Politechnika Wrocławska

Faculty of Pure and Applied Mathematics

Field of study: Applied Mathematics
Specialty: Computational Mathematics

Master Thesis

APPLICATIONS OF WAVELETS IN DATA MINING

Kinga Kurowska

keywords:
wavelets
statistics
learning and adaptive systems

Short summary:
** *To be changed* **
The aim of this thesis is an analysis and comparison of statistical methods employing wavelets. Special attention will be paid to data mining and machine learning methods.

Supervisor	dr inż. Andrzej Giniewicz		
	Title, degree, name and surname	*Grade*	*Signature*

*For the purposes of archival thesis qualified to: **

a) Category A (perpetual files)

b) Category BE 50 (subject to expertise after 50 years)

** Delete as appropriate*

stamp of the faculty

Wrocław, 2018

Contents

Introduction

*** Few words about my thesis... Generally what is a data mining, what are the wavelets, what are the applications in data mining and finally, what is the main purpose - edge detection. ***

Nowadays, people gather a huge amount of, often unstructured, data. High--dimensionality, missing values or noisiness are only some of the problems that we have to face in data mining. Wavelets can be really helpful in this area, thanks to their properties. First of all, there exist efficient algorithms for wavelet transform. Low computational complexity is a big advantage in case of data mining analysis.

Pre-processing:

- Denoising signals and images. The Wavelet noise reduction - transform data into wavelet domain, remove noise components (with lower frequency) and then back to the original domain.
- Dimensionality reduction. The idea is to simplify data, by getting rid off the less relevant information. In a wavelet domain we retain only the largest coefficients. Then, after getting back to the original domain we obtain simplified data.

Machine learning processing:

- Clustering. Low frequency parts are correlated with regions of objects concentration and the high frequency parts correspond to the areas with sudden changes in the objects distribution. Thus, clustering can be conducted by recognising correlated components in the wavelet domain.
- Classification. Wavelet-base algorithm in machine learning (2-D DWT) developed by Castelli 1996. Notable faster than the classic one.
- Regression. Uses to predict future values based on historical data. Wavelet approach is the non-parametric regression. Similar to the dimensionality reduction case.
- Distributed Data Mining. Thanks to the orthogonality, data mining tasks can be carried out on smaller subsets of data independently and than results can be combined.

Chapter 1

Wavelets theory

A "wavelet" literally means a small wave. This term says a lot about its nature. Wavelets are a family of functions which oscillates like wave and should be compactly supported.

Definition 1.1. A mother wavelet is a function $\psi(t)$, which satisfies the condition that the set of functions

$$\left\{\psi(2^j t - m),\ j, m \in \mathbb{Z}\right\} \tag{1.1}$$

is an orthonormal basis of $\mathbf{L}^2(\mathbb{R})$.

Definition 1.2. Wavelet functions are created by scaling and shifting of the mother wavelet $\psi(t)$. They are defined as

$$\psi^{(a,b)}(t) = |a|^{-\frac{1}{2}} \psi\left(\frac{t-b}{a}\right),\ a > 0, \tag{1.2}$$

where a is a scale parameter and b translation parameter.

There are plenty of different mother wavelets, for example

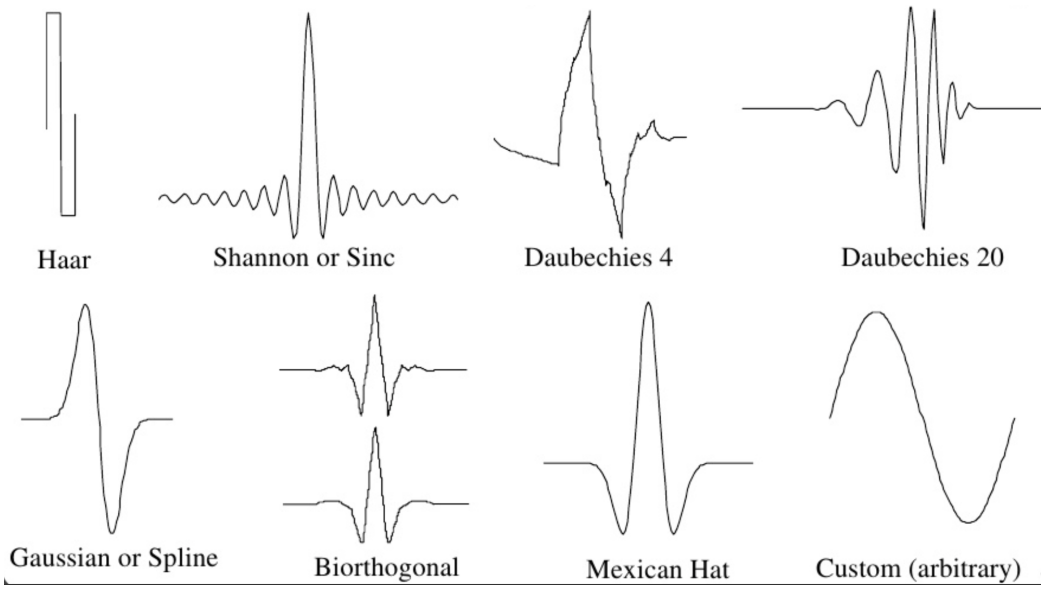


Figure 1.1: Different types of wavelets.

The wavelet functions have specific properties which are useful in data analysis. According to the [1] article, the most important are:

- Vanishing moments. Wavelet has n vanishing moments, when $\int t^j \psi(t) dt = 0$ for $j = 0, 1, \ldots, n-1$. For all wavelets at least the first moment, the mean value, is equal zero. Particular wavelets have more vanishing moments, e.g. Daubechies family, the $Daub4$ and $Daub20$ presented on the figure 1.1 has respectively, 2 and 10 vanishing moments. This property is also known as approximation order, because for the part of data which are represented as n-degree polynomials, the wavelet coefficients are also equal zero (according to the definition 1.4). So the bigger n, the more data is neglected.
- Compact support. Wavelets domain is a compact set, most of the time it is a finite interval. This property protects from leaving a data region during wavelet processing.
- Decorrelated coefficients. Wavelet transformations diminish the time correlation, thus the wavelet coefficients are less depended than the analysed data.

1.1. Haar and Daubechies wavelets

Each type of wavelet function is more suitable for different applications. As it is shown in [6], the Daubechies wavelets are very useful in compression, denoising and enhancement audio signals or images.

Definition 1.3. Daubechies wavelets are collection of orthogonal and compactly supported functions. A denotations for those wavelets are dbn or $DaubN$, where n is the number of vanishing moments and $N = 2n$. Moreover, the support in on the interval $[0, 2n-1]$.

The Haar wavelet is the simplest of Daubechies ($db1$), and also of all, wavelets. The mother function is illustrated on the graph 1.2 and has the following formula

$$\psi(t) = \begin{cases} 1, & \text{for } 0 \leq t < \frac{1}{2}, \\ -1, & \text{for } \frac{1}{2} \leq t < 1, \\ 0, & \text{otherwise.} \end{cases} \tag{1.3}$$

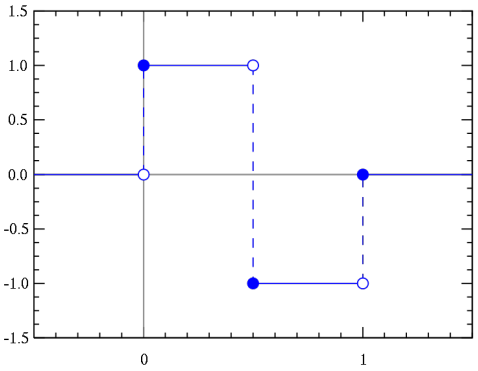


Figure 1.2: The Haar wavelet.

Thanks to this simple form the Haar wavelet has wide application in signal and image analysis. However, the simplest wavelet is not always enough. There are cases which require more complex wavelets. Examples of those Daubechies wavelets are shown on the figure 1.3. We can see that the bigger n, the function is smoother and more regular.

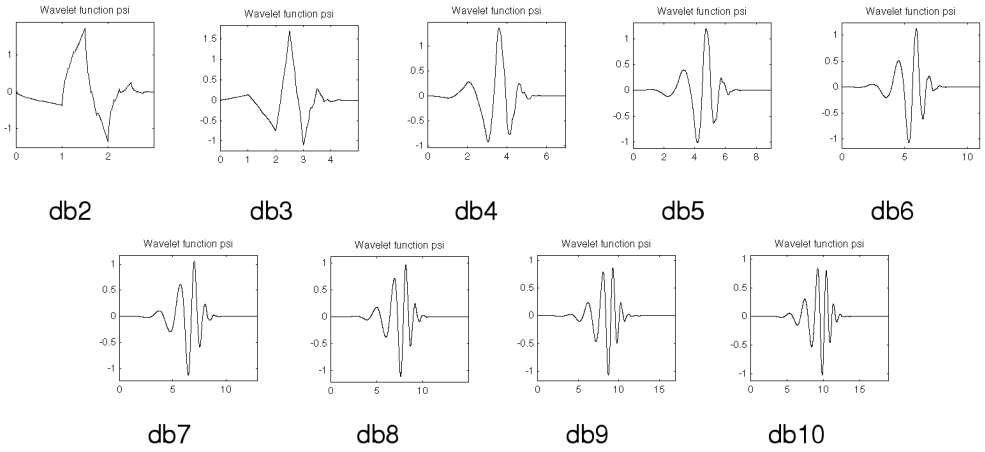


Figure 1.3: Daubechies wavelets.

1.2. Wavelet Transform

Generally, integral transforms are useful in data processing. The idea is to convert data into another domain, where it is easier to manipulate them, finding desired informations, etc. Finally, the results are transformed back to the original domain by inverse integral transform.
We can define integral transform based on the wavelet functions - Wavelet Transform. However, the most popular kind of integral transform is Fourier Transform. Here the question arises, what are the differences between both operators. Lets start from the definitions.

Definition 1.4. Continuous Wavelet Transform is expressed by the formula

$$W(a,b) = \int_{-\infty}^{\infty} y(t) a^{-\frac{1}{2}} \psi\left(\frac{t-b}{a}\right) dt, \tag{1.4}$$

where a is scale parameter, b translation parameter and $y(t)$ original signal.

Definition 1.5. Fourier transform is defined as

$$Y(f) = \int_{-\infty}^{\infty} y(t) e^{-i\omega t} dt, \tag{1.5}$$

where $y(t)$ is time domain signal and $Y(f)$ is frequency domain signal.

The most important differences are presented in the table below.
The obvious distinction between both transforms is the type of function. In Fourier case there are sine and cosine functions, wherein wavelet transform uses wavelets. Sine function oscillates on the whole real axis, thus it cannot

Wavelet transform	**Fourier transform**
Suitable for stationary and non--stationary signals	Suitable for stationary signals
High time and frequency resolution	Zero time resolution and very high frequency resolution
Very suitable for studying the local behaviours of the signal	No suitable
Scaled and translated mother wavelets	Sine and cosine waves

Table 1.1: The comparison of Wavelet and Fourier Transform.

represent abrupt changes. However, the Wavelet transform is localized in space and time, so it can be used to detect trends or sudden changes in signals and images. Moreover, wide range of wavelet functions is a main advantage of wavelet analysis, because we can adjust the type of wavelet to the specific case and thank to that obtain more accurate results.

1.3. Discrete Wavelet transform

There are two types of Wavelet Transform:
- Continuous Wavelet Transform (CWT),
- Discrete Wavelet Transform (DWT).

The Discrete Wavelet Transform has a wide range of applications in denoising and compressing signals and images.
*** More info about applications in pre-processing and machine learning is currently in the Introduction. I'm not sure where to finally put it ***
Scale and translation parameters are defined as

$$a = 2^j \text{ and } b = 2^j m, \ j, m = 1, 2, \ldots. \tag{1.6}$$

Therefore, assuming that $0 \leq t < 2^K$, $K \in \mathbb{Z}$, the coefficients of Discrete Wavelet Transform are defined as follow

$$d^{(j,m)} = \sum_{t=0}^{2^K-1} y(t) 2^{-\frac{j}{2}} \psi\left(2^{-j}t - m\right). \tag{1.7}$$

The figure 1.4 on a page 11 shows how DWT works. Discrete Wavelet Transform splits signal with two filters: $g(n)$ - low pass filter (LPF) and $h(n)$ - high pass filter (HPF). The LPF captures a part with lower frequencies which refers to the main signal. Whereas, the HPF captures higher frequencies - a noise of the signal. Subsequently, both parts are downsampled by a factor of 2. This decomposition can be repeated on the LPF part of the signal. Hence, the next levels of DWT coefficients.

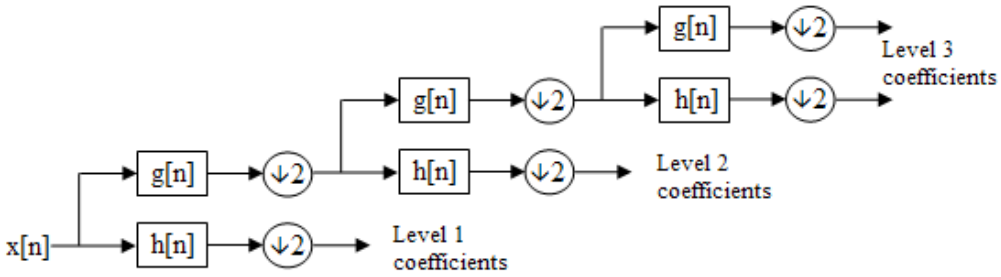


Figure 1.4: Discrete Wavelet transform on a signal $x(n)$.

1.4. 2-D Discrete Wavelet Transform

2-Dimensional Discrete Wavelet Transform works similar way as 1-D with High Pass Filter, Low Pass Filter and downsampling, except that one level of the decomposition includes double filtering, on columns and rows. The figure 1.5 shows an image decomposition. Firstly, the DWT is applied on columns of the input image and then on the rows of the both outputs. Ultimately, there are four results:

- LL - result of LPF applied on both, columns and rows,
- LH - result of LPF applied on columns and HPF on rows,
- HL - result of HPF applied on columns and LPF on rows,
- HH - result of HPF applied on both, columns and rows.

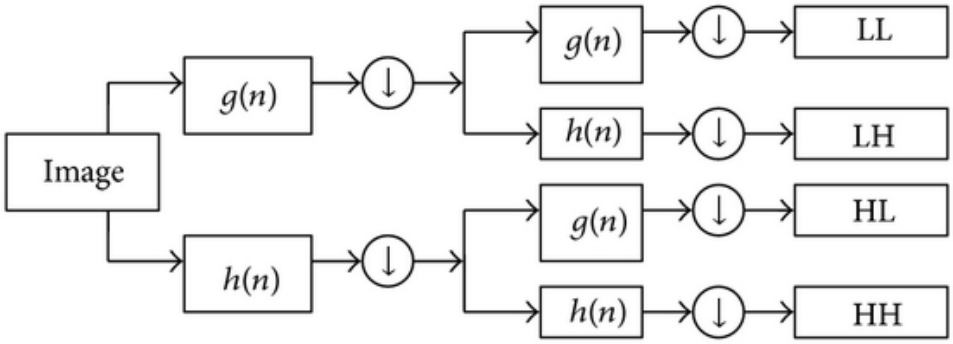


Figure 1.5: 2-D Discrete Wavelet transform on an image.

Recall that the outcome of Low Pass Filter in the previous case was the main signal (without a noise). Thus, a 2-Dimensional equivalent is an approximation of an analysed image. The High Pass Filter captures high frequencies, then for an image the outcome are sudden changes in the image contrast. Now, lets focus on what exactly each result represent. First one, the LL is just an approximation of the initial image. Next, the LH shows abrupt changes in a horizontal direction, whereas the HL part presents similar issues but in a vertical direction. The HH shows sudden changes in a diagonal way.

In conclusion, the output of 2-D DWT gives us an approximation of the image and three parts with abrupt changes in different directions.
The 2-D DWT can be really useful in compressing and denoising images. It can compress pictures better than the JPEG, often without changes visible by the human eye. This transform is applicable in other image processing, like edge enhancement, edge detection and shape recognition.

1.5. Inverse Wavelet Transform

Majority applications of Wavelet Transform assume that data is converted to wavelet domain and then we can receive the desired information. Although, we must return to the original domain for the information to be useful. The Inverse Wavelet Transform allows for such operation.

Definition 1.6. Inverse Discrete Wavelet Transform is defined as

$$y(t) = \sum_{j=1}^{K} \sum_{m=0}^{2^{K-j}-1} d^{(j,m)} 2^{-\frac{j}{2}} \psi\left(2^{-j}t - m\right) + \psi_0, \tag{1.8}$$

where $y(t)$ is reconstruction of the input data and $d^{(j,m)}$ coefficients of wavelet transform. The ψ_0 is an average value of $y(t)$ over $t \in [0, 2^K - 1]$. This parameter can be approximated by zero, without loss of generality.

Summarizing, now we are able to transform data to a wavelet domain. Process them simply and efficiently, as well as, we can back to the original domain to read the obtained results.

Chapter 2

Edge detection

In this chapter let us focus on the main goal of the thesis - edge detection. This problem is a special case of the data dimensionality reduction, which is a significant part of the pre-processing in data mining.
There are various methods to identify such discontinuities. The most popular are gradient based (e.g. Canny, Prewitt, Sobel) and Laplacian based. However, there is also another method, which provides similar results and can be more efficient in terms of computation. This method is based on the 2-Dimensional Discrete Wavelet Transform described in section 1.4.

The mentioned method contains the following steps. *** Link to article Edge detection ***
1. Convert image to grey scale.
2. Apply 2-D DWT on an image.
3. Remove the LL part.
4. Denoise the LH, HL and HH coefficients.
5. Reconstruct the initial image.
6. Post-processing - modify contrast to emphasize obtained edges.

2.1. Implementation of an algorithm

The algorithm is implemented in `Python` using `PyWavelets` package. There are used also auxiliary packages like: `numpy`, `matplotlib`, `PIL` and `scipy`. The `PyWavelets` package contains all features required to edge detection algorithm, i.e. 2D Forward and Inverse Discrete Wavelet Transform, build-in many wavelet functions and thresholding functionality (used to denoise coefficients).
Lets go deeper into algorithm, using a simple example - white square on a black background.
Initially, a colour image must be simplified by conversion to grey scale. Edges are recognised as changes in brightness, so a single pixel should contains only information about a colour (black) intensity. Our initial image is already black and white so we do not need to convert it. Now, we can apply the 2D DWT function. As a result, according to the description in section 1.4, we obtain four components of wavelet coefficients: LL, LH, HL and HH shown on a graph 2.2. It is clearly visible that the LL part reflects the initial image and the rest components contains information about rapid brightness changes in particular directions.

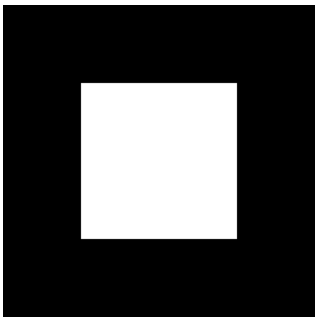


Figure 2.1: An initial image - a white square.

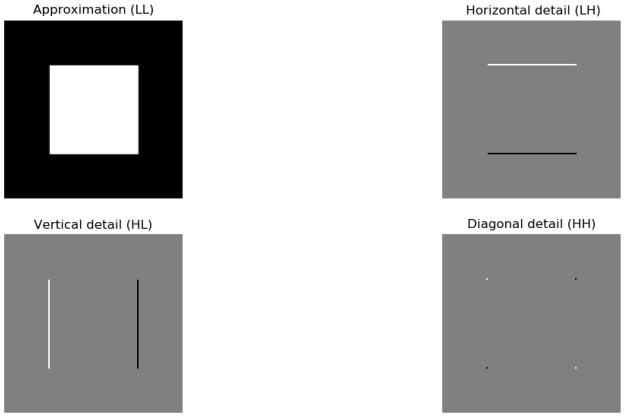


Figure 2.2: 2-D DWT coefficients.

Thus, we are interested only in components which gives information about edges, so we remove the approximation part - figure 2.3.
Subsequently, remaining components can be denoised. It means, we can get rid off small, insignificant coefficients by thresholding. More about setting the threshold is described in section 2.2. This simple example do not require any denoising, so lets go further.
The last main step is reconstruction of the initial image, i.e. application an inverse DWT on the denoised coefficients. In the result, we obtain edges of the initial image presented on a graph 2.4.
At the end, we can manipulate contrast to emphasize obtained lines. Currently, a background has grey colour and the edges are white or black. Therefore, using simple mathematical calculations we can modify image to have black background and white edges. It is enough to get an absolute value, subtract 128 and then scale by multiplying 2 times.
*** Calculations taken from article Edge detection. Add note about values in an image array ***
Finally, we obtain the edges of the square shown on the figure 2.5.

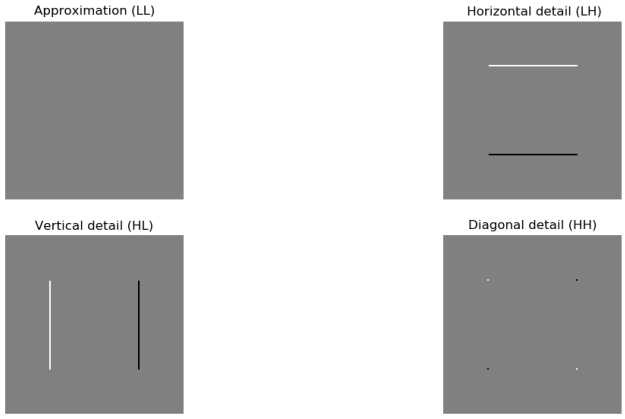


Figure 2.3: 2-D DWT coefficients with removed the LL part.

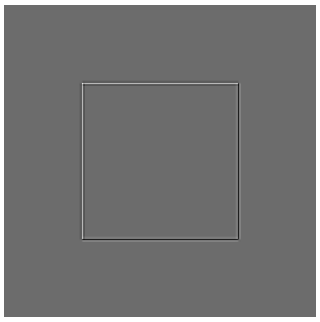


Figure 2.4: A reconstructed image showing the edges.

2.2. Thresholding

The wavelets property of vanishing moments guarantee that bigger wavelet coefficients are more significant. Thus, the noise is placed in lowest values. To remove those values we can use, so-called, thresholding, which has two types: hard and soft. Lets denote λ as threshold value and d as wavelet coefficient. The hard thresholding is defined as follow

$$D^H(d|\lambda) = \begin{cases} 0, & \text{for } |d| \leq \lambda, \\ d, & \text{for } |d| > \lambda. \end{cases} \tag{2.1}$$

It assigns zero value for coefficients below the set threshold. The soft thresholding works in the same way on the coefficients smaller than λ, but additionally the coefficient bigger than the set threshold are "shrinked" towards zero, as it is defined below.

$$D^S(d|\lambda) = \begin{cases} 0, & \text{for } |d| \leq \lambda, \\ d - \lambda, & \text{for } d > \lambda, \\ d + \lambda, & \text{for } d < -\lambda. \end{cases} \tag{2.2}$$

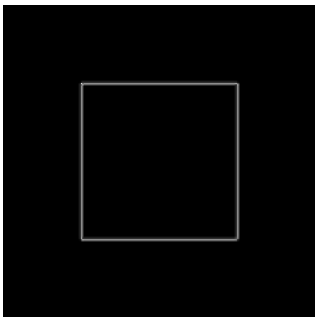


Figure 2.5: The edges of the square image after post-processing.

2.2.1. Hard or soft thresholding

To see the differences between both types of thresholding lets consider a noisy image (fig. 2.6).

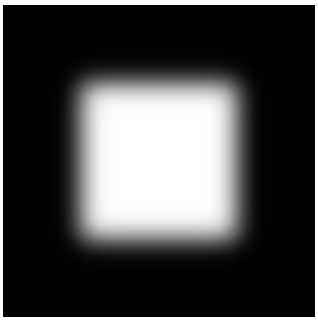


Figure 2.6: A noisy square image.

The coefficients of 2-D DWT are shown on the graph 2.7.
Now, according to the edge detection algorithm, we remove the LL component and we can denoise the other ones. Results of hard and soft thresholding are presented respectively on a figures 2.8 and 2.9.
We can see the distinction between both thresholding, especially on the LH and HL components. After soft one the lines are smoother and slightly narrower, then in the other case. Therefore, the soft thresholding seems to be better for edge detection. Because of the smoother results, it is especially useful for the real life images, where edges are often blurred. To confirm this conclusion lets compare the final results, after inverse DWT (fig. 2.10)

2.2.2. Setting the threshold

The corollary from the previous subsection is that the soft thresholding provides better results for edge detection. However, there is also a question how to choose the value of a threshold λ to denoise coefficients and do not loose any significant information. A simple idea is to get a quantile of the coefficients in the specific components.
We decided to start with a quantile on level 0.95. It occurs to be accurate for the majority of standard images. All the earlier results are computed for threshold set as 0.95 quantile. Although, images which are noisier could require changing the quantile level.

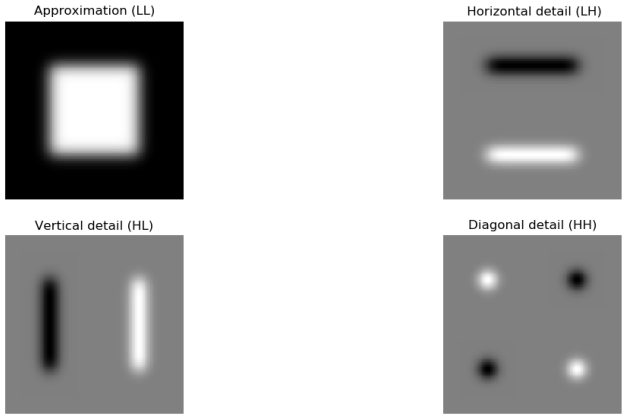


Figure 2.7: 2-D DWT coefficients.

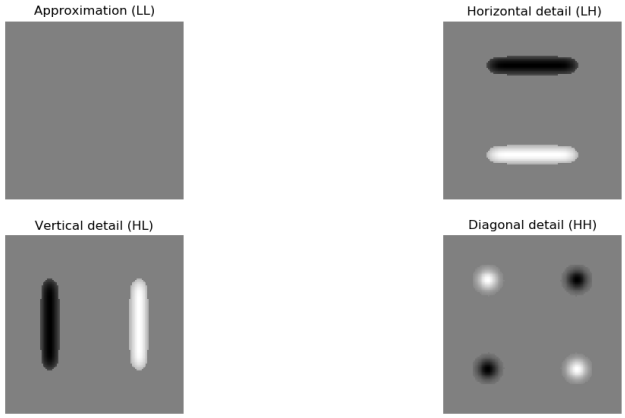


Figure 2.8: 2-D DWT coefficients after hard thresholding.

Lets consider three images of basic diamond shape with different noisiness σ, shown on the figure 2.11.
At first, see how looks the result with mentioned earlier threshold set as 0.95 quantile (fig. 2.12). Algorithm found the edges properly but they are quite wide, especially for the noisier image. Lets try to increase the threshold to quantile on the 0.99 level. The results are presented on the graph 2.13. It can be seen that for (b) and (c) images the edges are narrower and visible. Nevertheless, result for the (a) image is incorrect, threshold is too high and too much information is removed.
Hence, lets consider one more threshold on a 0.98 quantile level. The recognised edges for (a) and (b) images are presented on the figure 2.14. For this threshold edges are correctly detected.
In conclusion, for noisier images increasing the threshold provides more accurate result, but it must be done carefully to not ignore crucial informations. Additionally, when image shows one sharp object and noisy background, by

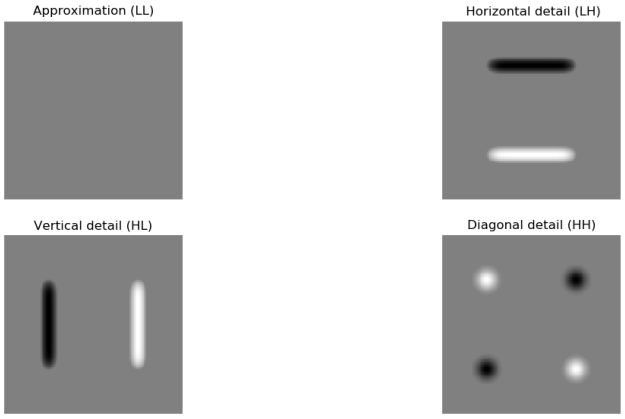
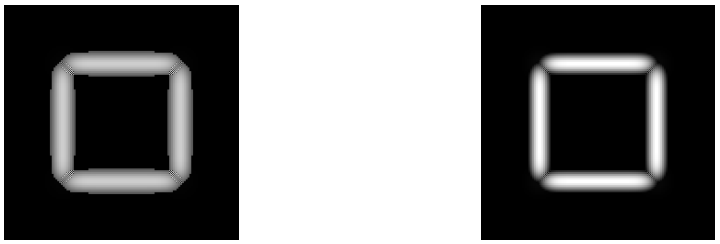


Figure 2.9: 2-D DWT coefficients after soft thresholding.



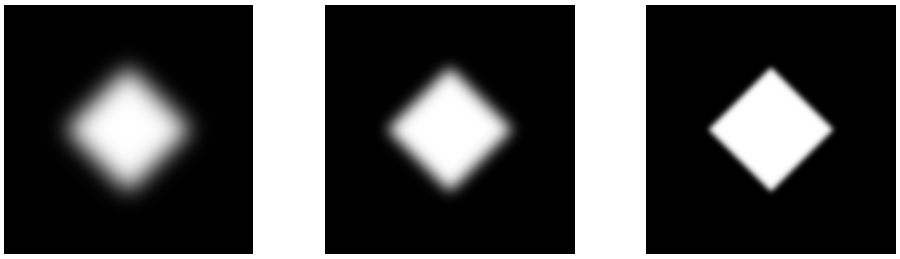
(a) hard (b) soft

Figure 2.10: Results of edge detection with hard and soft thresholding.

increasing the threshold we can control the amount of background details in the result.

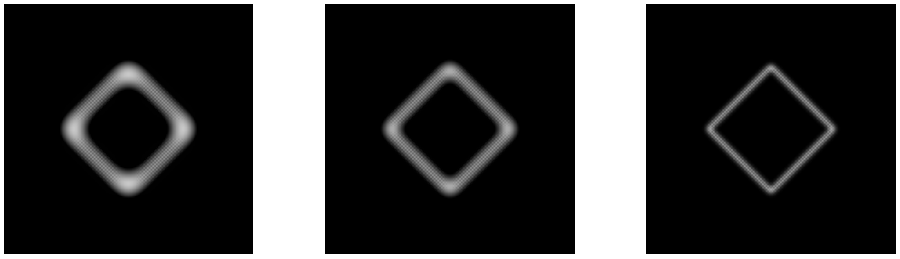
2.3. Selection of a wavelet type

As it was mentioned in the chapter 1, the family of Daubechies wavelets are very suitable to image processing. Therefore, we carry out the edge detection using this kind of wavelets. However, each of *dbn* wavelet has slightly different properties, mainly because of changing the n value, the number of vanishing moments.
Lets consider which of the Daubechies wavelets, are the best for edge detection problem. Remark that, all of the analysis below are conducted with the soft threshold set as 95 quantile. The figure 2.15 (a) presents an example image in grey scale, the man with a camera. Firstly, we take the simplest one, Haar ($db1$) wavelet. Looking at the detected edges on the figure 2.15 (b), we can clearly observe boundaries of main objects - the man and a camera. Not every edge is recognised, e.g. a hand of the man, because of a small difference in brightness.
Next, lets consider Daubechies wavelets with the higher n. The results for selected n are presented on the figure 2.16. They provide worse edge ap-



(a) $\sigma = 16$ (b) $\sigma = 10$ (c) $\sigma = 4$

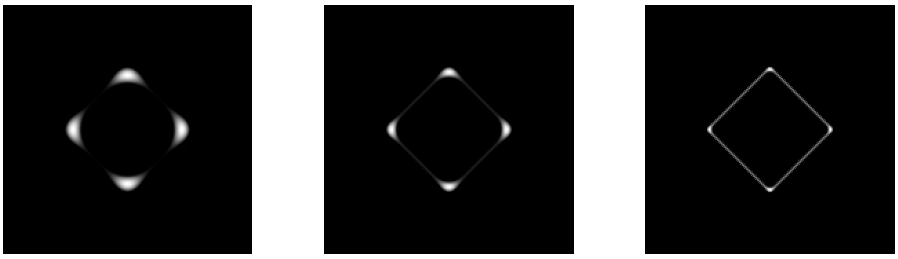
Figure 2.11: The initial images of diamond with different noise level.



(a) $\sigma = 16$ (b) $\sigma = 10$ (c) $\sigma = 4$

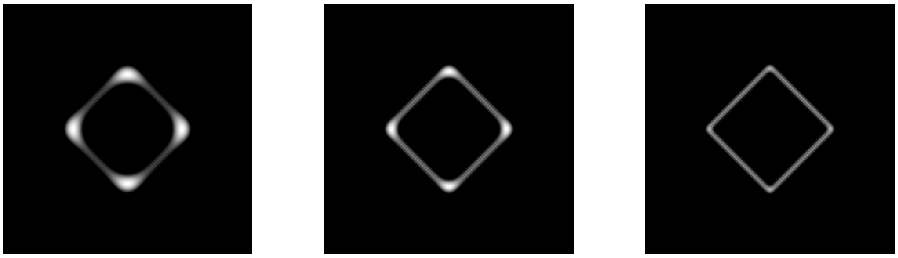
Figure 2.12: The recognized edges for the diamonds with λ equals 0.95 quantile.

proximation than the Haar transform. There are small differences between specific *dbn* but no regularity can be seen.
Based on the example above, the conclusion can be made that the Haar wavelet provides the best results in edge detection. Unfortunately, this simplest wavelet not always works properly. For the basic square image, the Haar transform does not recognise any edges (fig. 2.17 (b)). Hence, we need to apply Daubechies wavelet with the bigger n value. For the $n = 2$ or more, the square's perimeter is correctly detected and the results for different $n = 2, 3, \ldots$ are very similar to each other (fig. 2.17). The whole edge detection analysis presented in section 2.1 was made with *db*2 wavelets.
Lets consider the third example, a noisy square (fig. 2.18). In this case only the *db*1 (Haar) provides proper result - the square's perimeter. The edge is quite wide, but this feature can be improved by thresholding described in section 2.2. The rest of presented results (c) - (f) are incorrect. Therefore, we revealed that the Haar wavelet is also the most appropriate for recognizing edges on noisy images.
The last example is a photo of a frog presented on the figure 2.19. Incidentally, this photo was made in The Wroclaw Botanical Garden. Back to the results, it can be seen that for Haar wavelet (b) edges are the clearest and sharpest. On the other graphs, (c) and (d), the contour of the frog is also visible, but it is more blurred. Moreover, the bigger n, the edges are more unclear.
To sum up, for the majority of images the Haar wavelet provides the best



(a) $\sigma = 16$ (b) $\sigma = 10$ (c) $\sigma = 4$

Figure 2.13: The recognized edges for the diamonds with λ equals 0.99 quantile.



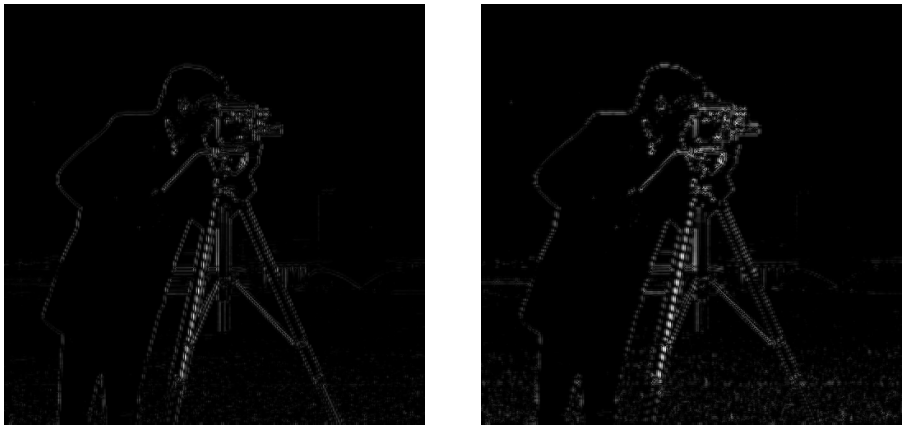
(a) $\sigma = 16$ (b) $\sigma = 10$ (c) $\sigma = 4$

Figure 2.14: The recognized edges for the diamonds with λ equals 0.98 quantile.

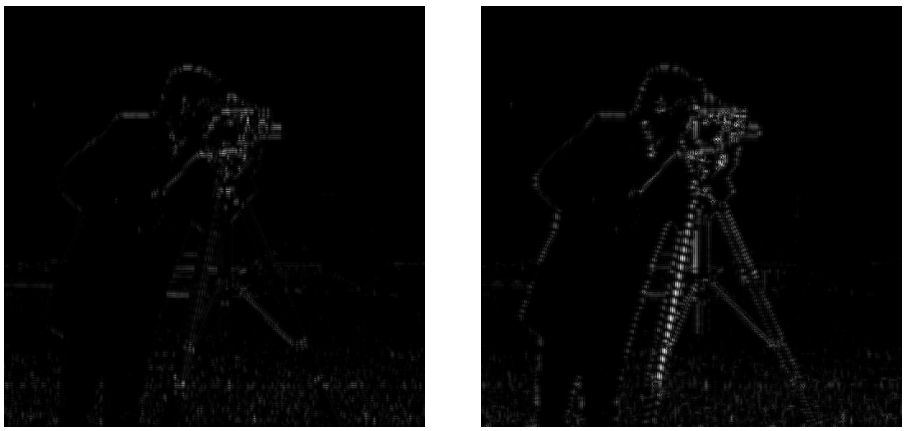
results. However, there are cases, when the *db*1 wavelet fails. Then, it is required to use another of Daubechies wavelet, e.g. *db*2.

(a) original (b) *db*1

Figure 2.15: A man with a camera and detected edges using Haar wavelet.

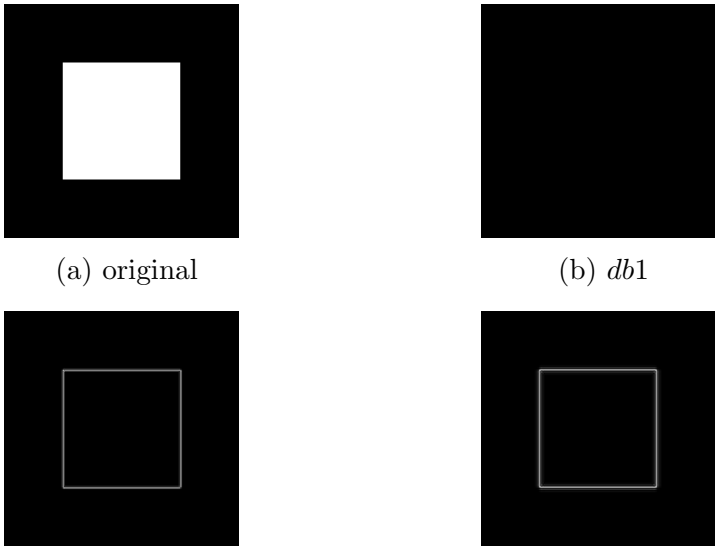


(a) *db*2 (b) *db*4



(c) *db*6 (d) *db*8

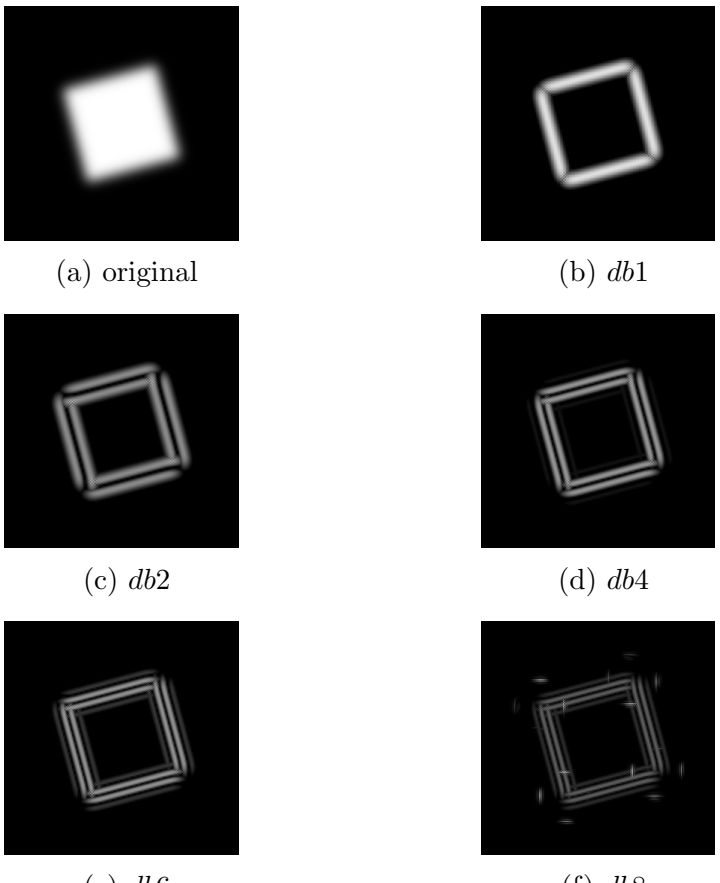
Figure 2.16: A man with a camera - detected edges using selected Daubechies wavelets.



(a) original (b) *db*1

(c) *db*2 (d) *db*4

Figure 2.17: A simple square - detected edges using selected Daubechies wavelets.



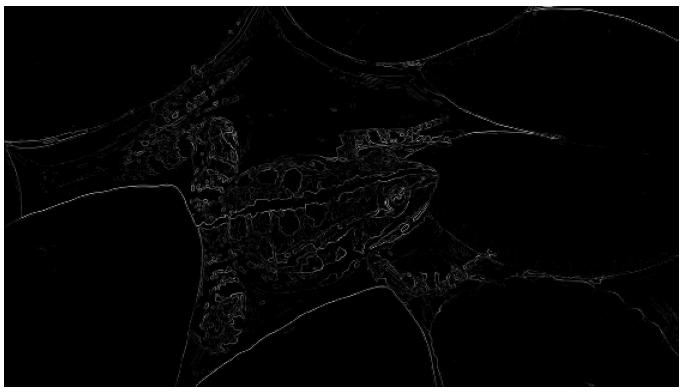
(a) original (b) *db*1

(c) *db*2 (d) *db*4

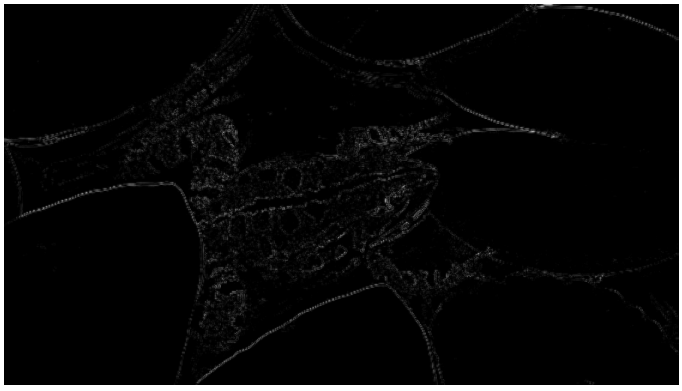
(e) *db*6 (f) *db*8

Figure 2.18: A noisy rotated square - detected edges using selected Daubechies wavelets.

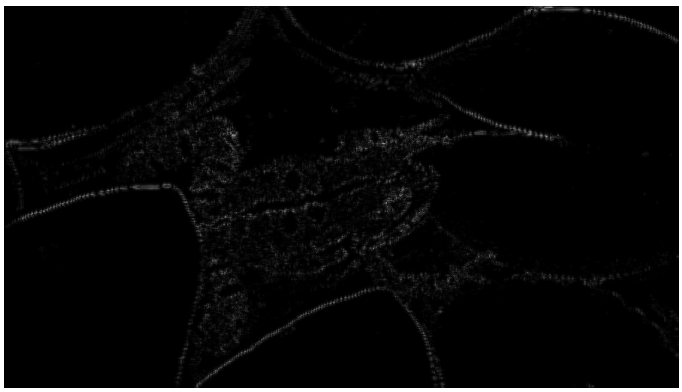
(a) original



(b) *db*1



(c) *db*2



(d) *db*6

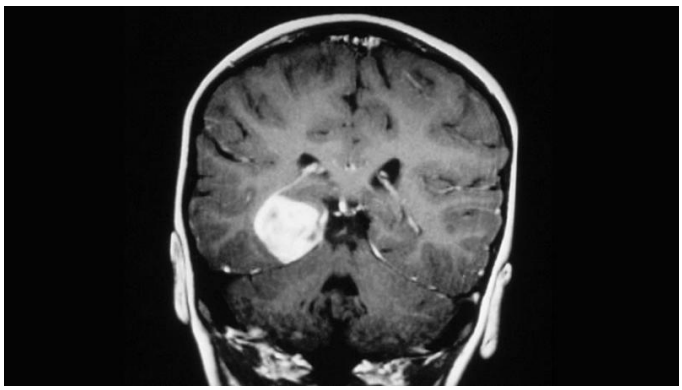
Figure 2.19: A frog - detected edges using selected Daubechies wavelets.

Chapter 3

Application in medicine

The usage of DWT in biomedical images is described in the article [5], mainly classification problem.

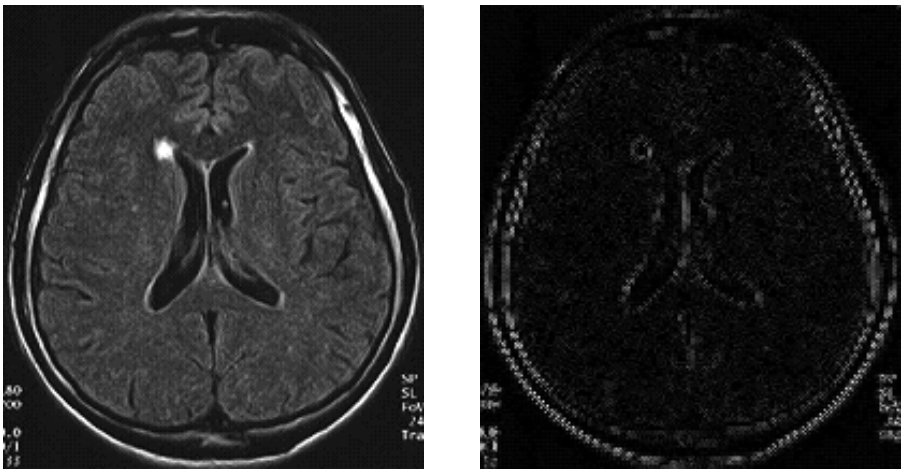
*** I am not sure if my result has any value but I put them below. Maybe I'll add some description that also edge detection can be useful in pre-processing before carrying out the classification algorithms ***



(a) original

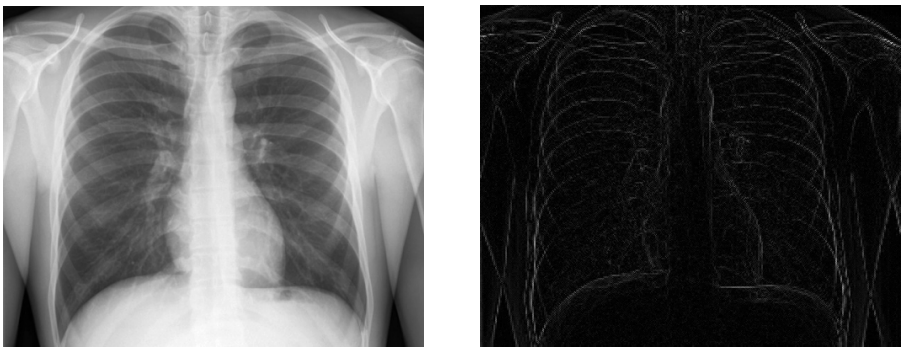
(b) detected edges

Figure 3.1: A brain MRI - tumour.



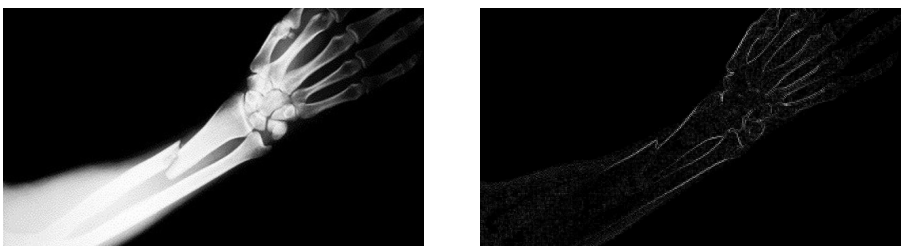
(a) original (b) detected edges

Figure 3.2: A brain MRI - small tumour.



(a) original (b) detected edges

Figure 3.3: Lungs.



(a) original (b) detected edges

Figure 3.4: Broken bone - arm.

(a) original (b) detected edges

Figure 3.5: Broken bone - leg.

Chapter 4

Summary

*** Wavelets can be really useful in data mining tasks. Edge detection algorithm was developed, as an example of usage wavelets in dimensionality reduction in pre-processing of machine learning. There are parameters like threshold and type of wavelet, which enable flexibility of the algorithm. The application in analysing biomedical images ***

List of Figures

Bibliography

[1] T. Li, S. Ma, M. Ogihara. Wavelet methods in data mining. *Data Mining and Knowledge Discovery Handbook*, pages 603–626, 2005.

[2] L. Zhang, W. Zhou, L. Jiao. Wavelet support vector machine. *IEEE Transactions on Systems, Man, and Cybernetics*, Part B (Cybernetics)(34.1):34–39, 2004.

[3] B. Kessler, G. Payne, W. Polyzou. Wavelet notes. The University of Iowa, February 2008.

[4] Venkata Ravikiran Chaganti. Edge detection of noisy images using 2-d discrete wavelet transform. Master's thesis, The Florida State University, 2005.

[5] S. Lahmiri, M. Boukadoum. Hybrid discrete wavelet transform and gabor filter banks processing for features extraction from biomedical images. *Journal of Medical Engineering*, 2013.

[6] James S. Walker. *A Primer on Wavelets and their Scientific Applications.* Taylor & Francis Group, second edition, 2008.