavelet transform as was explained above, where the approximation coefficients of the images would go through feature extraction. The wavelet transform will go for three levels and will use eight different mother wavelets for the wavelet decomposition. These mother wavelets are the haar, db2, db3, db6, db8, db9, db10 and bior3.7 wavelets which will result in eight matrices.

Feature extraction will be done on the three levels of the approximation coefficients on the image and be saved into a matrix, with each row of the matrix being an image and each column of the matrix be the feature that is extracted from the image. At the end of each row, is a label of each image, where 1 is a benign, 2 cancerous, or 3 normal images this results in eight matrices of features resulted from each mother wavelet.

Algorithm 2 Image cropping

```
for all Images do
    Wavelet decomposition on image
    Extract approximation coefficients on image
    Extract features from approximation coefficients
    Save extracted features into numeric matrix
    Save label for image at the end of each row
end for
```

A total number of 21 features that are extracted from each image. These are the mean, median, standard deviation, peak-signal-to-noise ratio, signal-to-noise ratio, entropy and

skewness of each level of approximation. This is saved as a $M \times 22$ matrix with M being the total number of images, with 21 features. The 22nd column is the label of the image.

For features from the detail, coefficients are the mean of each orientation and level of detail coefficients. These features are saved as a $M \times 10$ matrix, with M be a number of images and the 10th column being the label of the image.

The features extraction matrix is divided into three separate matrices: benign-cancerous, benign-normal, and cancerous-normal. These three groups are labelled as BC, BN and CN respectfully. The resulting data is three arrays of data - (Size of BC$\times$22), (Size of BN$\times$22) and (Size of CN $\times$ 22). The three groups are individually fed to the SVM for training and testing.

Training the SVM for one group is done by first shuffling the data by rows and proportion the data for training and testing purposes.

Algorithm 3 Data shuffling

```
Get data input
Prepare random permutation
Shuffle feature extraction data
Separate feature extraction and labels
Save shuffled data
```

Training the SVM is done by setting a proportion of the data to be used as training data and the rest to be testing data. The training data that is selected is fed into the SVM with their appropriate labels. The SVM will use the Gaussian kernel function for non-linear classification.

Cross-validation is done by using the predict function on the training data and compare the predicted labels with the correct expected labels. The cross validation error is

$$\text{Error} = \frac{\text{No. of incorrect labels}}{\text{Total No. of images}} \times 100. \tag{3.39}$$

Including the error of the cross-validation of the SVM classifier, the confusion plotting of the generated for cross-validation. Testing the SVM classifier uses the testing data selected when setting the proportion of the original data. This data is using the predict function to get the predicted labels from the data and is compared to the expected labels. The error and confusion plot is also generated the same way as the cross-validation of the SVM classifier.

The full code of the wavelet transform, shuffling data, training and testing of SVM will be in the appendix D.

3.9.1 Support Vector Machine

The SVM that will be used is the built-in statistics and Machine Learning toolbox in MatLAB. The SVM classifier in MatLAB uses the function

```
1 Mdl = fitcsvm(X,Y,Name,Value).
```

Algorithm 4 SVM training and testing

```
Load shuffled data
Set proportion of data to be training and testing data
for Training data do
    Separate data and labels
    Train SVM with training data
    Predict label from training data
    for all Predicted labels do
        if Predicted label ≠ expected label then
            No. of incorrect label+1
        end if
    end for
    Display error rate for cross validation
    Display confusion matrix
end for
for Testing data do
    Separate data and labels
    Predict label from testing data
    for all Predicted labels do
        if Predicted label ≠ expected label then
            No. of incorrect label+1
        end if
    end for
    Display error rate for classification
    Display confusion matrix
end for
```

Mdl is the trained classifier, X is a numeric matrix containing the data to be used for training the classifier, Y is the class levels to be used for one or two class classification.The Name and Value pair arguments are used to specify SVM options to be used for the classifier, such as choosing what kernel function to use. Prediction of data is done by using the function

```
label = predict(SVMModel,X)
```

with the label being the predicted label of the data in the matrix X using the trained SVM classification model SVMModel.

Full code of training and predicting using SVM is in appendix D.

Chapter 4

Invariant Scattering Convolution Networks

The scattering convolution networks is a method for computing an image representation that is translation invariant and stable to deformations used for classification purposes. It has achieved state-of-the-art results on handwritten digits and texture analysis using a Gaussian kernel SVM and a generative PCA classifier. This chapter will detail how the scattering convolution network is formed and to see how accurate the results can be achieved using the network on the breast images. It will detail the theory behind scattering, the properties, and the methods utilised for the classification of the breast images. The following is based on [5], where the scattering convolution method will be utilised for the breast cancer images.

4.1 Fourier modulus

A signal x and a representation Φx is invariant to global translations $x_c(u) = x(u-c)$ by $(c_1, c_2) \in \mathbb{R}^2$ if $\Phi x_c = \Phi x$. The representation $\Phi x(u) = x(u - a(x))$ is canonical invariant when it registers x with and anchor point $a(x)$, which is translated when x is translated: $a(x_c) = a(x) + c$. The anchor point may be a filtered maximum $a(x) = argmax_u |x \star h(u)|$ for some filter $h(u)$.

The Fourier transform modulus provides a translation invariant representation. If $\hat{x}(\omega)$ is the Fourier transform of a signal $x(u)$ and $\hat{x}(\omega) = e^{ic.\omega}\hat{x}(\omega)$ then the results is that $|\hat{x}_c| = |\hat{x}|$ does not depend on c. Autocorrelation $Rx(v) = \int x(u)x(u-v)du$ is translation invariant: $Rx = Rx_c$ [5].

For a representation to be stable to additive noise $x'(u) = x(u) + \epsilon(u)$ then a Lipschitz continuity condition which supposes that there exists $C > 0$ such that for all x and x':

$$||\Phi x' - \Phi x|| \leq C||x' - x||, \tag{4.1}$$

where $||x||^2 = \int |x(u)|^2 du$. The Plancherel formula $\int_{-\infty}^{\infty} f(x)g^*(x)dx = \frac{1}{2\pi}F(k)G^*(k)dk$ means that the Fourier modulus $\Phi x = |\hat{x}|$ satisfy the property $C = 2\pi$.

Stability to deformations requires Φ to be Lipschitz continuous to deformations. Small deformations are written as $x_\tau(u) = x(u - \tau(u))$ where $\tau(u)$ is a non-constant displacement field that deforms an image. $\nabla\tau(u)$ is a matrix that measures the deformation amplitude at u and $\sup_u|\nabla\tau(u)|$ is the global deformation amplitude. Small deformations is when $\nabla\tau(u) < 1$. If there exists $C > 0$ such that for all τ and x: $||\Phi x_\tau - \Phi x|| \leq C||x||sup_u|\nabla\tau(u)|$, then it is Lipschitz continuity relative to deformations.

With Φ being Lipschitz continuous to deformations τ then the map that transforms τ into Φx_τ is almost everywhere differentiable. For small deformations, $\Phi x - \Phi x_\tau$ is closely approximated by a boundary linear operator of τ. Deformations are linearised by Φ, handling deformations variabilities for linear classifiers.

The Fourier Transform is unstable to the small deformations at high frequencies, where $||\hat{x}(\omega)| - |\hat{x}_t au(\omega)||$ can be large at a high frequency ω. The Fourier modulus also loses too much information when two signals such as a Dirac $\delta(u)$ and a linear chirp e^{iu^2} have the same moduli and constant; the Fourier modulus may not be able to discriminate between the signals when classifying.

So while the Fourier transform of a signal $x(u)$ is translation invariant, it is unstable to deformations. Also, if $x' = x_\tau$, the instability at high frequencies ω implies that $\Phi(x(u) = x(u - a(x))$ is also unstable with respect to deformations [5].

4.2 Scattering Wavelets

The wavelet transform is stable to deformations. However, it is translation covariant, rather that translation invariant. The scattering transform builds non-linear invariants from the wavelet coefficients using modulus and averaging pooling functions.

The scattering wavelet for images uses the two-dimensional wavelets, done by rotating a band-pass filter ψ and dilating the wavelet by 2^j. The wavelet is rotated by $r \in G$ where G is a group of rotations r of angles $2k\pi/K$ where $0 \leq k < K$. This scattering wavelet is written as:

$$\psi_\lambda(u) = 2^{-2j}\psi(2^{-j}r^{-1}u), \lambda = 2^{-j}r, \tag{4.2}$$

and Fourier transform of dilated and rotated wavelet is

$$\hat{\psi}_{2^{-j}r}(\omega) = \hat{\psi}(2^j r^{-1}\omega), \tag{4.3}$$

whom has a support centred at $2^{-j}r\eta$ and a bandwidth proportional to 2^{-j}. The index $\lambda 2^{-j}r$ gives the frequency location of ψ_λ where its amplitude is $|\lambda| = 2^{-j}$. The wavelet transform of x is $\{x \star \psi_\lambda(u)\}_\lambda$, which is a redundant transform with no orthogonality property [5]. On discrete images, we only capture frequencies in the circle $|\omega| \leq \pi$ in the image frequencies square.

The scattering transform can be used with general wavelets, however, complex wavelets that have the form:

$$\psi(x) = e^{i\{\eta,x\}}\theta(x), \tag{4.4}$$

are the Gabor family of wavelets and are desired for the scattering transform. The Morlet wavelet is one example of the complex family of wavelets:

$$\psi(u) = \alpha(e^{iu.\xi} - \beta)e^{-|u|^2/(2\delta^2)} \tag{4.5}$$

Since wavelet transforms are computed with translation, it cannot be translation invariant. For a translation invariant image using the wavelet, it is necessary to introduce non-linearity. If Q is a linear or non-linear operator that computes translations, then $\int Qx(u)du$ is translation invariant on a signal x. $Qx = x \star \psi_\lambda$ gives a trivial invariant $\int x \star \psi_\lambda(u)du = 0$ since $\int \psi_\lambda(u)du = 0$. If $Qx = M(x \star \psi_\lambda)$ where M is linear and commutes with translations, the integral still vanishes, meaning that computing invariants require a non-linear pooling operator M.

For the pooling operator M to allow stability to deformations, M needs to compute with the action of any diffeomorphism, to preserve stability to additive noise and to be non-expanding: $||My - Mz|| \leq ||y - z||$. A M that is non-expanding and is able to compute the action of diffeomorphism can prove that M is necessarily a point-wise operator - where $My(u)$ is a function of the value $y(u)$. An invariant representation that preserves signal energy, means that a modulus operator over complex signals $y = y_r + iy_i$:

$$My(u) = |y(u)| = (|y_r(u)|^2 + |y_i(u)|^2)^{\frac{1}{2}}. \tag{4.6}$$

It results in translation invariant coefficients $\mathbf{L}^1(\mathbb{R}^2)$ norms:

$$||x \star \phi_\lambda||_1 = \int |x \star \phi_\lambda(u)|du. \tag{4.7}$$

The $\mathbf{L}(\mathbb{R})^2$ norms $\{||x \star \phi_\lambda||_1\}$ form a signal representation that measures the sparsity of wavelet coefficients. There is a loss of information, but it does not come when modulus that removes the complex phase of $x \star \phi_\lambda(u)$, loss of information comes from the integration of $|x \star \phi_\lambda(u)|$, where it removes non-zero frequencies.

The recovery of the lost information happens when calculating the wavelet coefficients $\{|x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}(u)\}_{\lambda_2}$ of $|x \star \lambda_{\phi_1}|$. This results in a larger family of invariants for λ_1 and λ_2:

$$|||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}||_1 = \int ||x \star \psi_{\lambda_1}(u)| \star \psi_{\lambda_2}|du. \tag{4.8}$$

Further translation invariant coefficients are computed by further iterating on the wavelet transform and modulus operators. Letting $U[\lambda]x = |x \star \psi_\lambda|$, any sequence $p = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ is a path which computes an ordered product of non-linear and non-commuting operators:

$$U[p]x = U[\lambda_m] \ldots U[\lambda_2]x = |||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \ldots | \star \psi_{\lambda_m}|, \tag{4.9}$$

with $U[\emptyset]x = x$ - the original signal. The scattering transform on the path p is defined as an integral, normalised by the response of a Dirac:

$$\overline{S}x(p) = \mu_p^{-1} \int U[p]x(u)du \tag{4.10}$$

with $\mu_p = \int U[p]\delta(u)du$.

Every scattering coefficients $\overline{S}x(p)$ are invariant to a translation of x, which have similarities to the Fourier transform modulus, but is also Lipschitz continuous to deformations in contrast to the Fourier transform.

In classification tasks, it is better for the localised descriptors - that are invariant to translations, to be smaller than a predefined scale 2^J, keeping the spatial variability at scales larger than 2^J. This is obtained by a localisation of the scattering integral with a scaled spatial window $\phi_{2^J}(u) = 2^{-2J}\phi(2^{-J}u)$ where it defines a windowed scattering transform in the neighbourhood of u:

$$S[p]x(u) = U[p]x \star \phi_{2^J}(u) = \int U[p]x(v)\phi_{2^J}(u-v)dv, \tag{4.11}$$

hence

$$S[p]x(u) = |||x \star \psi_{\lambda_1}| \star_{\lambda_2} \ldots| \star \psi_{\lambda_m}| \star \phi_{2^J}(u), \tag{4.12}$$

with $S[\emptyset]x = x \star \phi_{2^J}$. Each path p, $S[p]x(u)$ is a function of the window position u, which can be sub sampled at intervals proportional to the window size 2^J. The averaging of ϕ_{2^J} means that if $x_c(u) = x(u-c)$ with $|c| \ll 2^J$, then the windowed scattering is nearly translation invariant: $S[p] \approx S[p]x_c$.

4.3 Scattering Convolution Network

$p = (\lambda_1, \ldots, \lambda_m)$ is a path of length m, the windowed scattering transform coefficient will be $S[p]x(u)$ and has order of m. The windowed scattering transform is computed at the layer m of the convolution network. Several layers are necessary to avoid losing information. The first-order coefficients $S[\lambda_1]x$ are the same as Scale-Invariant Feature Transform (SIFT) coefficients. This computes the local sum of image gradient amplitudes among image gradients having nearly the same direction in a histogram having eight different direction bins. These coefficients are approximated by $S[\lambda_1]x = |x \star \psi_{\lambda_1}| \star \phi_{2^J}(u)$, where $psi_{lambda_1}]$ are partial derivatives of a Gaussian computed at the finest image scale, along eight different rotations. The averaging filter ϕ_{2^J} is a scaled Gaussian.

Detecting edges and sharp transitions are done with partial derivative wavelets. Theses are suited when the edges and sharp transitions do not have enough frequency and directional resolution to discriminate complex directional structures. The average wavelet coefficient amplitudes $|x \star \psi_\lambda \star \phi_{2^J}(u)$ are used for texture analysis, and with a complex wavelet ψ, it produces a better frequency and directional resolution.

Scattering transform computes higher order coefficients by iterating on wavelet transforms and modulus operators. It is computed up to a maximum scale 2^J where lower frequencies are filtered by $\phi_{2^J}(u) = 2^{-2^J}\phi(2^{-J}u)$. The Morlet wavelet's ψ averaging filter

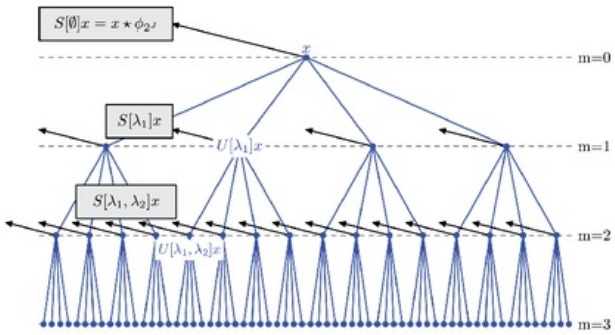

Figure 4.1: A scattering convolution network

ϕ is chosen to be Gaussian. Since images are real-valued signals, then we only need to consider positive rotations $r \in G^+$ with angles in $[0, \pi)$:

$$Wx(u) = \{x \star \phi_{2^J}(u), x \star \psi_\lambda(u)\}_{\lambda \in \mathcal{P}}, \tag{4.13}$$

with an index set $\mathcal{P} = \{\lambda = 2^{-j}r : r \in G^+, j \leq J\}$.

The wavelet modulus propagator keeps the low-frequency averaging and computes the modulus of complex wavelet coefficients:

$$\tilde{W}x(u) = \{x \star \phi_{2^J}(u), |x \star \psi_\lambda(u)|\}_{\lambda \in \mathcal{P}}. \tag{4.14}$$

The figure 4.1 taken from [5], details the scattering convolution network is done. Network nodes of layer m corresponds to the set $\mathcal{P}^m$ of all paths $p = (\lambda_1, \ldots, \lambda_m)$ of length m. The mth layer consists of the propagated signals $\{U[p]x\}_{p \in \mathcal{P}^m}$ and the scattering coefficients $\{S[p]x\}_{p \in \mathcal{P}^m}$. Any $p = (\lambda_1, \ldots, \lambda_m)$, we then denote $p + \lambda = (\lambda_1, \ldots, \lambda_m, \lambda)$. With $S[p]x = U[p]x \star \phi_{2^J}$ and $U[p + \lambda]x = |U[p]x \star \psi_\lambda|$, then

$$\tilde{W}U[p]x = \{S[p]x, U[p + \lambda]x\}_{\lambda \in \mathcal{P}}. \tag{4.15}$$

The scattering propagator $\tilde{W}$ applied to all propagated signals $U[p]x$ of the mth layer $\mathcal{P}^m$ outputs the scattering signals $S[p]x$ and the propagated signals $U[p + \lambda]$ of the next later $\mathcal{P}^{m+1}$. The output scattering signals are calculated by first calculating the $\tilde{W}x = \{S[\emptyset]x, U[\lambda]x\}_{\lambda \in \mathcal{P}}$ and afterwards applying the scattering propagator on every layer.

The figure 4.1 shows the scattering convolution network where a scattering propagator $\tilde{W}$ is applied to an image x is computed with the first layer of wavelet coefficient modulus $U[\lambda_1]x = |x \star \psi_{\lambda_1}|$ and the local average $S[\emptyset]x = x \star \phi_{2^J}$. The first-order scattering coefficients $S[\lambda_1] = U[\lambda_1] \star \phi_{2^J}$ is computed by the scattering propagator $\tilde{W}$ being applied on the first later $U[\lambda_1]$ which also results with the propagated signal $U[\lambda_1, \lambda_2]x$ of the second layer. The scattering propagator $\tilde{W}$ being applied to signal $U[p]x$ outputs the next-order scattering coefficients $S[p]x = U[p]x \star \phi_{2^J}$ and the next layer of propagated signals.

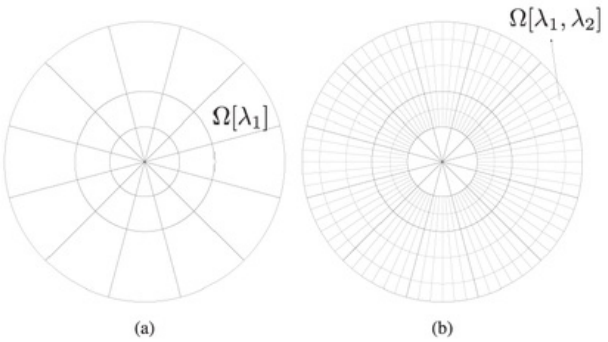

Figure 4.2: The disk showing the Fourier support of an image x.

The scattering signals $S[p]x$ is translation invariant by the averaging of $U[p]x$ by ϕ_{2^J}. Loss of information is prevented from recovering the wavelet coefficients at the next layer which is the reason for creating a convolution network. The difference between the scattering coefficients and standard convolutions is that the scattering coefficients are produced at every layer rather than the last layer. Filters are predefined wavelets rather than being learned from the data. It builds invariance relative to the action of the translation group using these predefined wavelets which perform convolutions along rotation or scale variables.

Separating signal variations at different scales is fundamental for deformation stability. Removing high-frequency oscillations of the wavelet coefficients is done by the modulus operator where it pulls together quadrature phase filters [5].

4.3.1 Visualising Scattering Convolution Network Example

The scattering convolution network is visualised with a disk to represent the Fourier support of an image x. On the disk for a fixed position u is the windowed scattering coefficients $S[p]x(u)$ of order $m = 1, 2$ displayed as piecewise constant images. The frequency disk is divided into $\{\Omega[p]\}_{p\in\mathcal{P}^m}$ sectors that are indexed by the path p. The figure 4.2 taken from [5] is the disk in question.

At $m = 1$, the scattering coefficients of $S[\lambda_1]x(u)$ depends on the local Fourier transform energy of x of the support of $\hat{\psi}_{\lambda_1}$. The value over the sector $\Omega[\lambda_]$ approximates the frequency support of $\hat{\psi}_{\lambda_1}$ at figure4.2(a). Second-order scattering coefficients $S[\lambda_1, \lambda_2]x(u)$ computed are displayed over frequency sector $\Omega[\lambda_1, \lambda_2]$ that has subdivided the sectors $\Omega[\lambda_1]$ of the first wavelets $\hat{\psi}_{\lambda_1}$ at figure 4.2(b).

The figure 4.3 from [5] shows the Fourier transform and the amplitude of the scattering coefficients. The top and bottom images have the same first-order scattering coefficients, but the second-order coefficients are different for the two images.

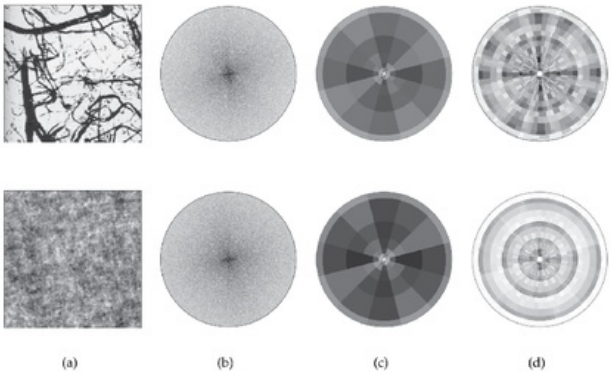

Figure 4.3: (a) Two images. (b) Fourier modulus. (c) First-order scattering coefficients. (d) Second-order scattering coefficients.

4.4 Energy Propagation and Deformation Stability Properties

The windowed scattering S is computed with a cascade of wavelet modulus operators $\tilde{W}$, whose properties depend on the wavelet transform properties. The wavelets that are to be used on the scattering transform have certain conditions to define the scattering transform to be non-expansive and to preserve the signal normalisation.

The norm and distance on a transformation $Tx = \{x_n\}_n$ and output a family of signals is

$$||Tx - Tx'||^2 = \sum_n ||x_x - x'_n||^2. \tag{4.16}$$

An $\epsilon > 0$ exists, for all $\omega \in \mathbb{R}^2$,

$$1 - \epsilon \leq |\hat{\phi}(\omega)|^2 + \frac{1}{2}\sum_{j=0}^{\infty}\sum_{r\in G}|\hat{|}\psi(2^j r\omega|^2| \leq 1. \tag{4.17}$$

After applying Plancherel formula which proves that if x is real, then $Wx = \{x \star \phi_{2^J}, x \star \psi_\lambda\}_{\lambda\in\mathcal{P}}$ satisfies

$$(1-\epsilon)||x||^2 \leq ||Wx||^2 \leq ||x||^2 \tag{4.18}$$

with

$$||Wx||^2 = ||x \star \phi_{2^J}||^2 + \sum_{\lambda\in\mathcal{P}} ||x \star \psi_\lambda||^2 \tag{4.19}$$

The wavelet transform is non-expansive and invertible with a stable inverse when $\epsilon < 1$.

The Morlet wavelet with $\phi(u) = \exp(-|u|^2/(2\sigma^2))/(2\pi, \sigma^2)$, $\sigma = 0.7$ and $\epsilon = 0.25$ makes the wavelet transform non-expanding and invertible operator with a stable inverse.

The modulus is non-expansive in that $||a| - |b|| \leq |a - b$ for all $(a, b) \in \mathbb{C}^2$. With the wavelet modulus propagator $\tilde{W} = \{x \star \phi_{2^J}, |x \star \psi_\lambda|\}_{\lambda \in \mathcal{P}}$ is obtained by a wavelet transform W and a modulus that are non-expansive, then $\tilde{W}$ is also non-expansive

$$||\tilde{W}x - \tilde{W}y|| \leq ||x - y||. \tag{4.20}$$

If we let $\mathcal{P}_\infty$ be the set of all paths of any length $m \in IN$, the norm of $Sx = \{S[p]x\}_{p \in \mathcal{P}_\infty}$ is

$$||Sx||^2 = \sum_{p \in \mathcal{P}_\infty} ||S[p]x||^2, \tag{4.21}$$

and since S iteratively applies to $\tilde{W}$, which is non-expansive, S is also non-expansive as well as being stable to additive noise $||Sx - Sy|| \leq ||x - y||$.

For W to be unitary, then $\epsilon = 0$, and when W is unitary, then $\tilde{W}$ preserves the signal norm $||\tilde{W}x||^2 = ||x||^2$. The signal energy is then equal the the sum of the scattering energy of each layer as well as the last propagate layer.

$$||x||^2 = \sum_{m=0}^{\overline{m}} \sum_{[\epsilon \mathcal{P}^m]} ||S[p]x||^2 + \sum_{p \in \mathcal{P}^{\overline{m}+1}} ||U[p]||^2. \tag{4.22}$$

Using appropriate wavelets (in this case the Morlet wavelet), the energy of the mth layer of $\sum_{p \in \mathcal{P}^{\overline{m}+1}} ||U[p]||^2$ converges to zero when m increases, and the energy of the scattering coefficients of orders $\geq m$. If $\overline{m}$ goes to infinity, then the scattering transform preservers the entire signal energy:

$$||x||^2 = \sum_{p \in \mathcal{P}_\infty} ||S[p]x||^2 = ||Sx||^2. \tag{4.23}$$

For more sparse wavelet coefficients, more energy is propagated at deeper layers. The first layer $S[\lambda_1]x = |x \star \psi_{\lambda_1}| \star \phi_{2^J}$ converges to $||\phi||^2 ||x \star \psi_\lambda||_1^2$ and the more sparse $x \star \psi_\lambda$, the smaller $||x \star \psi\lambda||$ is and the more energy is propagated in deeper layers.

Scattering coefficients converges to zero as m increases, and according to [5] on CalTech101 dataset, the scattering energy goes below one percent for $m \geq 3$. The energy decay occurs as $U[p]x$ becomes a lower frequency signal as m increases. The modulus computes a regular envelope of oscillating wavelet coefficients, where the wavelet coefficient energy is pushed towards lower frequencies. The important portion of the energy of $U[p]x$ is captured by the low-pass filter ϕ_{2^J}, that comes from the outputs of $S[p]x = U[p]x \star \phi_{2^J}$.

An extra property of the scattering transform, is that the scattering energy propagates along a subset of frequency decreasing paths. The envelop $|x \star \psi_\lambda|$ is more regular than $x \star \psi_\lambda$ which causes $x \star \psi_{\lambda'}$ being non-negligible if $\psi_{\lambda'}$ is located at lower frequencies than ψ_λ. The wavelet modulus propagates the scattering energy along frequency-decreasing paths $p = (\lambda_1, \ldots, \lambda_m)$, where $|\lambda_k| < |\lambda_{k-1}$ for $1 \leq k < m$.

Scattering coefficients that go on any other path compared to these frequency decreasing paths have negligible energy. This causes 99 percent of the energy from the scattering transform is absorbed in $m \leq 3$. When computing the scattering transform then, it is enough to compute along these frequency-decreasing paths with results in a small convolution network, with less computing time.

Energy preservation does not cause signal information to be preserved. The scattering transform is calculated by iteratively applying $\tilde{W}$, and inverting S would require inverting $\tilde{W}$. The wavelet transform W is linear invertible operator, so inverting $\tilde{W}z = \{z \star \phi_{2^J}, |z \star \psi_\lambda|\}_{\lambda \in \mathcal{P}}$ will recover the complex phases of the wavelet coefficients that were removed by the modulus. Phase of Fourier coefficients cannot be recovered by the modulus, but the wavelet coefficients can be recovered by their modulus, with $\tilde{W}$ having a continuous inverse and the phase recoverable by using a convex optimization.

S still cannot be inverted because information is lost when computing the scattering coefficients of the last layer. The propagation coefficients $|U[p]x \star \psi_\lambda|$ of the next layer are removed because they are not invariant and that they contain minimal energy. The number of these coefficients are larger than the total number of scattering coefficients at previous layers and inverting these small coefficients will produce an error. Reconstruction of the signal with $\overline{m} = 2$ still produces a good quality with the caveat that number of scattering coefficients is comparable to the number of signal samples.

When used in classification, the important property of the scattering transform is its Lipschitz continuity to deformations. To reiterate, wavelets are stable to deformations and the modulus is computed with the deformations. Let an image that has been deformed by the displacement field τ be $x_\tau(u) = x(u - \tau(u))$ and let $||\tau||_\infty = \sup_u |\tau(u)|$ and $||\nabla||_\infty = \sup_u |\nabla\tau(u)| < 1$. The scattering transform Sx along paths of length $m \leq \overline{m}$, then for signals x of compact support:

$$||Sx_\tau - Sx|| \leq C\overline{m}||x||(2^{-J}||\tau||_\infty + ||\nabla\tau||_\infty)). \tag{4.24}$$

The scattering transform is Lipschitz continuous to deformations if $2^j \geq ||\tau||_\infty / ||\nabla\tau||_\infty$, where the translation term can be neglected

$$||Sx_\tau - Sx|| \leq \overline{m}||x||||\nabla||_\infty. \tag{4.25}$$

4.5 Fast Scattering Computations

A fast scattering computations is the scattering transform over the frequency decreasing paths where most of the scattering energy is. The frequency decreasing path is along $p = (2_1^{-j_1}, \ldots, 2^{-j_m} r_m)$ which satisfies $0 < j_k \leq j_{k+1} \leq J$. A wavelet transform computed over K rotation angels has the total number of frequency-decreasing paths $K^m \binom{J}{m}$ of length m. With ϕ_{2^J} is the low-pass filter that is scaled by 2^J, the scattering transform $S[p]x(u) = U[p]x \star \phi_{2^J}(u)$ is uniformly sampled at intervals $\alpha 2^J$ with $\alpha = 1$ or $1/2$. The resulting number of scatting coefficients $S[p]x$ of an image $x(u)$ with N number of pixels is $\alpha^{-2} 2^{-2J} N$. Therefore the scattering network on the image with the depth of $\overline{m}$ is

$$P = N\alpha^{-2}2^{-2J}\sum_{m=0}^{\overline{m}} K^m \binom{J}{m}. \tag{4.26}$$

The depth of $\overline{m} = 2$ would be chosen since as above, the energy beyond $m = 2$ is negligible.

The algorithm here details the process of computing the scattering transform on set $\mathcal{P}_{\downarrow}^{m}$ of the frequency decreasing paths of length $m \leq \overline{m}$. The set $\sqrt{}^{p}_{\downarrow}\{\emptyset\}$ is the original image $U[\emptyset]x = x$. $p + \lambda$is the path that begins by p and ends $\lambda \in \mathcal{P}$. At $\lambda = 2^{-j}r$, $U[p + \lambda]x(u) = |U[p]x \star \psi_\lambda(u)$ has the energy frequencies mostly below $2^{-j}\pi$). Reducing computations is done by sub-sampling the convolution at intervals of $\alpha 2^j$ with $\alpha =$ 1or 1/2 which will avoid aliasing.

Algorithm 5 Fast Scattering Transform

```
for all m do
    for all p do
        Output S[p]x(α2^J n)
    end for
    for all p + λ_m with λ_m = 2^{-j_m} r_m do
        Compute U[p + λ_m]x(α2^{j_m} n)
    end for
end for
for all p do
    Output S[p]x(α2^J n)
end for
```

At layer m are $K^m\binom{J}{m}$ propagated signals $U[p]x$. These signals are samples at the intervals $\alpha 2^{j_m}$ where it is dependent on p. There are also $K^m\binom{J}{m}$ scattering signals $S[p]x$, who are sub-sampled by 2^j, and so, will have fewer coefficients. The number of operations needed to compute each layer is driven by the $O((K/3)^m N \log N)$ operations to compute the internal propagated coefficients. At $K > 3$, overall computational complexity is $O((K/3)^{\overline{m}} N \log N)$.

4.6 Scattering Stationary Processes

Image textures can be seen as realizations of station processes $X(u)$. The expected value of X by $E(X)$ is not depended on u. The power spectrum of images are not sufficient to discriminate between two images; the second-order moments can instead. A scattering representation of stationary processes depends on the second-order and higher-order moments to be able to discriminate between two image textures. Scattering representations of stationary processes also do not suffer from the large variance of higher moments estimators, as scattering representation are computed with a non-expansive operator.

The propagated signals of $X[u]$ is $U[p]X(u)$, and are stationary as it is computed with a cascade of convolutions and modulus which keeps it stationary. The expected value of the propagates signals $U[p]X(u)$ does not depend on u and the expected scattering transform of the signal is

$$\overline{S}X(p) = E(U[p]X). \tag{4.27}$$

The windowed scattering transform calculates the estimator $\overline{S}X(p)$ by averaging the propagated signals $U[p]X$ with ϕ_{2^J} which results in the formula:

$$S[p]X(u) = U[p]X \star \phi_{2^J}(u), \tag{4.28}$$

and since $\int \phi_{2^J}(u)du = 1$, then the estimator is:

$$E(S[p]X) = E(U[p]X) = \overline{S}X(p). \tag{4.29}$$

The windowed scattering transform conserves the second-order of stationary processes:

$$\sum_{p \in \mathcal{P}_\infty} E(|S[p]X)^2 = E(|X|^2), \tag{4.30}$$

and the second-order of wavelet coefficients - which are useful for texture discrimination, can be recovered from scattering coefficients. For a path $p = (\lambda_1, \ldots, \lambda_m)$, and write $\lambda + p = (\lambda, \lambda_1, \ldots, \lambda_m)$ then the scattering coefficients are:

$$S[p]|X \star \psi_\lambda| = S[p]U[\lambda]X = S[\lambda + p]X, \tag{4.31}$$

and when replacing X with $|X \star \psi_\lambda|$ results in:

$$\sum_{p \in \mathcal{P}_\infty} E(|S[\lambda + p]X|^2) = E(|X \star \psi_\lambda|^2). \tag{4.32}$$

The scattering transform$\overline{S}X(p)$ depends on the normalized high-order moments of X of orders up to 2^m if p has the length of m. The scattering coefficients discriminate textures having the same second-order moments that have different higher order moments. Textures that have the same power spectrum and the second-order moments being discriminated by scattering coefficients are viewable with figure 4.3. Scattering coefficients $S[p]X$ of $m = 1$ and $m = 2$ are shown where the order 1 scattering coefficients of the two textures are not discriminated, whereas the order 2 scattering coefficients are able to discriminate between the two texture images.

Signal processing has difficulties with high-order moments due to the large variance estimators produce. Large variance from high-order moments is caused by the large coefficient outliers that came from X^q for $q \geq 2$. Since scattering computes with a non-expansive operator, it results in lower variance estimators. Variance of the estimators $\overline{S}X(p) = E(U[p]X)$ by $S[p]X = U[p]X \star \phi_{2^J}$ are reduces as the averaging scale 2^J increases. Texture images are observed to have their scattering variance $\sum_{p \in \mathcal{P}_\infty} E(|S[p]X - \overline{S}X(p)|^2)$ reduce to zero as 2^J increases.

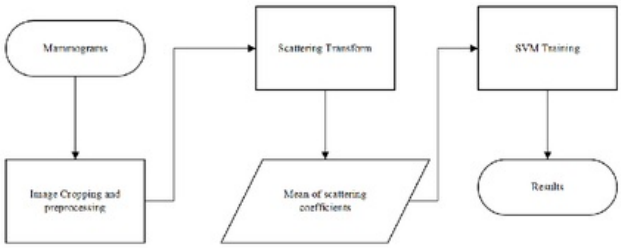

Figure 4.4: Overall method of the scattering transform

Let $\overline{\mathcal{P}}_\infty$ be the set of all paths $p = (\lambda_1, \ldots, \lambda_m)$ for all $\lambda_k = 2^{j_k} r_k \in 2^{\mathbb{Z}} \times G^+$ and all length m. The scattering variance decal implies that second-order moment equal to the energy of expected scattering coefficients in $\overline{\mathcal{P}}_\infty$

$$||\overline{S}X||^2 = \sum_{p \in \overline{\mathcal{P}}_\infty} |\overline{S}X(p)|^2 = E(|X|^2) \tag{4.33}$$

Expected value $E(S[p]X) = \overline{S}X(p)$, then

$$E(|S[p]X^2|) = \overline{S}X(p)^2 + E(|S[p]X - E(S[p]X)|^2) \tag{4.34}$$

4.7 Method for Scattering Transform

The scattering transform will be applied to all of the images, which results in scattering coefficients for each image. The resulting scattering convolution network contains the coefficients of the sub-sampled coordinates of the original image, where the mean of these coefficients is the energy from the scattering coefficients. These will be used at as the training data for the SVM classifier. The images are grouped the same way as the method in the previous chapter - with three groups of images to result in three different SVM classifiers.

The images that will undergo the scattering transform are the same that was used with the wavelet transform for feature extraction on the previous chapter. Computing the scattering coefficients of these images is computationally heavy, so these images will are to be resized to compute these coefficients in a reasonable time. The figure 4.4 is the flow diagram of the overall method of the scattering transform

4.8 MatLAB Method

Scattering transform is done with MatLAB using the software ScatNet which can be obtained at: http://www.di.ens.fr/data/software/scatnet/. This software performs the scattering transform on audio - 1-dimensional files, and images - 2-dimensional files. With this thesis focusing on images, we will use the software's 2-dimensional functions. ScatNet

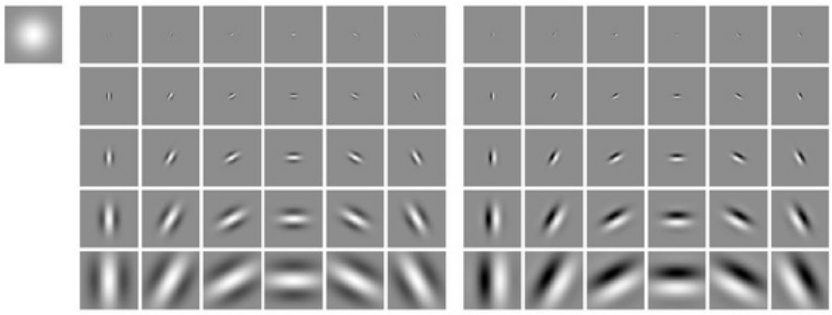

Figure 4.5: Wavelet filter banks

includes code that has been used for other papers that use the scattering transform for classification - in particular a reproduction of the code for translation-invariant scattering on texture images has been included [4].

The code used for the breast image scattering transform is a slightly modified code to the original where modifications are done to adapt to the breast images and to use SVM classification rather than affine classification. The core of the code - generate image database and scattering transform is mostly the same.

ScatNet's functions that focuses on images are the Wop = wavelet_factor_2d(size(x)) - which computes the wavelet transform operators that will be applied to an image of size x, S = scat(x, Wop), which performs the translation-invariant representation of an image (or other one-dimensional signal) using the wavelet operators Wop.

An alternative function for computing the wavelet transform operator is the Wop = wavelet_factory_2d_pyramid(), which gives a faster algorithm for computing the scattering coefficients at different scales and does not require the size of the image to compute the scattering coefficients. Since we are dealing with multiple sizes for the breast images - Wop = wavelet_factory_2d_pyramid() is used for computing the wavelet transform operator. There is a difference between the two functions, in the 'wavelet_factory_2d' stores the filters in the Fourier domain and the 'wavelet_factory_2d_pyramid' uses its spatial support. The 'wavelet_factory_2d_pyramid' is also not as customisable in terms of anti-aliasing and numerical approximation.

The wavelet filter banks that is used for the wavelet transform operator is the Morlet wavelet. The filter banks of the wavelet transform operators obtained through the wavelet_factory functions [Wop, filters] = wavelet_factory_2d(), and can be viewed with the display_filter_bank_2d(filters). The figure 4.5 shows the wavelet filter banks with 5 scales and 6 orientations.

This results in a cell structure containing the scattering coefficients of the different layers and the filters used for the scattering transform. There includes options to specify the scale J, the number of orientations L, scale per octave Q, and the maximum scattering order M for the wavelet transform operator. For the cascading wavelet transform operator - Wop = wavelet_factory_2d_pyramid(), we only specify the number of octaves J to be 5.

The structure S is transformed into a three dimensional array using the function S_mat = format_scat(Sx), where the resulting array S_mat, is capable of numeric operations. The scattering transform on images is a computationally heavy process with the size of original image. By taking advantage of the scattering transform being translation invariant and stable to deformation, the images can be resized to a smaller image without affecting the scattering coefficients. The breast images that are usually in excess of thousands of pixels in height and width are resized to a 200×200 size before computing the scattering coefficients.

Once an image has been resized and undergone the scattering transform, the margins along the second and third dimensions are removed, and the mean (calculating the normalised scattering variance) across the positions of the scattering coefficients is calculated for the SVM classifier.

ScatNet performs classification using an affine space training, and a pipeline to the LibSVM toolbox for SVM classification. The LibSVM toolbox is not part of MatLAB's classification toolbox, but it is still a well regarded SVM classification model. ScatNet's pipeline for SVM classification depends on the proportion of data to be training and testing data.

Batch computing is possible with ScatNet by creating a database of the files to undergo the scattering transform. The database is generated by setting up the source of the files, and obtaining the path to those files. ScatNet has a built in function called kthtips_src or uiuc_src, which obtains the location of the images for the transform and creates the database for those files.

Same with the feature extraction on wavelet transform in the previous chapter, the images are split into three groups of images - benign-cancerous, benign-normal and cancerous-normal. The images are divided into those files, and a database is created for the three groups to compute the scattering.

The next step for batch computing the images is to specify an anonymous function 'fun' that will compute the scattering of each image in the database. This function is applied to every image in the database by using the function ScatNet 'trans_scatt_all = srcfun(fun, src)'. The resulting 'trans_scatt_all' stores the scattering coefficients of every image in a cell array, with each cell containing the complete scattering representation of each image.

The next step for the scattering representation of the images is to reorganise the cell array into a numeric array for use in the SVM. Another anonymous function is created that will transform the scattering coefficient that has been saved as a structure within a cell - into a numeric array within a cell. The format_scat(Sx) that is mentioned before is used to transform the scattering coefficient structure into a numeric array for numeric manipulation. Once the scattering coefficients are made into a numeric array, the margins along the second and third dimensions are removed. The resulting numeric array then gets the average across each position which the resulting array is the data that will be used for training and testing.

The array is still a cell array with each cell containing the average of the scattering coefficients across each position - the cell array is manipulated to be a numeric array.

This is done with the 'cellsrc2db', which simply obtains the numeric array from each cell and adds the numeric array into a new numeric array within a database - with each row in the array being an image and each column is the average of each position.

Once the scattering is complete and the mean of the images, the data is saved to save time as computing the scattering coefficients is computationally heavy. The next time the MatLAB script is run, it locates the save precomputed scattering coefficients without running the transformation again.

The database is a structure that contains the directory of the images, and the features of each image in a numeric array. SVM classification for ScatNet is mentioned above which uses the libSVM library for the SVM. Training and testing data are split with the same proportions with the previous chapters with 5% to 95% of data being training data. The ratio of the data is selected with the 'create_partition'. It randomly selects the images the will be used as the training data at the proportion that the user specifies.

Training the SVM uses the database that contains the feature - mean of each image in that database and the selected testing data that was picked when the proportion was set. These are arguments that for the function 'svm_train' which is the pipeline for training the libSVM SVM classifier. This pipeline gathers the average of the images in the library and calculates the kernel of these average before it passes the kernel off to LibSVM as the features of the images. The labels of those images are also passed to the LibSVM. The SVM uses the Gaussian kernel for non-linear classification - the same used in the previous chapter.

The resulting SVM model then goes through testing using the testing data chosen previously. Testing is done by using the testing data is done with the 'svm_test' function that takes the database of images, the SVM model and the testing set as arguments to generate the labels for the testing data. The function 'svm_test' is also a pipeline for the SVM model created by libSVM, where similarly - it calculates the kernel of those averages before passing the kernel as the features of the testing data. The resulting labels are then used to calculate the error rate of the SVM.

Error calculation is done with the ScatNet function 'classi_err', which takes the arguments of the testing labels, the chosen testing data and the database of the images. It simply compares the labels generated from the SVM and compares to the expected labels from the database. Error is calculated as the number of incorrect labels divided by the total number of images in the database:

$$\text{Error} = \frac{\text{No. of incorrect labels}}{\text{Total No. of images}} \times 100 \tag{4.35}$$

The training and test of the SVM is done multiple times for proportion of 5% to 95% in 5% increments. This is done with the three databases benign-cancerous, benign-normal and cancerous-normal.

Algorithm 6 Scattering transform of the images

```
for all Image groups do
    Generate source structure of image database
    Specify proportions for training and testing data
    Build wavelet transform filters
    for all Images in database do
        Apply scattering transform
        Reform scattering structure
        Obtain mean across all position
    end for
    for all Training size proportions do
        Train SVM with training data
        Test SVM with testing Data
        Output accuracy rates for training size proportion
    end for
end for
```

Chapter 5

Results

5.1 Wavelet Transform Support Vector Machine Results

There is a total of 24 matrices, three groups of data to be sorted into each groups containing eight matrices for representing each mother wavelet used for wavelet decomposition. Each matrix that goes through SVM training and testing is done multiple times, for various proportions of training and testing data. These percentages are range from 5% of the data to 95% of the data and are saved to compare how different mother wavelet affect the accuracy rate of the SVM classifier. The total number of images in the benign-cancerous group is 6,811, benign-normal is 6,007, and the cancerous-normal group is 6,236. Complete accuracy rate data is in the appendix.

With SVMs, accuracy rates rises and the training data increases. This increase in accuracy is observed in the data gathered, as the accuracy rates of all the groups and mother wavelets used increases as a higher percentage of training data is being used to train the SVM.

Unsurprisingly, the lowest accuracy rates are then the SVM is trained with 5% of the data and that the highest accuracy rates are when the SVM is trained with 95% of the data. Something that is observed is that the accuracy rates do not rise much after 20-25% of the data are used as training data with the benign-cancerous group, about 1-2%. The trend is observed with all wavelet types. The benign-cancerous group has a similar pattern, with minor increase in accuracy after

The overall testing accuracy results from the Haar wavelet is 67.86%, db2 is 71.89%, db3 is 71.81%, db6 is 72.69%, db8 is 72.33%, db9 is 72.31%, db10 is 72.67% and bior3.7 is 75.67%. Observing the data indicates that there is not much difference between the Daubechies families of wavelet when it comes to the accuracy rates for feature extraction, with minimal differences between them. The largest difference is the features extracted using the Haar wavelet and the biorthogonal 3.7 wavelets, with the features extracted using the Haar wavelet being the least accurate, while the accuracy from biorthogonal 3.7 wavelets being the highest by a larger margin.

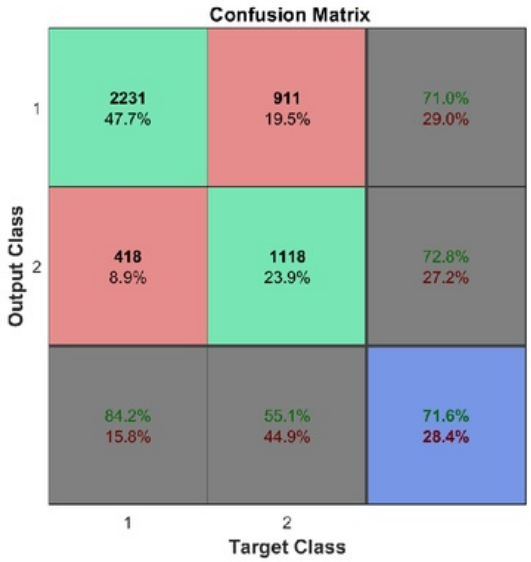

Figure 5.1: Confusion matrix of cancerous-normal class with bior3.7 wavelet

The accuracy rates in the cancerous-normal group are unexpected; common sense would expect a cancerous breast image is very different to a normal healthy breast. The confusion matrix shows that the majority of the error is with a false-positive result - a normal image being labelled as a cancerous image. The figure 5.1 shows the confusion matrix of the cancerous-normal group using the biorthogonal 3.7 mother wavelet at 75% training data.

Accuracy with cross validation is not the focus of this thesis. However, it is observed that the cross-validation results show a higher accuracy rate, averaging 80%. It is observed as well that the accuracy rate of cross-validation decreases as the training size increases. This trend occurs throughout the three groups of images with all mother wavelets tested.

With the features from the detail coefficients, the overall accuracy rate is very similar compared to using the approximation coefficients for feature extraction. Using the same mother wavelet for the transform, the average accuracy between the two methods are within 2-3% of each other, with the features of the detail coefficients being slightly less accurate. Within the three groups of classes, The accuracy rate for the benign-normal class is the highest, with the lowest accuracy comes from the benign-cancerous group. Between the two methods, the trend is that the use of approximation for feature extraction show similar or better accuracy with the benign-cancerous and the cancerous-normal groups.

At 50% training data, the accuracy rate of using the detail features is slightly lower compared to using the approximation features except for using the Haar mother wavelet.

The most noticeable difference between the feature extraction methods is the accuracy rates using the Haar wavelet. The method of using the detail coefficients for feature extraction shows an overall 5% higher accuracy compared to using approximation coefficients for feature extraction, and the accuracy rates between the two methods benign-normal group have an accuracy different of a little over 10%, with a average 79% accuracy rate for that group.

The full code for the scattering will be in the appendix.

5.2 Scattering Support Vector Machine Results

The results of the SVM using the scatting transform showed that the average accuracy of 71.23% of all three groups and all proportions of training size. Complete accuracy data is in the appendix. As expected with SVMs, the accuracy rate increases as the training size increases as well. The greatest increase in accuracy is when the proportion of the data reaches 20-25% of training data which is observed with all three groups of images. Beyond the 20-25% proportion of training data, there is roughly a 1-2% increase in accuracy.

The highest average accuracy rate is with the benign-normal group, with the accuracy rate of 73.32% and the lowest average accuracy rate is with the cancerous-normal group with an accuracy rate of 69.43%. The benign-cancerous group has an accuracy rate that averages at 70.93%.

Accuracy percentages at 50% of training data and above show an average accuracy rate of 72.25% of the three groups, with the benign-cancerous group being and average 72.2% accurate, benign-normal being an average 74.86% accurate and the cancerous-normal being and average 70% accurate. Same observation as overall accurate, the cancerous-normal group being the least accurate and the benign-normal group being the most accurate. The difference between the accuracy of the benign-normal and cancerous-normal is somewhat minimal at roughly 5% difference.

Within the 50% and 95% training data, there isn't much difference for the accuracy rate between proportions. While the accuracy rates do increase, the standard deviation is 0.53% for benign-cancerous, 0.65% for benign-normal and 0.58% for cancerous-normal.

5.3 Results Summary

Between the two transform methods, the results show a comparable average accuracy rate of roughly 70%. The wavelet transform and feature extraction method revealed that there are some differences when using different mother wavelet for the wavelet transform when getting the accuracy rates of various mother wavelets.

Using the wavelet transform method showed a difference in accuracy when using a different choice of features. The features extracted from the approximation coefficients are slightly higher than the features extracted from the detail coefficients. The biggest change is the accuracy rates from using the Haar wavelet where it showed a larger difference in

accuracy, where the features from the detail coefficients are much more accurate compared to the features from the approximation coefficients.

The scattering transform method showed slightly higher accuracies when compared to the wavelet transform that used the Haar and Daubechies family of wavelets, and a little lower accuracy rate when compared to the bi-orthogonal 3.7 wavelets. However, these differences are minimal and result in similar accuracies. The exception is when using the features of the detail coefficients using the haar wavelet the accuracy rates between using the scattering transform where using the haar wavelet transform showed a higher average accuracy rate with the benign-normal group.

In both cases, increasing the size of the training data increases the accuracy rate. It is observed with both methods that the increase in accuracy is minimal when the training data reaches to around $20 - 25\%$.

Chapter 6

Conclusions

The original objective was to use the wavelet transform for feature extraction and to use SVM for classification. This thesis covered the use of wavelet transform and feature extraction from the wavelet transform. It has also covered the use of the scattering transform for translation invariant representation of images.

The theories of the wavelet transform and the scattering transform is studied and applied using the MatLAB onto breast images for image classification. The breast images are provided by Abdulla-Al Nahid and Dr Yinan Kong and were provided with the goal of developing a classifier to discriminate between benign, malignant and normal breast images.

Using MatLAB, the breast images undergone wavelet transform and went through feature extraction and a proportion of the data was sent as training data for an SVM and tested with the remaining data. The images split into three groups, the benign-cancerous group, the benign-normal group and the cancerous and normal group. These groups will go through SVM classification separately.

Similarly, the breast images had the scattering transform applied, and the mean of the resulting scattering coefficients was used as the training and testing data for the SVM classifier.

The proportion of data used for training and test data start at 5% to 95% at 5% increments. The accuracy results are gathered for both the wavelet transform method and the scattering transform method.

The result collected from the classifier showed a 70% accuracy with both the wavelet feature extraction method and the scattering transform method. It revealed that the amount of training data determines the accuracy of the SVM classifier, where increasing the training data also enhances the accuracy of the SVM.

The results also show that a relatively small amount of training data was needed to achieve an accuracy rate that is marginally worse than the overall average accuracy - indicating that the SVM is an effective classifier with small amounts of data.

The resulting thesis project produced a method of using the wavelet transform for feature extraction and the scattering transform for a convolution network. Both approaches showed similar results when the wavelet transform used the biorthogonal 3.7 wavelets as

the mother wavelet for the transforms.

6.1 Future work

The accuracy rates are not what is expected when compared to other papers that have done similar work with wavelets for feature extraction and classification. The accuracy rate of previous research shows accuracy rate up to and around 70% - 80%.

Many papers that have worked on similar problems used various methods and chosen features for the wavelet transform and feature extraction [18].

Joan Bruna and Stéphane Mallat were able to get state-of-the-art classification results using the scattering transform on kth-tips texture images, which has been reproduced using the code in ScatNet and achieved the same accuracy results of around 98% using the affine training method [5].

Similarly, Henrik Nicolay Finsberg in his master thesis [8], used the scattering transform on ultrasound images and achieved an average error rate of 20% and a best case error rate of 6% also using the affine training method.

Future work can be done with wavelet transform and feature extraction with several papers introduce Gaussian filtering to the images before undergoing the wavelet transform to grant a better contrast on the image that can potentially produce more accurate results.

Laurent Sifre and Stéphane Mallat extended the scattering transform for rotation, scaling and deformation invariant scattering [15] representation. This method produces even more accurate classification results on texture images and could be applied to the breast images to see if this method produces more accurate results.

Also, SVM classification can be extended for multi-class classification. MatLAB's statistical learning toolbox is already prepared for multi-class classification when using SVM by having one-vs-one and one-vs-all multi-class classifier. The library LibSVM is already built to accept multi-class classification which ScatNet default is a pipeline for the library [2].

Chapter 7

Abbreviations

MRI	Magnetic Resonance Imaging
CT	Computed Tomography
DW	Discrete Wavelets
DWT	Discrete Wavelet Transform
CWT	Continuous Wavelet Transform
ST	Scattering Transform
SCN	Scattering Convolution Network
WST	Window Scattering Transform
SVM	Support Vector Machine
RBF	Radial Basis Function

Appendix A

Consultation Form

Consultation Meetings Attendance Form

Week	Date	Comments (if applicable)	Student's Signature	Supervisor's Signature
1	1/8/16	Met to establish Context		
1	3/8/16	Received hard copies		
2	10/8/16	Need to write our wanted aims and to discuss last version		
3	19/8/16	Nabil gave me a paper to study for setting world		
4	29/8/16	Start better understanding		
5	1/9/16 ~~1/8~~	Asked for advice for setting Goals		
6	8/9/16	Get results for word		
7	15/9/16			
8	22/9/16	Start writing report		
10	12/10/16	Write report only		

Figure A.1: Consultation Meetings Attendance Form

Appendix B

Brest Image Wavelet Transform

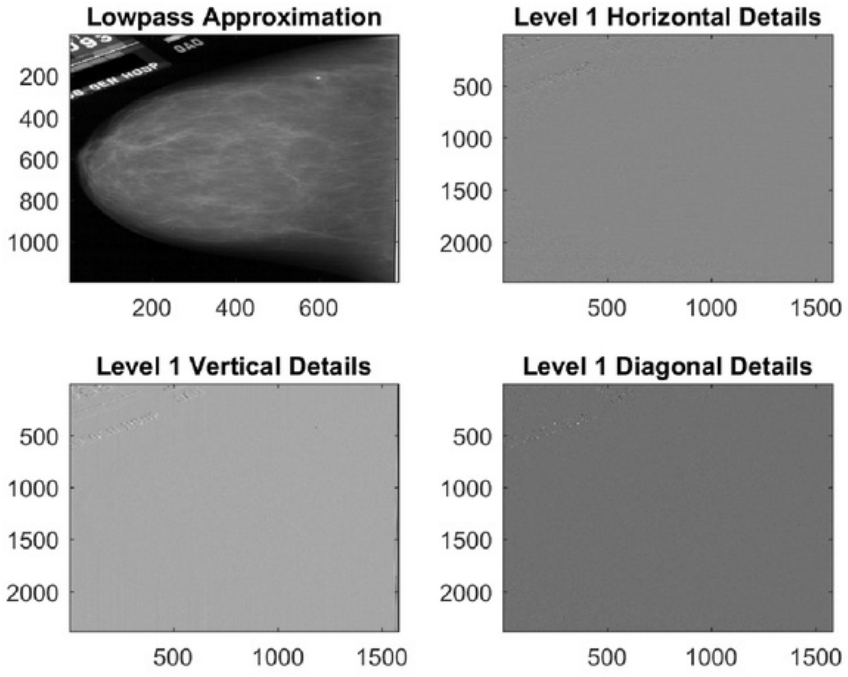

Figure B.1: Breast mammogram undergone wavelet transform

Appendix C

Breast Image Scattering Transform

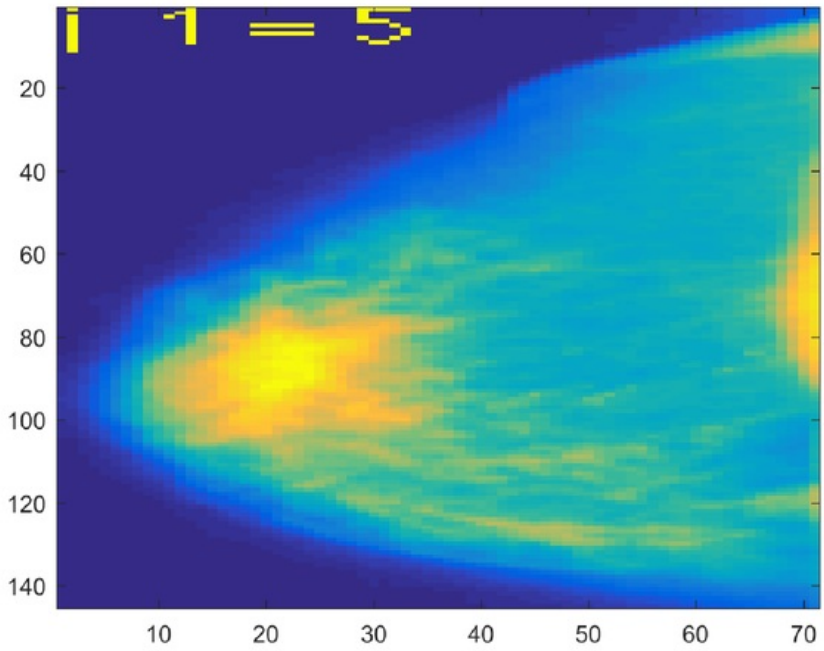

Figure C.1: 1st layer of a breast image undergone scattering transform$S_0x = \{x \star \phi\}$

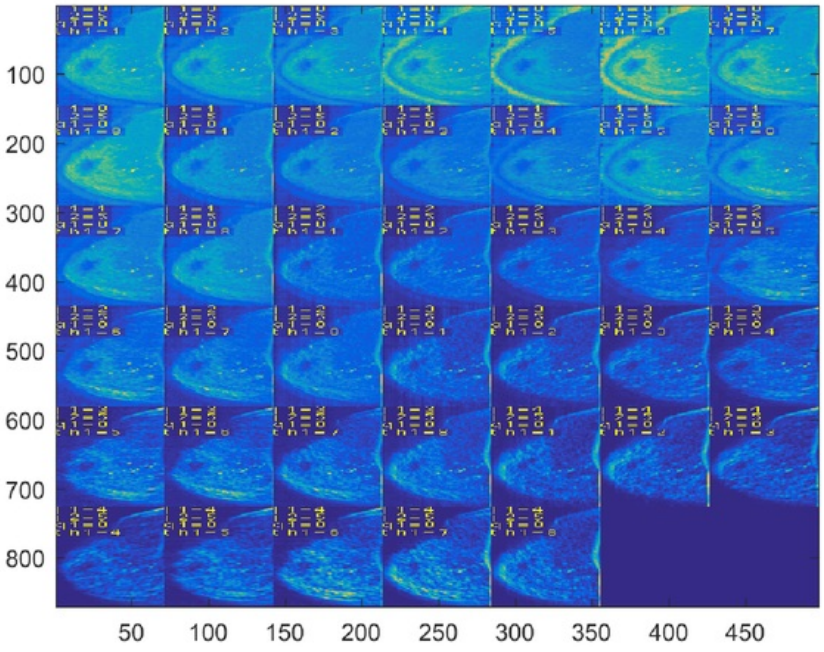

Figure C.2: 2nd layer of a breast image undergone scattering transform$S_1x = \{|x \star \psi_{j_1,\theta_1}| \star \phi\}$

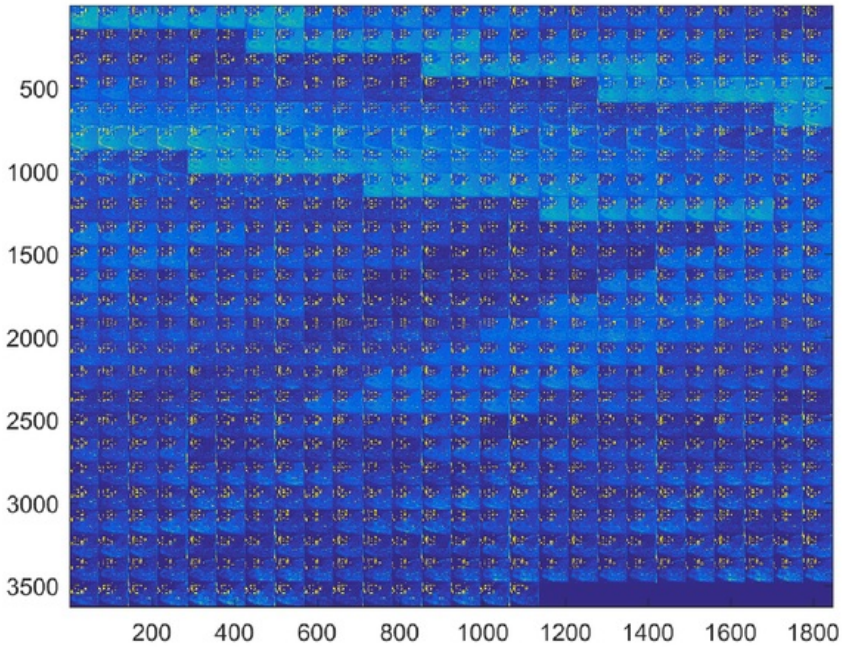

Figure C.3: 2nd layer of a breast image undergone scattering transform$S_2x = \{||x \star \psi_{j_1,\theta_1}| \star \psi_{j_2,\theta_2}| \star \phi\}$

Appendix D

Wavelet Transform Code

D.1 Cropping Images

```
files = dir('Images\Beningn\*.png');
for i = 1 : length(files);
    filename = strcat('Images\Beningn\',files(i).name);
    I = imread(filename);
    I2 = imcrop(I);
    imwrite(I2,['Images\Beningn \',num2str(i),'png']);
end
```

D.2 Wavelet transform

D.2.1 Features from Approximation Coefficients

```
% Takes file location, wavelet and level
% and saves a database of images that have undergone wavelet
   decomposition

function wave_dec_func(files, wname, lvl)
% File Location for Beningn Images
files_loc = dir(strcat('Cropped Images\',files,'\*.png'));

%Preallocate coef and time
coef = zeros(length(files_loc),22);

time_start = clock;
time_elapsed_at_last_print= 0;
time_between_each_print = 1;

for i = 1 : length(files_loc);
```

```
16 filename = strcat('Cropped Images\',files,'\',files_loc(i).name)
       ;
17
18 X = imread(filename);
19
20 % Wavelet Decomposiition
21 [C,S] = wavedec2(X,lvl,wname);
22
23 for j = 1:lvl
24 A{j} = appcoef2(C,S,wname,j);
25 end
26
27 % Extract feature
28 coef(i,1) = mean2(A{1});
29 coef(i,2) = median(A{1}(:));
30 coef(i,3) = std2(A{1});
31 [coef(i,4), coef(i,5)] = psnr(double(A{1}),imresize(double(X),
       size(A{1})));
32 coef(i,6) = entropy(A{1});
33 coef(i,7) = skewness(A{1}(:));
34
35 coef2(i,8) = mean2(A{2});
36 coef2(i,9) = median(A{2}(:));
37 coef2(i,10) = std2(A{2});
38 [coef2(i,11), coef2(i,12)] = psnr(double(A{2}),imresize(double(X
       ),size(A{2})));
39 coef2(i,13) = entropy(A{2});
40 coef2(i,14) = skewness(A{2}(:));
41
42 coef3(i,15) = mean2(A{3});
43 coef3(i,16) = median(A{1}(:));
44 coef3(i,17) = std2(A{3});
45 [coef3(i,18), coef3(i,19)] = psnr(double(A{3}),imresize(double(X
       ),size(A{3})));
46 coef3(i,20) = entropy(A{3});
47 coef3(i,21) = skewness(A{3}(:));
48
49 % Apply label
50 if(strcmp(files, 'Benign'))
51 coef(i,end) = 1;
52 elseif(strcmp(files, 'Cancer'))
53 coef(i,end) = 2;
54 elseif(strcmp(files, 'Normal'))
```

```
55 coef(i,end) = 3;
56 end
57
58 % Get estimated time to finish
59 time_elapsed = etime(clock, time_start);
60 if (time_elapsed - time_elapsed_at_last_print >
      time_between_each_print)
61 time_elapsed_at_last_print =  time_elapsed;
62 estimated_time_left = time_elapsed * (length(files_loc)-i) / i;
63 fprintf('%d / %d : estimated time left %d seconds\n',...
64 i,length(files_loc),floor(estimated_time_left));
65 end
66
67 end
68
69 % Save location
70 location = strcat('Cropped Details\',files,'\', wname, int2str(
      lvl), 'train.mat');
71
72 save(location, 'coef', 'coef2', 'coef3', '-v7.3');
73 clear;
74 end
```

D.2.2 Features from Detail Coefficients

```
1 % Takes file location, wavelet
2 % and saves a database of images that have undergone wavelet
     decomposition
3
4 function wave_dec_func(files, wname)
5 % File Location for Beningn Images
6 files_loc = dir(strcat('Cropped Images\',files,'\*.png'));
7
8 % Preallocate coef and time
9 coef = zeros(length(files_loc),10);
10
11 time_start = clock;
12 time_elapsed_at_last_print= 0;
13 time_between_each_print = 1;
14
15 for i = 1 : length(files_loc)
16 length(files_loc);
17 filename = strcat('Cropped Images\',files,'\',files_loc(i).name)
      ;
```

```
18
19 X = imread(filename);
20
21 % Wavelet Decomposiition
22 [C,S] = wavedec2(X,3,wname);
23
24 for j = 1 : 3
25 A{j} = appcoef2(C,S,wname,j);
26 [H{j}, V{j},D{j}] = detcoef2('all',C,S,j);
27 end
28
29 % Extract features
30 coef(i,1) = mean2(H{1});
31 coef(i,2) = mean2(V{1});
32 coef(i,3) = mean2(D{1});
33 coef(i,4) = mean2(H{2});
34 coef(i,5) = mean2(V{2});
35 coef(i,6) = mean2(D{2});
36 coef(i,7) = mean2(H{3});
37 coef(i,8) = mean2(V{3});
38 coef(i,9) = mean2(D{3});
39
40 % Apply label
41 if(strcmp(files, 'Benign'))
42 coef(i,end) = 1;
43 elseif(strcmp(files, 'Cancer'))
44 coef(i,end) = 2;
45 elseif(strcmp(files, 'Normal'))
46 coef(i,end) = 3;
47 end
48
49 % Get estimated time to finish
50 time_elapsed = etime(clock, time_start);
51 if (time_elapsed - time_elapsed_at_last_print >
      time_between_each_print)
52 time_elapsed_at_last_print = time_elapsed;
53 estimated_time_left = time_elapsed * (length(files_loc)-i) / i;
54 fprintf('%d / %d : estimated time left %d seconds\n',...
55 i,length(files_loc),floor(estimated_time_left));
56 end
57
58 end
59
```

```
% Save location
location = strcat('Cropped Details\',files,'\', wname, '.mat');

save(location, 'coef', '-v7.3');
clear;
end
```

D.3 Data Shuffling

```
function shuffle_the_data(input)
data_only = input;

shuffle_innitial=randperm(size(data_only,1));
shuffle_data=data_only(shuffle_innitial,:);
Measure=shuffle_data(1:end,1:end-1);
final_target=shuffle_data(:,end);

save data_nahid_1  -v7.3;
end
```

D.4 SVM Training

```
clc;
close all;
load data_nahid_1;

% Proportion of data as training data
prop = 0.25;
data_percentage=round(length(Measure)*prop);

% Separate trainig data and labels
inputs=Measure(1:data_percentage,:);
Target=final_target(1:data_percentage,:);

% Show percentage labels used for training data
[n,p] = size(inputs)
isLabels = unique(Target);
nLabels = numel(isLabels)
tabulate(categorical(Target));

% Options for parallel computing
options = statset('UseParallel',1);
% Create SVM template
```

```
22 tSVM = templateSVM('Standardize',1,'KernelFunction','gaussian');
23
24 % Train SVM
25 svmStruct = fitcecoc(inputs,Target,'Learners',tSVM,'Options',...
26 options,'Verbose',2);
27
28 outputs = predict(svmStruct,inputs,'Options',options);
29
30 % Figure of predicted and correct labels
31 k = 1:length(Target);
32 figure(1)
33 plot(k,Target,'+r')
34 hold on
35 plot(k,outputs,'og')
36 nm = 0;
37 for l=1: length(Target)
38 if Target(l) == outputs(l)
39 nm = nm+1;
40 end
41 end
42
43 % Print accuracy rate through manual cross validation
44 correct_rate = nm/length(Target)*100
45
46 % Confusion plotting
47 % Convert the integer label vector to a class-identifier matrix.
48 [~,grpOOF] = ismember(outputs,isLabels);
49 oofLabelMat = zeros(nLabels,n);
50 idxLinear = sub2ind([nLabels n],grpOOF,(1:n)');
51 oofLabelMat(idxLinear) = 1; % Flags the row corresponding to the
     class
52 [~,grpY] = ismember(Target,isLabels);
53 YMat = zeros(nLabels,n);
54 idxLinearY = sub2ind([nLabels n],grpY',(1:n));
55 YMat(idxLinearY) = 1;
56
57 figure(2);
58 plotconfusion(YMat,oofLabelMat);
59 h = gca;
60
61 [c,cm] = confusion(YMat,oofLabelMat);
62
63 % Print manual classification accuracy
```

```
fprintf('Percentage Correct Classification   : %f%%\n', 100*(1-c
    ));
fprintf('Percentage Incorrect Classification : %f%%\n', 100*c);

figure(3)
plotroc(YMat,oofLabelMat,'+r');
grid on;

save data_nahid_1_train -v7.3;
```

D.5 Testing SVM

```
clc;
close all;
load data_nahid_1_train;

% Clear previous inputs and labels
clear inputs;
clear Target;

% Separate testing data and labels
inputs=Measure(data_percentage:end,:);
Target=final_target(data_percentage:end,:);
% inputs=Measure;
% Target=final_target;

% Show percentage labels used for training data
[n,p] = size(inputs)
isLabels = unique(Target);
nLabels = numel(isLabels)
tabulate(categorical(Target));

%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%PCA Calculation
% [PCAA1, PCAA2, PCAA3]= pca(inputs);
% inputs=PCAA2;
%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%
```

```
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Use libsvm for predicting instead
% outputs=svmpredict(Target',inputs,svmStruct);
%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Options for parallel computing
options = statset('UseParallel',1);

% Manual Cross validation
%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

outputs = predict(svmStruct,inputs,'Options',options);

% Figure of predicted and correct labels
k = 1:length(Target);
figure(1)
plot(k,Target,'+r')
hold on
plot(k,outputs,'og')
nm = 0;
for l=1: length(Target)
if Target(l) == outputs(l)
nm = nm+1;
end
end

% Print accuracy rate through manual cross validation
correct_rate = nm/length(Target)*100
%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Confusion plotting
% Convert the integer label vector to a class-identifier matrix.
[~,grpOOF] = ismember(outputs,isLabels);
oofLabelMat = zeros(nLabels,n);
idxLinear = sub2ind([nLabels n],grpOOF,(1:n)');
oofLabelMat(idxLinear) = 1; % Flags the row corresponding to the
    class
```

```
62 [~,grpY] = ismember(Target,isLabels);
63 YMat = zeros(nLabels,n);
64 idxLinearY = sub2ind([nLabels n],grpY',(1:n));
65 YMat(idxLinearY) = 1;
66
67 figure(2);
68 plotconfusion(YMat,oofLabelMat);
69 h = gca;
70
71 [c,cm] = confusion(YMat,oofLabelMat);
72
73 % Print manual classification accuracy
74 fprintf('Percentage Correct Classification   : %f%%\n', 100*(1-c
     ));
75 fprintf('Percentage Incorrect Classification : %f%%\n', 100*c);
76
77 figure(3)
78 plotroc(YMat,oofLabelMat,'+r');
79 grid on;
80
81 save data_nahid_1_test  -v7.3;
```

Appendix E

Scattering Transform Code

```
%% load the database
clear; close all;

path_to_db = 'D:\Cropped Images';
db_name = 'BC';

use_precomputed_scattering = 1; % change to 0 to skip
    computation of scattering
precomputed_path = sprintf('./precomputed/%s/src.mat', db_name);
if (use_precomputed_scattering)
load(precomputed_path);
else
src = kthtips_src(path_to_db);
save(precomputed_path, 'src');
end

size_of_files = numel(src.files);

for i = 1 : 19
grid_train(i) = 0.05*i;
end

% grid_train = [5, 10, 20, 40, 80, 160, 320, 640, 1280,...
%     2560, 5120]; % number of training for classification
nb_split = 2; % number of split for classification

%% ------------------------------------------------------
%% ------------------- trans_scatt ----------------------
%% ------------------------------------------------------

```

```
30
31 %% compute scattering of all images in the database
32 feature_name = 'trans_scatt';
33 precomputed_path = sprintf('./precomputed/%s/%s.mat', db_name,
       feature_name);
34 if (use_precomputed_scattering)
35 load(precomputed_path);
36 else
37 %configure scattering
38 filt_opt = struct();
39 scat_opt.J = 5; % number of octaves
40
41 % build the wavelet transform operators for scattering
42 Wop = wavelet_factory_2d_pyramid(filt_opt, scat_opt);
43
44 % a function handle that
45 %   - read the image
46 %   - resize it to 200x200
47 %   - compute its scattering
48 fun = @(filename)(scat(imresize_notoolbox(imreadBW(filename)...
49 ,[200 200]), Wop));
50
51 % compute all scattering
52 trans_scatt_all = srcfun(fun, src);
53 %%
54 %   - format the scattering in a 3d matrix
55 % a function handle that
56 %   - remove order 0
57 %   - remove margins along dimension 2 and 3
58 %   - average accross position
59 fun = @(Sx)(mean(mean(remove_margin(format_scat(Sx)
       ,[1,0,1,1,1,1]),2),3));
60 trans_scatt = cellfun_monitor(fun,trans_scatt_all);
61 save(precomputed_path, 'trans_scatt');
62 end
63
64 % format the database of feature
65 db = cellsrc2db(trans_scatt, src);
66
67 %% classification
68 rsds_classif(db, db_name, 'trans_scatt', grid_train, nb_split);
```

Haar						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	91.79	56.48	95.00	60.30	93.91	64.30
10%	86.49	59.96	91.68	63.08	92.63	66.20
15%	86.20	61.45	88.57	64.66	89.95	65.96
20%	85.46	62.61	86.59	65.51	86.61	67.01
25%	84.03	64.04	84.49	66.20	86.47	67.83
30%	82.72	64.00	84.57	67.55	85.94	68.00
35%	82.17	65.45	84.06	67.64	85.39	67.69
40%	82.34	65.31	83.52	68.63	84.84	68.07
45%	81.14	65.15	82.91	69.17	84.50	68.41
50%	81.33	65.62	82.72	68.64	84.54	68.61
55%	80.46	65.69	83.44	68.64	84.29	68.97
60%	80.35	66.13	83.49	68.68	83.94	69.22
65%	80.01	66.42	83.05	68.43	83.84	68.96
70%	79.47	66.14	82.59	68.00	83.69	69.18
75%	79.05	65.73	82.22	68.20	83.86	69.62
80%	78.93	66.10	82.42	68.80	83.28	69.55
85%	78.86	65.40	82.31	68.85	82.89	69.34
90%	78.71	67.89	81.65	68.60	82.68	69.76
95%	78.76	64.62	81.62	66.78	82.38	69.33
Avg	82.20	64.42	84.74	67.20	85.74	68.15
Avg 50% train	79.83	66.03	82.68	68.60	83.75	69.16
Max	91.79	67.89	95.00	69.17	93.91	69.76
Min	78.71	56.48	81.65	60.30	82.68	64.30

Figure F.1: SVM Haar approximation results

Haar						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	86.80	64.95	89.67	70.25	89.42	63.19
10%	84.43	66.76	86.69	74.33	85.90	65.35
15%	84.44	67.18	86.79	76.64	85.99	66.48
20%	81.72	67.63	85.51	77.14	85.16	67.47
25%	80.56	68.00	85.42	76.79	85.05	67.25
30%	79.98	67.96	85.68	77.94	84.50	67.84
35%	79.15	69.65	85.44	78.08	84.24	67.83
40%	78.71	69.47	85.19	78.31	83.40	68.88
45%	77.81	69.76	85.46	78.55	83.57	68.93
50%	78.07	69.55	85.19	78.43	83.39	69.51
55%	77.52	69.99	85.08	79.07	82.94	69.18
60%	77.69	70.90	85.13	78.74	83.03	68.94
65%	77.19	71.07	85.15	79.55	82.68	69.41
70%	77.14	71.33	84.97	79.31	82.34	69.66
75%	77.19	72.18	85.15	79.77	82.38	69.62
80%	77.34	72.12	84.75	79.78	82.22	70.51
85%	77.51	72.14	84.78	79.93	82.21	70.51
90%	77.36	70.82	84.61	80.07	81.88	72.80
95%	77.16	69.30	84.42	80.07	81.67	75.72
Avg	79.48	69.53	85.59	77.93	83.91	68.52
Avg 50% train	77.48	70.99	85.03	79.32	82.66	69.91
Max	86.80	72.18	89.67	80.07	89.42	72.80
Min	77.14	64.95	84.61	70.25	81.88	63.19

Figure F.2: SVM Haar detail results

db2						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	94.72	62.68	96.00	60.30	96.15	64.37
10%	90.01	66.42	94.01	65.99	94.23	68.04
15%	88.26	66.39	92.79	67.28	92.51	68.52
20%	87.89	67.27	91.42	69.52	90.46	69.48
25%	87.02	68.60	89.95	70.57	90.12	69.90
30%	86.34	68.40	88.79	71.02	88.83	70.59
35%	85.07	68.74	88.53	71.61	89.88	70.89
40%	84.62	69.20	87.93	71.90	89.09	71.07
45%	85.12	68.80	89.16	70.98	87.70	72.25
50%	84.59	68.88	88.22	71.37	87.88	72.11
55%	84.65	70.03	87.80	71.78	87.55	71.82
60%	84.46	70.52	87.40	72.85	86.95	72.16
65%	84.46	70.52	87.40	72.85	86.95	72.16
70%	83.54	69.77	87.33	72.63	86.31	72.04
75%	83.79	72.77	87.26	72.92	86.47	72.31
80%	83.54	69.77	87.33	72.63	86.31	72.04
85%	83.37	69.70	87.23	73.50	86.00	72.22
90%	83.31	68.18	87.14	73.75	85.78	73.12
95%	83.40	71.05	87.03	75.75	85.60	75.40
Avg	85.82	68.70	89.21	70.75	88.84	70.84
Avg 50% train	84.08	69.89	87.63	72.53	86.79	72.22
Max	94.72	72.77	96.00	73.75	96.15	73.12
Min	83.31	62.68	87.14	60.30	85.78	64.37

Figure F.3: SVM db2 approximation results

db2						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	90.03	59.59	89.67	66.47	89.74	65.49
10%	87.67	61.75	87.69	72.39	88.94	67.29
15%	86.99	63.99	87.24	73.82	88.45	68.05
20%	87.37	60.20	88.34	74.58	87.97	68.26
25%	87.02	59.23	88.08	74.90	87.30	68.92
30%	84.63	60.54	86.79	75.89	86.58	69.51
35%	83.98	61.77	87.01	76.06	86.72	68.92
40%	83.99	62.43	86.97	75.51	86.01	69.36
45%	82.81	66.03	86.46	75.37	85.99	69.40
50%	82.09	66.82	86.92	75.10	85.57	69.67
55%	81.71	67.03	86.29	75.26	85.19	69.58
60%	81.26	67.89	86.18	76.12	85.03	69.94
65%	80.87	67.88	85.79	76.84	84.38	70.15
70%	80.94	67.07	85.59	76.37	84.60	69.93
75%	80.50	68.72	85.37	76.45	84.35	70.45
80%	80.38	67.42	85.19	77.54	84.00	71.15
85%	80.27	66.76	85.23	77.16	83.91	70.62
90%	79.98	65.98	84.79	76.91	83.73	72.32
95%	80.05	64.33	84.44	79.40	83.39	75.40
Avg	83.47	64.51	86.64	75.15	86.03	69.39
Avg 50% train	81.08	67.16	85.78	76.31	84.68	70.32
Max	90.03	68.72	89.67	77.54	89.74	72.32
Min	79.98	59.23	84.79	66.47	83.73	65.49

Figure F.4: SVM db2 detail results

db3						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	94.43	61.15	95.67	61.05	96.79	63.46
10%	88.99	64.61	94.18	66.19	93.27	67.36
15%	87.48	66.68	92.01	67.24	91.76	68.75
20%	86.93	67.41	91.59	69.63	90.46	68.98
25%	86.79	68.55	90.01	70.66	89.74	69.84
30%	86.20	68.76	89.35	70.92	88.99	70.18
35%	85.78	68.97	89.11	71.07	89.10	70.74
40%	85.06	68.91	88.51	71.90	88.93	71.49
45%	84.73	68.86	88.64	72.16	87.92	72.19
50%	84.35	68.61	88.81	71.94	88.10	72.04
55%	84.65	69.08	87.98	72.49	87.81	71.61
60%	84.22	69.54	88.32	72.84	87.12	72.67
65%	83.67	69.27	87.99	72.47	86.92	72.30
70%	83.66	69.81	87.87	73.88	86.85	71.69
75%	83.89	68.90	87.52	74.05	86.62	72.37
80%	83.13	68.31	87.37	73.46	86.51	72.12
85%	83.02	68.62	87.00	74.17	85.95	72.44
90%	83.10	67.74	87.25	74.75	85.87	74.40
95%	83.06	70.47	86.98	76.08	85.65	76.36
Avg	85.56	67.99	89.40	71.16	88.82	70.81
Avg 50% train	83.84	68.87	87.88	73.22	86.97	72.38
Max	94.43	69.81	95.67	74.75	96.79	74.40
Min	83.02	61.15	87.00	61.05	85.87	63.46

Figure F.5: SVM db3 approximation results

db3						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	89.44	58.38	90.00	70.76	87.18	67.12
10%	85.32	61.47	87.69	73.79	86.06	68.89
15%	84.64	62.56	88.12	74.41	85.45	69.62
20%	84.43	63.14	87.18	76.24	86.21	69.98
25%	83.15	64.30	87.35	76.76	85.76	69.84
30%	82.87	64.29	86.63	77.01	84.93	70.29
35%	83.05	65.04	86.82	76.88	84.79	70.15
40%	81.83	64.55	86.23	76.59	83.80	70.32
45%	80.98	64.80	85.61	77.03	84.11	70.80
50%	80.80	64.33	85.65	76.76	83.87	71.02
55%	80.35	64.94	85.62	77.29	83.41	71.18
60%	80.23	64.88	85.38	78.12	83.78	70.90
65%	80.12	65.53	85.20	78.79	83.02	71.15
70%	79.97	64.97	85.04	78.15	83.05	70.73
75%	79.44	65.43	84.77	78.11	82.94	71.22
80%	78.73	65.44	84.75	79.20	82.80	71.07
85%	78.60	65.98	84.92	78.94	82.51	71.05
90%	78.61	63.20	84.87	79.73	82.32	71.84
95%	78.50	60.53	84.51	80.07	81.99	76.36
Avg	81.81	64.07	86.21	76.92	84.22	70.40
Avg 50% train	79.78	64.95	85.18	78.21	83.18	71.10
Max	89.44	65.98	90.00	79.73	87.18	71.84
Min	78.60	58.38	84.75	70.76	82.32	67.12

Figure F.6: SVM db3 detail results

db6						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	95.89	61.44	97.33	61.26	97.12	63.76
10%	91.19	63.32	94.34	66.32	94.07	67.77
15%	89.33	66.51	92.45	66.59	92.30	69.03
20%	88.77	67.10	91.26	69.11	90.78	69.38
25%	88.20	68.25	90.61	70.95	90.51	70.01
30%	87.18	68.32	89.62	70.49	89.95	70.16
35%	87.16	69.60	89.63	71.22	89.51	70.42
40%	86.56	69.81	89.43	72.48	88.85	71.28
45%	86.53	69.39	89.72	71.98	88.13	71.58
50%	85.94	70.38	89.15	71.87	88.65	72.14
55%	86.15	71.17	88.80	72.26	88.28	71.64
60%	86.10	71.38	88.82	72.71	87.89	72.51
65%	85.84	71.82	89.09	72.85	87.81	72.07
70%	85.72	71.67	89.01	73.60	87.45	71.85
75%	85.85	71.48	88.50	74.05	87.06	71.73
80%	85.37	69.99	88.47	74.96	87.01	72.84
85%	84.82	72.14	88.33	74.06	86.55	72.97
90%	84.76	71.41	88.03	75.08	86.49	74.56
95%	84.76	72.51	87.58	75.75	86.24	77.32
Avg	87.30	69.18	90.14	71.21	89.36	70.87
Avg 50% train	85.71	71.08	88.79	73.34	87.53	72.39
Max	95.89	72.14	97.33	75.08	97.12	74.56
Min	84.76	61.44	88.03	61.26	86.49	63.76

Figure F.7: SVM db6 approximation results

db6						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	83.58	58.63	89.00	71.93	88.78	68.30
10%	82.53	61.44	86.52	73.76	85.58	69.55
15%	83.46	61.07	84.68	74.78	85.78	69.86
20%	82.09	62.06	85.68	74.54	85.40	69.60
25%	81.97	62.48	86.75	74.99	84.80	69.54
30%	80.86	63.20	86.46	74.85	84.87	70.18
35%	81.04	63.01	86.01	75.27	84.05	69.86
40%	80.76	63.28	85.31	75.59	83.64	70.08
45%	80.75	64.10	85.24	76.07	83.68	69.98
50%	79.54	64.89	85.05	76.30	83.35	70.44
55%	79.02	65.10	84.87	77.00	83.38	70.72
60%	78.37	65.32	84.77	76.91	83.06	70.42
65%	77.82	64.86	84.48	76.89	82.80	70.65
70%	77.62	66.10	84.26	76.59	82.47	70.83
75%	77.96	66.02	84.42	76.38	82.25	71.73
80%	77.90	66.03	84.44	77.12	81.72	72.60
85%	78.08	64.13	84.12	77.61	81.95	71.69
90%	78.11	62.90	84.13	78.74	81.77	73.60
95%	77.65	66.96	84.16	78.41	81.70	74.76
Avg	80.08	63.59	85.34	75.85	83.85	70.53
Avg 50% train	78.52	64.95	84.58	76.96	82.64	71.27
Max	83.58	66.10	89.00	78.74	88.78	73.60
Min	77.62	58.63	84.12	71.93	81.72	68.30

Figure F.8: SVM db6 detail results

db8						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	95.89	61.81	97.67	61.42	97.12	63.63
10%	92.07	63.37	94.84	66.10	93.91	67.56
15%	89.92	66.23	93.12	66.83	92.51	68.90
20%	89.13	67.39	91.59	68.42	91.10	69.16
25%	89.20	68.29	91.01	70.64	90.96	70.07
30%	88.06	68.17	90.34	70.64	89.58	70.22
35%	88.00	69.53	90.25	71.35	89.46	70.60
40%	87.22	70.13	89.68	72.15	89.01	71.12
45%	87.11	69.55	89.86	71.68	88.81	71.67
50%	86.52	70.05	89.18	71.80	89.10	71.85
55%	86.71	71.10	89.47	72.49	88.78	71.89
60%	86.67	71.49	89.57	72.09	88.43	72.14
65%	86.72	71.49	89.53	72.28	88.26	71.47
70%	86.35	71.82	88.70	73.16	87.93	71.53
75%	86.34	71.19	88.41	74.18	87.51	71.35
80%	86.09	70.65	88.58	74.04	87.25	71.96
85%	85.51	71.85	88.56	73.73	87.00	72.65
90%	85.68	71.26	88.01	74.75	86.78	74.24
95%	85.33	71.05	87.96	75.08	86.58	75.40
Avg	87.95	69.19	90.47	70.99	89.64	70.67
Avg 50% train	86.37	71.04	88.99	73.02	87.98	72.07
Max	95.89	71.85	97.67	74.75	97.12	74.24
Min	85.51	61.81	88.01	61.42	86.78	63.63

Figure F.9: SVM db8 approximation results

db8						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	87.68	58.32	90.33	68.20	90.06	65.22
10%	88.11	60.19	89.02	72.81	87.66	67.61
15%	87.38	61.07	88.35	74.54	88.66	67.86
20%	85.61	61.25	87.59	74.91	88.77	68.48
25%	83.62	61.58	87.62	75.43	88.01	68.85
30%	83.11	62.51	87.18	75.75	86.91	69.22
35%	82.38	63.12	87.16	75.83	86.81	68.89
40%	81.75	62.65	87.27	75.89	86.41	68.88
45%	81.50	62.72	87.20	75.67	86.21	69.08
50%	81.59	63.30	87.12	75.47	85.95	69.51
55%	81.10	63.47	87.05	76.00	85.25	69.83
60%	81.01	64.29	86.79	76.87	85.09	69.78
65%	80.71	63.94	86.71	77.22	84.78	70.05
70%	80.85	64.48	86.83	77.20	84.70	69.66
75%	80.52	65.38	86.61	77.64	84.56	69.87
80%	80.33	65.52	86.35	78.29	84.35	70.35
85%	79.94	65.10	86.37	77.16	84.21	69.87
90%	79.85	63.93	86.16	77.74	84.00	71.84
95%	79.52	61.99	85.91	77.41	83.73	75.72
Avg	82.61	62.93	87.32	75.70	86.24	69.16
Avg 50% train	80.74	64.21	86.72	76.93	84.91	69.98
Max	88.11	65.52	90.33	78.29	90.06	71.84
Min	79.85	58.32	86.16	68.20	84.00	65.22

Figure F.10: SVM db8 detail results

Appendix F

SVM Results

F.1 Haar results

F.1.1 Approximation Features

F.1.2 Detail Features

F.2 db2 results

F.2.1 Approximation Features

F.2.2 Detail Features

F.3 db3 results

F.3.1 Approximation Features

F.3.2 Detail Features

F.4 db6 results

F.4.1 Approximation Features

F.4.2 Detail Features

F.5 db8 results

F.5.1 Approximation Features

F.5.2 Detail Features

F.6 db9 results

F.6.1 Approximation Features

F.6.2 Detail Features

F.7 db10 results

F.7.1 Approximation Features

db9						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	95.60	63.17	98.00	61.21	97.44	63.24
10%	92.51	63.19	95.17	66.12	94.23	67.27
15%	90.80	66.06	93.12	67.01	92.94	68.90
20%	89.06	67.19	92.01	68.63	91.34	69.38
25%	89.02	68.53	91.61	70.44	90.64	70.22
30%	88.06	68.34	90.07	70.64	89.95	70.04
35%	88.00	69.38	90.44	71.27	89.37	70.79
40%	87.67	69.91	89.97	71.90	88.97	71.36
45%	87.24	69.28	89.86	71.86	88.95	71.61
50%	86.64	70.02	89.21	72.07	89.29	71.82
55%	87.05	70.84	89.35	72.19	89.15	71.86
60%	86.96	71.27	89.51	72.71	88.51	72.02
65%	86.97	71.28	89.42	72.33	88.43	71.70
70%	86.60	71.62	89.06	72.93	88.09	71.74
75%	86.51	70.42	88.70	73.65	87.75	71.73
80%	86.27	70.51	88.76	74.13	87.41	72.68
85%	85.61	71.65	88.54	73.95	87.21	73.29
90%	85.79	71.70	88.07	73.92	87.05	71.04
95%	85.67	71.93	88.05	75.75	86.77	76.68
Avg	88.13	69.13	90.60	70.94	89.82	70.59
Avg 50% train	86.56	70.86	89.05	72.97	88.18	71.95
Max	95.60	71.70	98.00	74.13	97.44	73.29
Min	85.61	63.17	88.07	61.21	87.05	63.24

Figure F.11: SVM db9 approximation results

db9						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	89.15	54.49	94.00	69.11	86.86	66.82
10%	88.25	55.88	86.69	73.83	88.62	67.72
15%	86.50	58.60	87.79	74.47	87.27	68.60
20%	85.54	58.29	86.34	75.35	87.65	68.88
25%	85.20	59.93	86.82	75.68	87.11	68.98
30%	83.75	59.38	86.46	75.82	86.05	69.38
35%	83.31	59.35	86.68	76.11	86.30	69.22
40%	81.94	60.59	86.56	76.89	85.53	69.52
45%	81.17	61.12	86.35	76.49	86.32	69.66
50%	80.04	60.45	86.65	76.30	85.92	70.09
55%	80.41	60.60	85.93	76.37	84.96	69.97
60%	80.25	61.80	85.54	77.45	84.77	70.18
65%	79.72	61.93	85.43	78.08	84.16	71.43
70%	79.82	61.84	85.64	76.98	84.24	70.78
75%	79.42	62.32	85.35	77.45	83.86	69.49
80%	78.66	61.92	85.16	77.79	83.90	71.55
85%	78.32	61.09	85.21	77.72	83.59	71.15
90%	77.68	60.56	85.05	77.91	83.71	71.36
95%	77.98	59.06	84.93	78.41	83.59	75.72
Avg	82.17	60.01	86.54	76.10	85.60	69.71
Avg 50% train	79.55	61.36	85.63	77.25	84.54	70.57
Max	89.15	62.32	94.00	78.08	88.62	71.55
Min	77.68	54.49	85.05	69.11	83.59	66.82

Figure F.12: SVM db9 detail results

db10						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	95.60	63.17	97.67	61.02	96.79	63.04
10%	92.51	63.20	95.67	66.36	94.39	67.56
15%	90.80	66.06	93.34	67.14	92.94	68.69
20%	89.50	67.27	92.67	68.53	91.42	69.06
25%	89.25	68.51	91.61	70.75	91.28	70.03
30%	88.11	68.04	90.73	70.42	90.11	70.50
35%	87.88	69.24	90.77	71.02	90.01	71.04
40%	87.74	70.11	90.47	72.09	89.25	71.65
45%	87.44	69.71	90.09	72.07	89.31	71.76
50%	87.23	70.08	89.85	72.37	89.42	72.01
55%	87.00	70.91	89.44	72.97	89.36	71.93
60%	86.96	71.49	89.84	73.04	88.86	72.51
65%	87.01	71.32	89.81	71.99	88.63	72.02
70%	86.60	71.77	89.39	73.54	88.25	71.58
75%	86.77	70.89	88.92	74.92	87.86	71.92
80%	86.40	70.51	88.85	74.71	87.53	72.84
85%	85.82	71.26	88.90	74.61	87.47	72.44
90%	85.74	71.41	88.59	75.08	87.03	75.04
95%	85.84	71.35	88.31	76.08	86.88	76.68
Avg	88.24	69.16	90.92	71.26	89.99	70.87
Avg 50% train	86.70	70.93	89.37	73.53	88.37	72.40
Max	95.60	71.77	97.67	75.08	96.79	75.04
Min	85.74	63.17	88.59	61.02	87.03	63.04

Figure F.13: SVM db10 approximation results

db10						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	88.56	58.37	92.00	70.29	83.33	67.19
10%	85.76	61.38	89.35	74.13	83.65	69.23
15%	84.15	62.12	88.46	75.74	83.10	70.50
20%	82.45	61.87	87.26	87.26	81.80	71.32
25%	82.09	62.01	87.02	75.43	82.42	71.29
30%	81.35	62.89	86.79	75.39	82.04	71.05
35%	80.12	63.10	86.63	75.40	83.19	71.19
40%	79.96	62.84	85.81	75.59	82.92	70.77
45%	80.00	63.06	85.53	75.22	83.00	71.44
50%	79.48	63.74	85.69	76.03	82.71	71.11
55%	79.44	63.76	85.38	76.48	82.57	71.71
60%	79.13	64.51	85.32	76.12	82.66	71.70
65%	78.90	64.03	84.84	76.32	82.61	71.38
70%	78.31	64.38	84.92	76.48	82.68	71.26
75%	77.88	65.38	84.68	76.65	82.70	71.35
80%	77.46	66.03	84.56	76.96	82.56	71.55
85%	77.58	63.93	84.76	76.16	82.21	70.73
90%	77.83	63.64	84.44	76.91	82.34	71.68
95%	77.88	62.87	84.18	79.40	82.16	71.25
Avg	80.58	63.17	86.30	76.25	82.69	70.91
Avg 50% train	78.60	64.25	85.01	76.33	82.60	71.39
Max	88.56	66.03	92.00	87.26	83.65	71.71
Min	77.46	58.37	84.44	70.29	81.80	67.19

Figure F.14: SVM db10 detail results

bior3.7						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	96.77	66.74	97.67	61.05	97.76	65.13
10%	92.95	70.71	96.17	68.98	95.67	68.59
15%	92.27	72.04	93.90	70.00	92.94	69.07
20%	90.46	73.06	93.51	71.48	91.90	69.46
25%	89.72	73.44	92.01	72.90	91.21	71.14
30%	89.77	73.83	91.23	73.89	90.11	71.71
35%	89.01	74.53	91.01	74.24	89.97	71.98
40%	89.02	75.29	90.39	75.17	89.37	72.00
45%	88.87	74.83	90.42	74.92	89.13	73.36
50%	88.17	75.04	90.21	75.83	88.90	73.13
55%	87.83	74.89	89.92	76.78	88.43	72.78
60%	87.77	75.01	90.34	77.16	88.03	73.43
65%	87.46	75.68	90.14	76.80	87.86	73.58
70%	86.87	75.83	90.01	76.65	87.72	73.77
75%	86.77	75.65	89.77	77.58	87.64	73.53
80%	86.22	74.91	89.78	77.87	87.17	74.52
85%	86.49	75.76	89.56	78.16	86.87	74.89
90%	86.08	76.10	89.31	77.41	86.92	75.36
95%	86.01	76.02	89.07	79.07	86.73	77.00
Avg	89.03	74.07	91.41	74.27	89.87	72.08
Avg 50% train	87.25	75.37	89.95	76.91	87.87	73.84
Max	96.77	76.10	97.67	78.16	97.76	75.36
Min	86.08	66.74	89.31	61.05	86.87	65.13

Figure F.15: SVM bior3.7 approximation results

bior3.7						
	BC		BN		CN	
Proportion	Validate	Testing	Validate	Testing	Validate	Testing
5%	91.20	58.58	93.33	66.12	87.82	65.30
10%	87.52	59.81	89.02	71.72	87.34	67.08
15%	86.01	60.50	88.79	73.33	87.70	67.88
20%	83.85	60.75	88.93	74.20	87.17	68.56
25%	82.91	60.91	88.68	75.03	87.24	69.24
30%	81.99	61.23	88.18	75.42	86.80	69.45
35%	81.54	61.72	88.30	75.22	86.81	68.92
40%	80.65	61.82	87.93	75.51	86.25	69.20
45%	80.78	62.48	88.05	75.40	85.92	68.90
50%	80.92	62.13	87.88	74.80	85.41	69.48
55%	80.11	62.33	87.62	75.81	85.45	69.08
60%	80.13	63.01	87.38	76.62	85.52	69.74
65%	79.31	63.23	86.84	76.41	85.12	70.74
70%	79.55	63.45	86.66	76.43	85.04	70.73
75%	79.23	63.20	86.59	76.58	84.76	70.90
80%	78.67	62.95	86.18	77.29	84.51	71.96
85%	78.84	62.85	86.25	77.61	84.12	71.05
90%	79.04	60.85	86.02	78.90	84.16	71.68
95%	78.79	60.82	85.93	79.07	83.79	75.40
Avg	81.63	61.72	87.82	75.34	85.84	69.75
Avg 50% train	79.46	62.48	86.73	76.95	84.79	71.07
Max	91.20	63.45	93.33	79.07	87.82	75.40
Min	78.67	58.58	85.93	66.12	83.79	65.30

Figure F.16: SVM bior3.7 detail results

Translation invarient scattering			
Training size	BC	BN	CN
5%	65.09	66.44	65.84
10%	67.58	69.95	67.99
15%	68.93	70.92	68.56
20%	69.40	71.86	68.97
25%	69.94	72.18	69.17
30%	70.48	72.71	69.40
35%	71.08	73.47	69.62
40%	71.60	73.20	70.14
45%	71.47	73.67	69.56
50%	71.73	74.44	69.64
55%	71.53	74.22	70.31
60%	71.69	73.83	69.69
65%	71.98	74.80	69.93
70%	72.40	74.41	69.99
75%	72.75	75.32	69.98
80%	71.86	75.18	70.93
85%	72.64	75.72	70.86
90%	73.14	74.94	69.55
95%	72.32	75.78	69.07
Avg	70.93	73.32	69.43
Avg 50% train	72.20	74.86	70.00
Max	73.14	75.78	70.93
Min	65.09	66.44	65.84

Figure F.17: SVM Scattering results

Bibliography

[1] 16. learning: Support vector machines. [Online]. Available: https://www.youtube.com/watch?v=_PwhiWxHK8o

[2] *ScatNet : a MATLAB Toolbox for Scattering Networks.*

[3] "Support vector machine (and statistical learning theory) tutorial." [Online]. Available: www.cs.columbia.edu/~kathy/cs4701/documents/jason_svm_tutorial.pdf

[4] A. Ben-hur and J. Weston, "Chapter 13 a users guide to support vector machines."

[5] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1872–1886, Aug. 2013. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2012.230

[6] C. J. Burges, "A tutorial on support vector machines for pattern recognition," vol. 2, January 1998, pp. 121–167. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-tutorial-on-support-vector-machines-for-pattern-recognition/

[7] L. Chung-lin, "A tutorial of the wavelet transform," 2010.

[8] H. N. Finsberg, "Wavelet techniques in medical imaging: Classification of ultrasound images using the windowed scattering transform."

[9] C. Jeyashree.K, "Detection of breast cancer using continuous wavelet transform and support vector machine," *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, vol. 5, pp. 790–795, March 2016.

[10] P.-Y. Lin, "An introduction to wavelet transform."

[11] S. I. Niwas, P. Palanisamy, and K. Sujathan, "Wavelet based feature extraction method for breast cancer cytology images," in *Industrial Electronics Applications (ISIEA), 2010 IEEE Symposium on*, Oct 2010, pp. 686–690.

[12] W. H. Organization, *World Cancer Report 2014.*

[13] O. N. U. Perlin Gorgel, Ahmet Sertbas, "Feature extraction based wavelet transform in breast cancer diagnosis using fuzzy and non-fuzzy classification," *International Journal of Electronics, Mechanical and Mechatronics Engineering*, vol. 2, no. 4, pp. 237–333.

[14] Y. I. A. Rejani and S. T. Selvi, "Early detection of breast cancer using SVM classifier technique," *CoRR*, vol. abs/0912.2314, 2009. [Online]. Available: http://arxiv.org/abs/0912.2314

[15] L. Sifre and S. Mallat, "Rotation, scaling and deformation invariant scattering for texture discrimination," in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 1233–1240. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2013.163

[16] A. Tirtajaya and D. D. Santika, "Classification of microcalcification using dual-tree complex wavelet transform and support vector machine," in *Advances in Computing, Control and Telecommunication Technologies (ACT), 2010 Second International Conference on*, Dec 2010, pp. 164–166.

[17] C. Valens, "A really friendly guide to wavelets," 1999.

[18] J. Wan and S. Zhou, "Features extraction based on wavelet packet transform for b-mode ultrasound liver images," in *Image and Signal Processing (CISP), 2010 3rd International Congress on*, vol. 2, Oct 2010, pp. 949–955.