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:
The thesis treats statistical methods employing wavelets, focusing mainly on data mining and machine learning tasks. The special attention is paid to the edge detection problem, as an example of the pre-processing. The edge detection algorithm was developed and the thesis provides results for various images. There are also examples of the wavelet theory applications in medicine.

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

Data mining is a process of searching big datasets for transparent and useful regularities or relationships. Nowadays this field of study has become important because 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 tasks.

Wavelets, thanks to their properties, can be really helpful in this area. First of all, there exist efficient algorithms for wavelet transform. Low computational complexity is a big advantage in case of a big data analysis. Secondly, the vanishing moments' property, described in chapter 1, provides distinction in an importance of the wavelet coefficients, the bigger are more relevant. Thus, by removing those less significant we can obtain the trend of analysed data. Also, the wavelet coefficients are mainly decorrelated, hence a transformation data to wavelet domain facilitates conducting data mining algorithms.

Data mining is composed of a few problems: data management, pre-processing, a main mining process and post-processing. Wavelets are especially useful in two of them: pre-processing and core process. Before the main analysis, it is often required to clear the data from noisiness or simplify them. This stage is known as pre-processing. The most common tasks are:

- 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.

The core analysis is often carried out with machine learning algorithms. Here are the examples of processing types:

- 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. The idea is to determine the class for the analysed object. A wavelet-base algorithm in machine learning was presented by Castelli in 1996 [2]. This algorithm is notably faster than the classic one.
- Regression. Uses to predict future values based on historical data. Wavelet approach is the non-parametric regression. This case is similar to the dimensionality reduction.

Overview

The wavelet theory is introduced in chapter 1. The special attention is paid to the Daubechies family of wavelets, which are widely applied in data analysis. Also, the definitions of forward and backward Wavelet Transforms are characterized. There is also described, the most important in image analysis, 2-Dimensional Discrete Wavelet Transform.

Chapter 2 treats the main goal of the thesis, an edge detection problem, as an example of the dimensionality reduction in pre-processing. The wavelet-based algorithm is introduced and developed. Moreover, sections 2.2 and 2.3 are about selecting the parameters – threshold and wavelet type – for the various images.

Ultimately, chapter 3 presents the pre-processing application in the problem of biomedical images classification. The feature extraction is performed using 2-D Discrete Wavelet Transform and specific filters. There are examples of the biomedical images with the extracted features using the algorithm from the chapter 2.

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

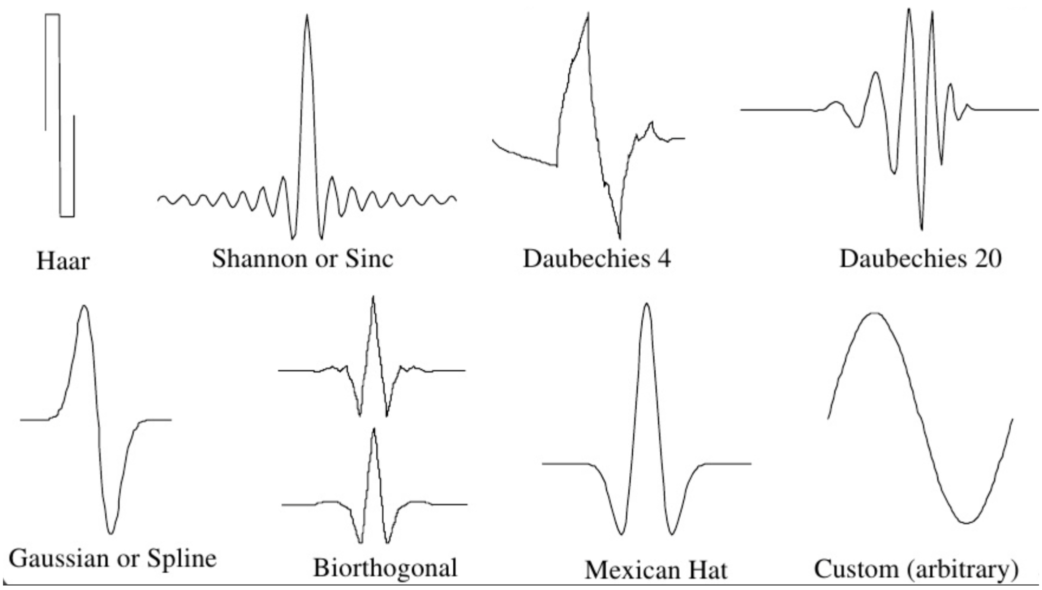


Figure 1.1: Different types of wavelets.
Source: https://aharchaou.com/from-fourier-to-curvelets/.

According to T. Li, S. Ma and M. Ogihara, wavelets should meet a set of important properties [7]. The mentioned properties are:

- Vanishing moments. Wavelet has n vanishing moments, when

$$\int_{\Omega} t^j \psi(t) dt = 0, \text{ for } j = 0, 1, \ldots, n-1. \tag{1.3}$$

  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 *Daub*4 and *Daub*20 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 by J. Walker [8], 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. Each of the Daubechies mother wavelets is denoted by *dbn* or *DaubN*, where $n \in \mathbb{N}$ 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 (*db*1), and also of all, wavelets. The mother function is illustrated on the graph 1.2 and has the following formula

$$\psi(t) = \begin{cases} 1, & \text{for } 0 \leq t < \frac{1}{2}, \\ -1, & \text{for } \frac{1}{2} \leq t < 1, \\ 0, & \text{otherwise.} \end{cases} \tag{1.4}$$

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.

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,

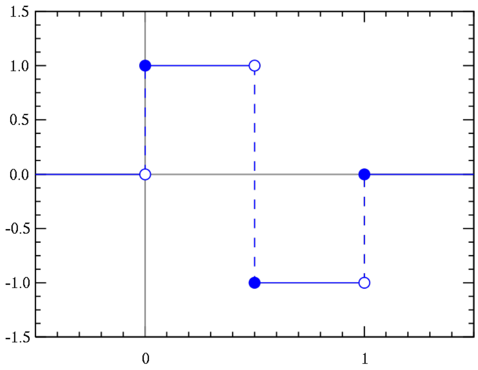


Figure 1.2: The Haar wavelet.
Source: https://en.wikipedia.org/wiki/Haar_wavelet.

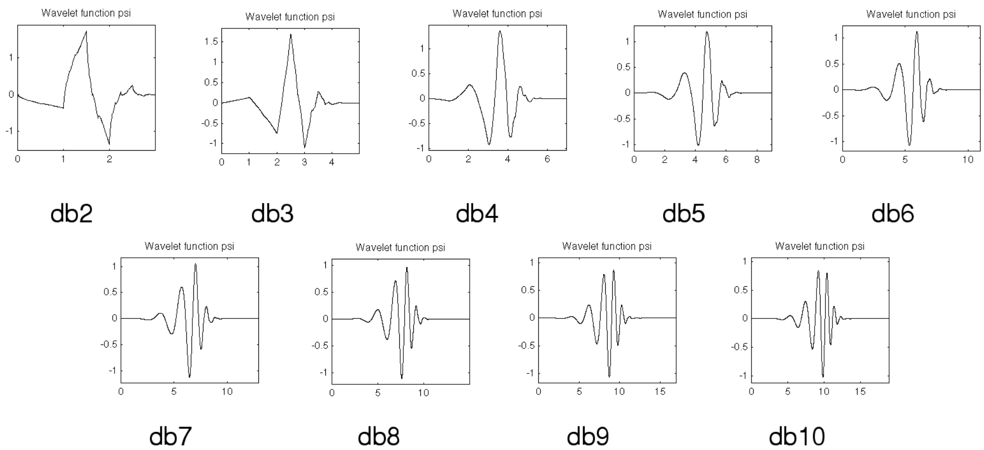


Figure 1.3: Daubechies wavelets.
Source: http://radio.feld.cvut.cz/matlab/toolbox/wavelet/ch01_31a.html.

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

$$W(a,b) = \int_{-\infty}^{\infty} y(t) a^{-\frac{1}{2}} \psi\left(\frac{t-b}{a}\right) dt, \tag{1.5}$$

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

Definition 1.5. Fourier transform is defined as

$$Y(f) = \int_{-\infty}^{\infty} y(t) e^{-i\omega t} dt, \tag{1.6}$$

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

The most important differences are presented in the table 1.1 on page 10.

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	Not suitable
Scaled and translated mother wavelets	Sine and cosine waves

Table 1.1: The comparison of Wavelet and Fourier Transform.

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 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.

Scale and translation parameters are defined as

$$a = 2^j \text{ and } b = 2^j m, \;\; j, m = 1, 2, \ldots . \tag{1.7}$$

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

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

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.

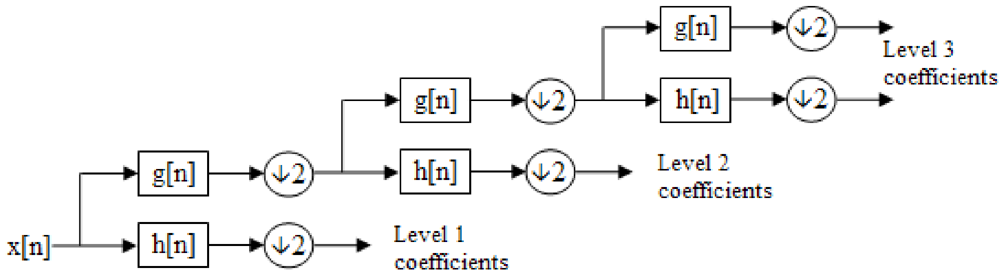


Figure 1.4: Discrete Wavelet transform on a signal $x(n)$.
Source: https://en.wikipedia.org/wiki/Discrete_wavelet_transform.

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.

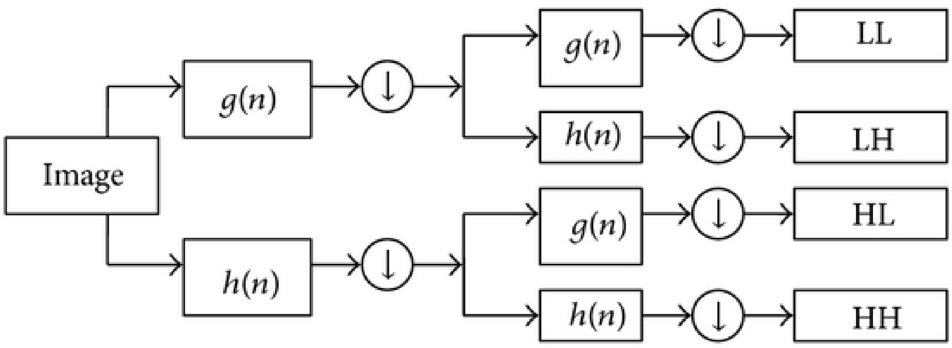


Figure 1.5: 2-D Discrete Wavelet transform on an image.
Source: https://www.hindawi.com/journals/jme/2013/104684/.

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

$$y(t) = \sum_{j=1}^{K} \sum_{m=0}^{2^{K-j}-1} d^{(j,m)} 2^{-\frac{j}{2}} \psi\left(2^{-j}t - m\right) + \psi_0, \tag{1.9}$$

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 [4]. 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 [9].

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.

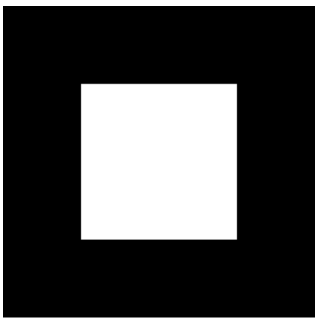


Figure 2.1: An initial image – a white square.

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.

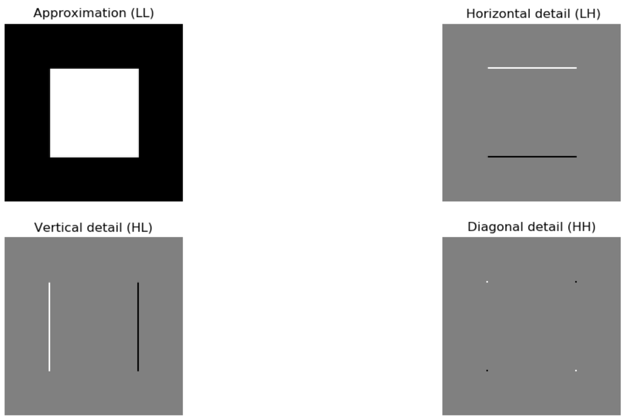


Figure 2.2: 2-D DWT coefficients.

Thus, we are interested only in components which gives information about edges, so we remove the approximation part – figure 2.3.

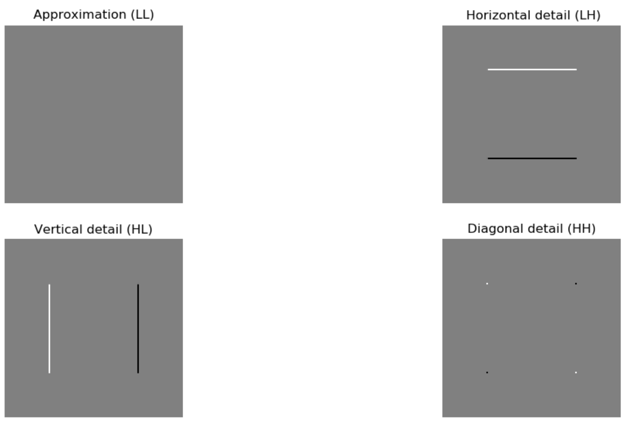


Figure 2.3: 2-D DWT coefficients with removed the LL part.

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.

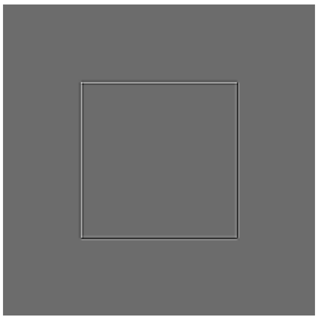


Figure 2.4: A reconstructed image showing the edges.

At the end, we can manipulate contrast to emphasize obtained lines. Currently, a background has grey colour and the edges are white or black. Values in the reconstructed image are between -127 and 128. The lowest are black, the highest white and grey background has 0 value. Therefore, using simple mathematical operation we can modify image to have black background and white edges. It is enough to get an absolute value [9]. Then, all black edges becoming white and the background is black, because of the lowest value – zero.

Finally, we obtain the edges of the square shown on the figure 2.5.

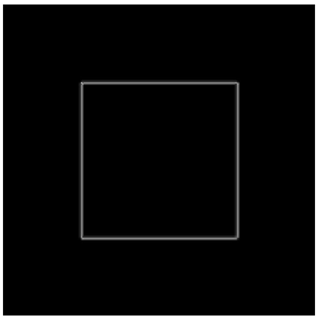


Figure 2.5: The edges of the square image after post-processing.

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

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

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.

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

2.2.1. Hard or soft thresholding

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

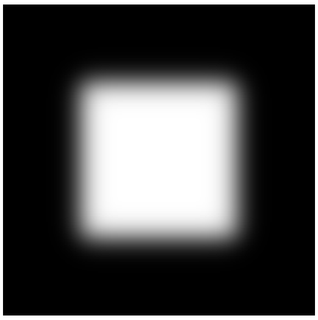


Figure 2.6: A noisy square image.

The coefficients of 2-D DWT are shown on the graph 2.7.

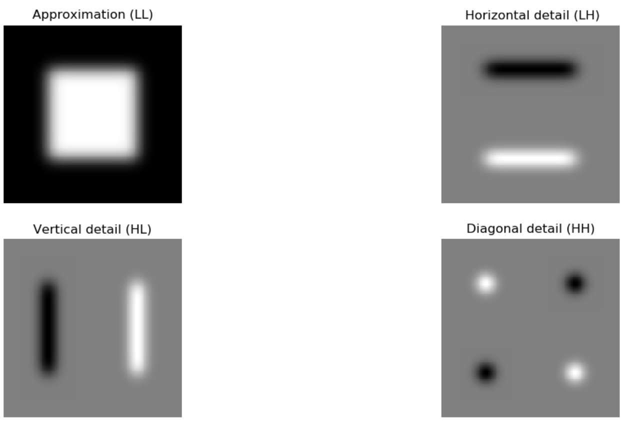


Figure 2.7: 2-D DWT coefficients.

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.

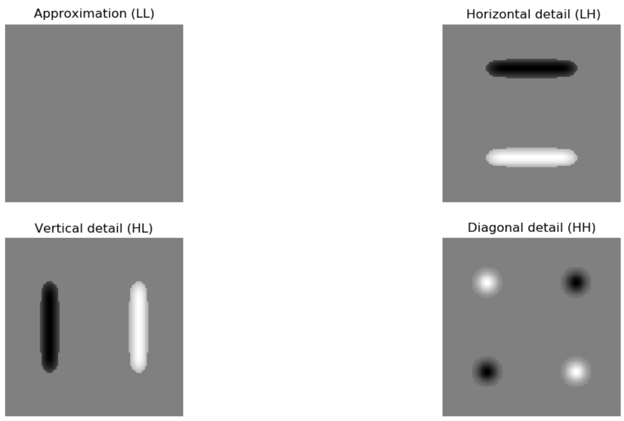


Figure 2.8: 2-D DWT coefficients after hard thresholding.

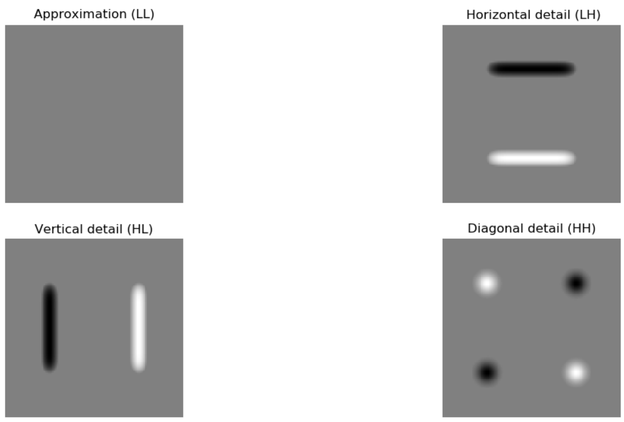


Figure 2.9: 2-D DWT coefficients after soft thresholding.

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

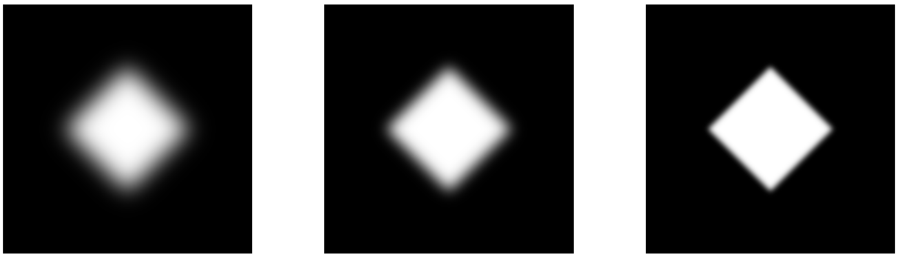
(a) hard (b) soft

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

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.

Lets consider three images of basic diamond shape with different noisiness σ, shown on the figure 2.11.



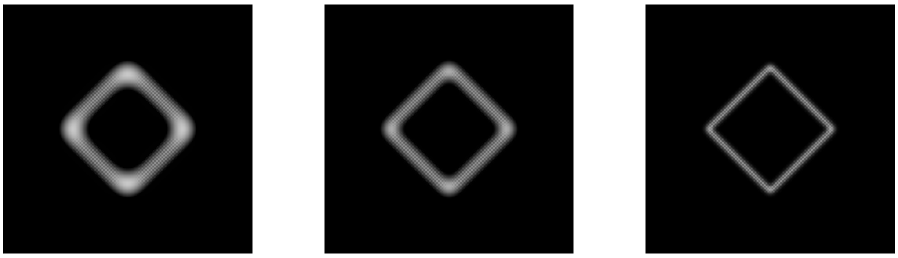
(a) $\sigma = 16$ (b) $\sigma = 10$ (c) $\sigma = 4$

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

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.

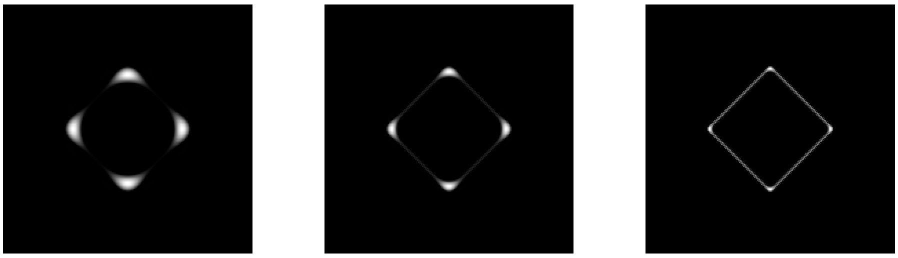
Hence, lets consider one more threshold on a 0.98 quantile level. The recognised edges for (a) and (b) images are presented on the figure 2.14. For this threshold edges are correctly detected.

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



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

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



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

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

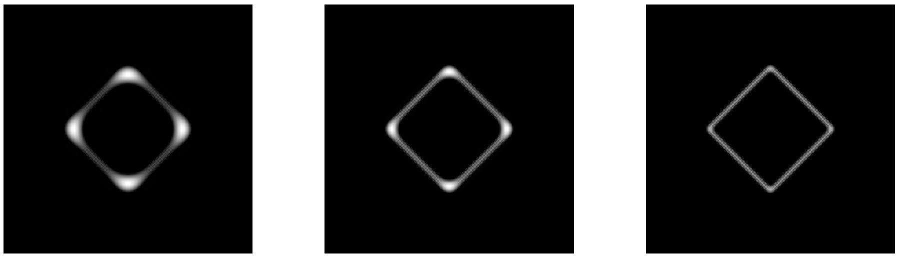
2.3. Selection of a wavelet type

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 (*db*1) 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 approximation 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



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

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

(a) original (b) $db1$

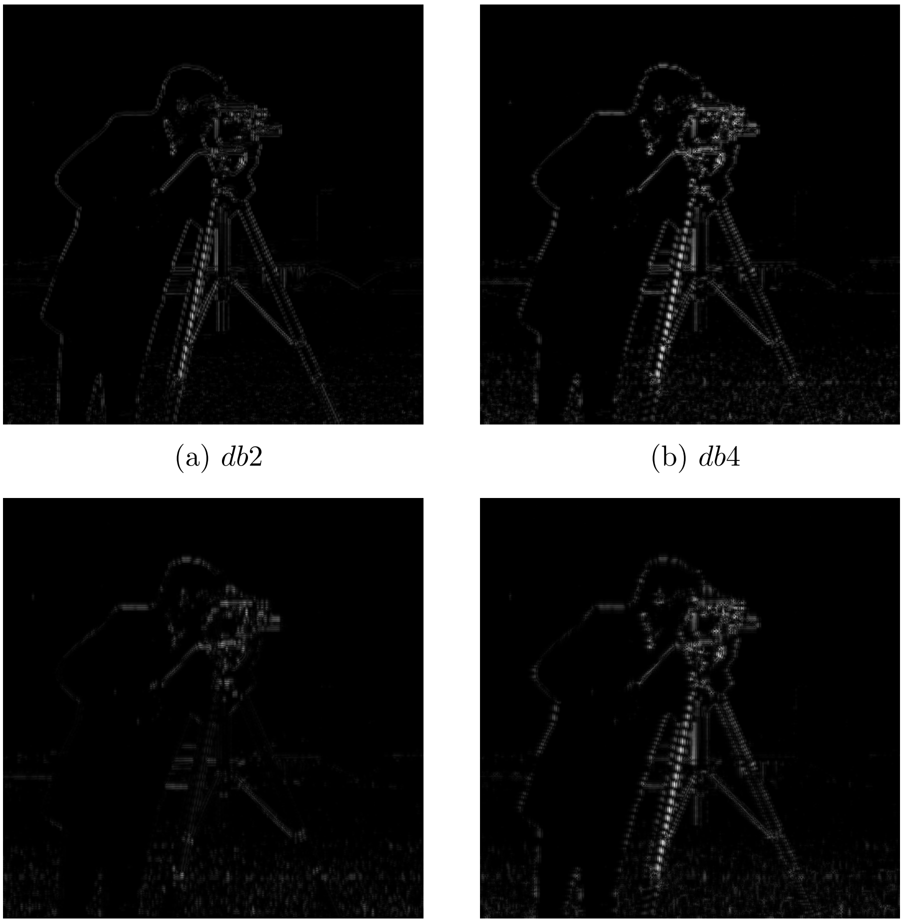
Figure 2.15: A man with a camera and detected edges using Haar wavelet.

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 $db2$ wavelets.

Lets consider the third example, a noisy square (fig. 2.18). In this case only the $db1$ (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.

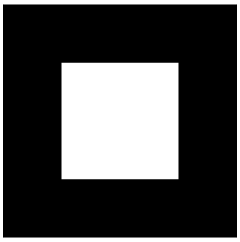
To sum up, for the majority of images the Haar wavelet provides the best results. However, there are cases, when the $db1$ wavelet fails. Then, it is required to use another of Daubechies wavelet, e.g. $db2$.



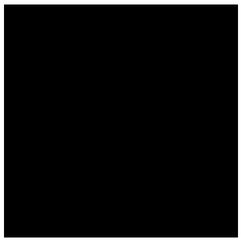
(a) *db*2 (b) *db*4

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

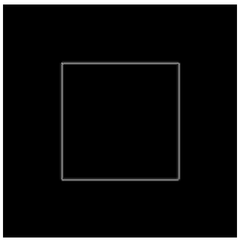
Figure 2.16: A man with a camera – detected edges using selected Daubechies wavelets.



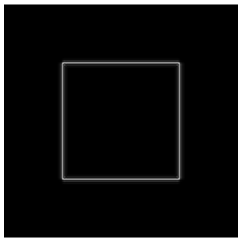
(a) original



(b) $db1$

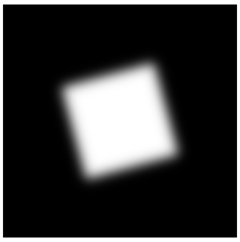


(c) $db2$

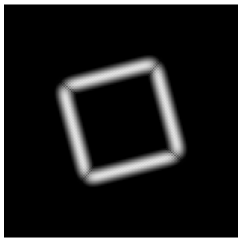


(d) $db4$

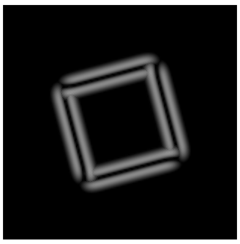
Figure 2.17: A simple square – detected edges using selected Daubechies wavelets.



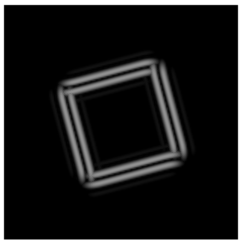
(a) original



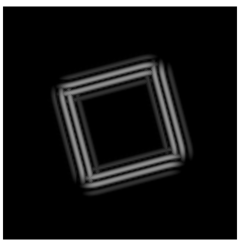
(b) $db1$



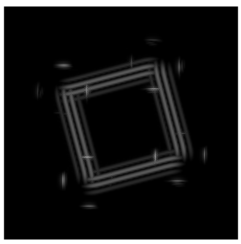
(c) $db2$



(d) $db4$



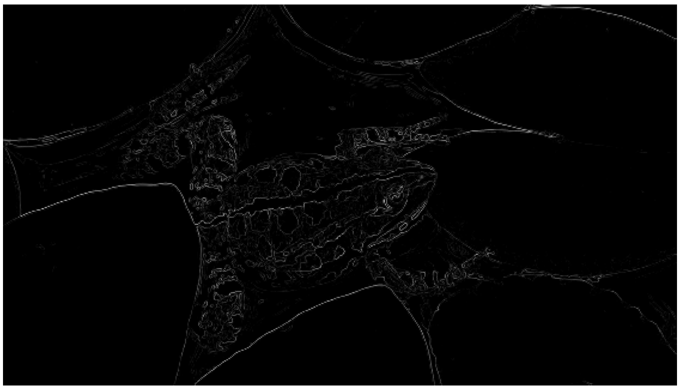
(e) $db6$



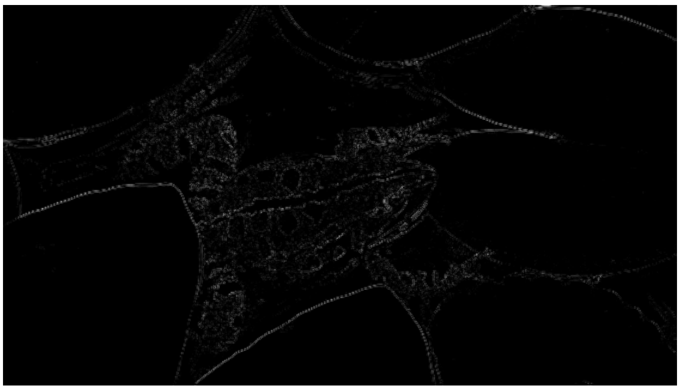
(f) $db8$

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

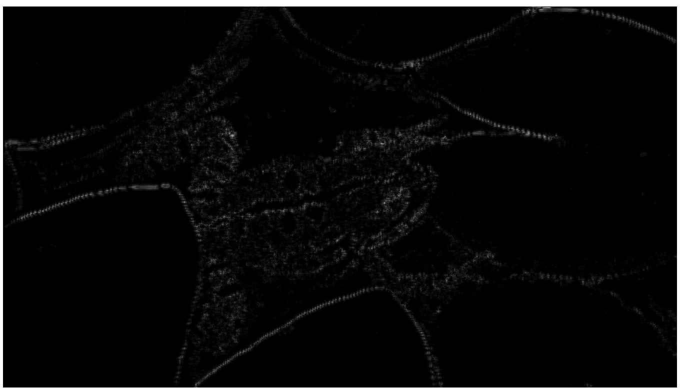
(a) original



(b) $db1$



(c) $db2$



(d) $db6$

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

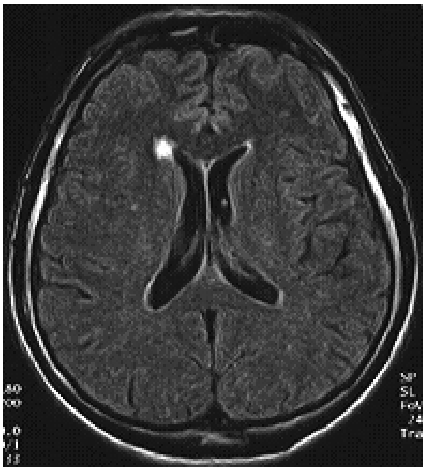
Chapter 3

Application in medicine

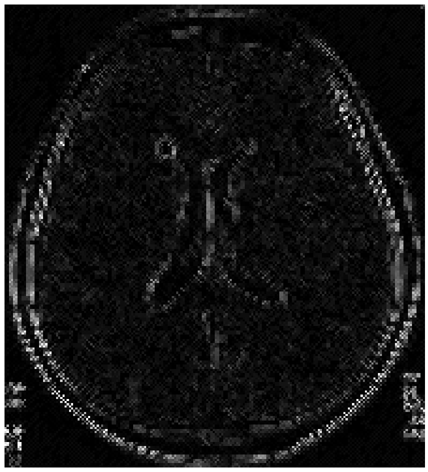
The present-day medicine uses computer algorithms to improve detecting, e.g. cancer, in the early stage. This kind of algorithms needs to extract features from the biomedical images and then, classify if there are any cancer changes. The 2-D Discrete Wavelet Transform provides information about brightness, shape and structure of images, because of the high time-frequency resolutions. The horizontal (LH) and vertical (HL) details of the wavelet decomposition can describe well the modifications in biological tissue [6].

Thus, in this application the DWT feature extraction is the pre-processing task for the machine learning classification. The Support Vector Machine binary classifier is applied to determine a disease on the analysed image. Algorithms with the wavelet-base pre-processing have widely application in detecting lots of diseases, e.g. the brain tumour, sclerosis, Alzheimer's, giloma, as well as the breast cancer, skin cancer or lung nodules [5].

The edge detection is one of the problems of features extraction. Lets look at the examples, how recognizing the edges can improve the clearness of the biomedical images. The figures 3.1-3.5, shows the detected edges of different medical images, i.e. brain MRI, x-ray of the limbs and lungs. The results were obtained slightly different then how it was described in the chapter 2. According to the S. Lahmiri and M. Boukadoum [6], only the LH and HL components leave, while the LL and HH are removed. Furthermore, there is no thresholding applied. As we can see, the main features of the specific images are visible after detection. This kind of dimensionality reduction can greatly improve the machine learning classification.

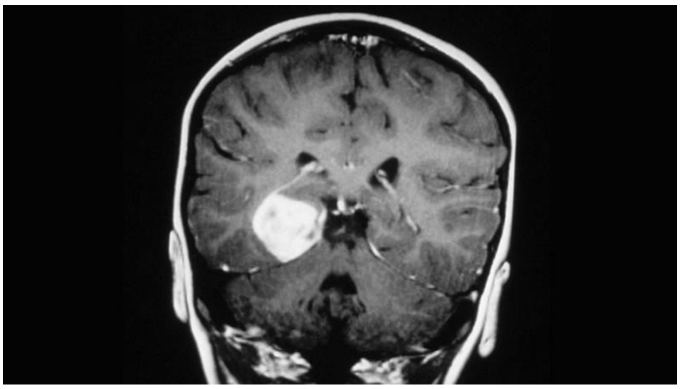


(a) original

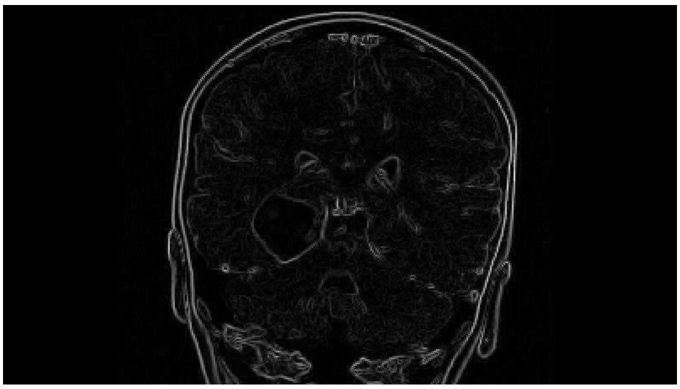


(b) detected edges

Figure 3.1: Brain MRI – a small tumour.

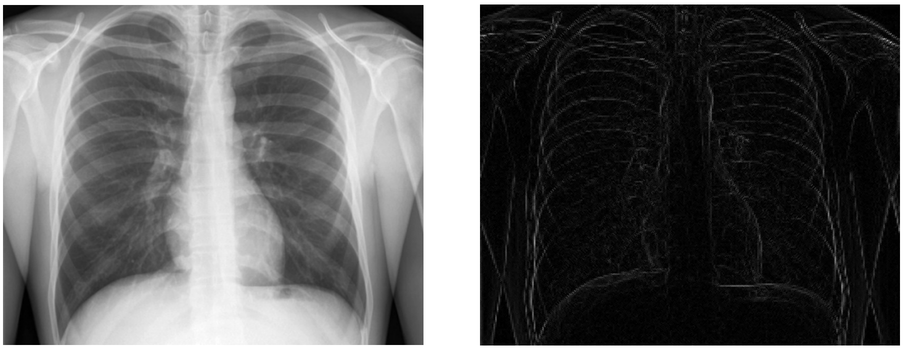


(a) original



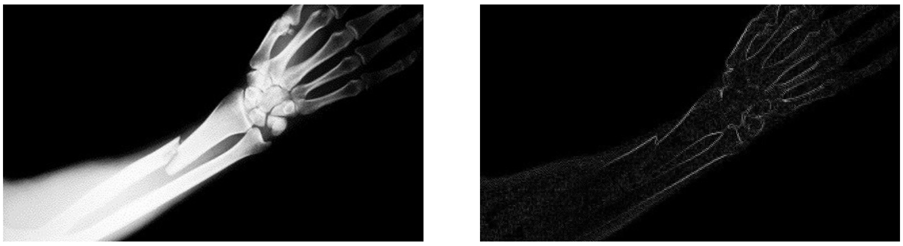
(b) detected edges

Figure 3.2: Brain MRI – a tumour.



(a) original (b) detected edges

Figure 3.3: Lungs.



(a) original (b) detected edges

Figure 3.4: A broken bone – arm.

(a) original (b) detected edges

Figure 3.5: A broken bone – leg.

Summary

In conclusion, the wavelet theory, described in the chapter 1, has wide applications in science and engineering problems, including data mining tasks. It is showed that particular wavelet properties facilitate and improve data analysis. Also the wavelet transform algorithms have low computational complexity, which is a huge advantage in case of big datasets operations.

The dimensionality reduction is one of the pre-processing tasks. As an example, the edge detection problem was considered. The chapter 2 contains the wavelet-based algorithm for edges recognition and the results for various type of images with different parameters. What characterize the Discrete Wavelet Transform, are the low and high frequency components. Thanks to those resolutions, we can filter only the desired informations. There are many different wavelets which can be used in wavelet transform, however Daubechies family is considered, because of their orthogonality and vanishing moments. It occurs that the simplest Haar wavelet provides the best results for the majority of images. Nevertheless, the other Daubechies wavelets are also useful, especially when the Haar fails. During the processing data, after transformation to the wavelet domain, we can also denoise coefficients using thresholding. It is showed that the soft thresholding leads to the smoother edges, hence it is more suitable for edge detection. We also considered different values of the threshold λ for various images. Noisier pictures, or those with different sharpness require increase the threshold to obtain more accurate results. Generally, changing the parameters, i.e. wavelet type and threshold λ, allows for the better fitting to the analysed image.

The chapter 3 presents an application of wavelet analysis in classification of the biomedical images. Wavelet transform helps with pre-processing like feature extraction. There are papers ([6], [5]), which show how wavelet-base algorithms can successfully recognize the disease, like cancer (brain, breast, lungs). We also introduced simple feature extraction, similar to the edge detection on the selected biomedical images.

Summarising, wavelets tend to be really great tool for data mining tasks, in pre--processing and also as an kernel in the machine learning algorithms. We shown an example of usage – wavelet-based edge detection, which provides good results and can be applied for various images thanks to the selection of parameters. The second presented example, the wavelets usage in analysis of the biomedical images, shows that wavelet theory is already applied in real life problems. Moreover, remark that the presented examples are just a few of many wavelets usages in data processing and currently the number of different papers with wavelet theory and applications is constantly growing.

Appendix

```
import numpy as np
import matplotlib.pyplot as plt

import pywt.data
from PIL import Image
from scipy import ndimage
import scipy.misc

wavelet_name = 'haar'
s = 8
image_name = 'camera'

# Load image
original = pywt.data.camera()

# im = np.zeros((256, 256)) # numpy square
# im[64:-64, 64:-64] = 1
# im = ndimage.rotate(im, 15, mode='constant') # diamond
# # im = ndimage.gaussian_filter(im, sigma=s)
# original = im

# original_image = Image.open(image_name + '.jpg').convert('L')
# # original_image = ndimage.gaussian_filter(original_image, sigma=s)
# original = np.asarray(original_image, dtype="int32")

# Wavelet transform of image, and plot approximation and details
titles = ['Approximation (LL)', ' Horizontal detail (LH)',
'Vertical detail (HL)', 'Diagonal detail (HH)']

coeffs2 = pywt.dwt2(original, wavelet_name)
LL, (LH, HL, HH) = coeffs2

t = [0, 0, 0, 0]
l = 95
fig = plt.figure()
for i, a in enumerate([LL, LH, HL, HH]):
   if i == 0:
      t[i] = np.percentile(a, 100) + 0.01
   elif i == 3:
      t[i] = np.percentile(a, l)
```

```
    else:
        t[i] = np.percentile(a, 1)
    a = np.flip(a, 0)
    ax = fig.add_subplot(2, 2, i + 1)
    ax.imshow(a, origin='image', interpolation="nearest",
     cmap=plt.cm.gray)
    ax.set_title(titles[i], fontsize=12)
    ax.set_axis_off()

coefs = [0, 0, 0, 0]
fig = plt.figure()
for i, a in enumerate([LL, LH, HL, HH]):
    th_mode = 'soft'
    da = pywt.threshold(a, t[i], mode=th_mode)
    coefs[i] = da
    da = np.flip(da, 0)
    ax = fig.add_subplot(2, 2, i + 1)
    if i == 0:
        ax.imshow(da, origin='image', interpolation="nearest",
         cmap=plt.cm.gray, vmin=-1, vmax=1)
    else:
        ax.imshow(da, origin='image', interpolation="nearest",
         cmap=plt.cm.gray)
    ax.set_title(titles[i], fontsize=12)
    ax.set_axis_off()

denoised_coeffs2 = coefs[0], (coefs[1], coefs[2], coefs[3])

# Now reconstruct and plot the original image
reconstructed = pywt.idwt2(denoised_coeffs2, wavelet_name)

r2 = np.abs(reconstructed)

# scipy.misc.toimage(original).save('medical/results/'
 + image_name + '.png')
# scipy.misc.toimage(reconstructed).save('graphs/' +
 image_name + '_' + wavelet_name + '_' + str(l) + '.png')
scipy.misc.toimage(r2).save('medical/results/' +
 image_name + '_' + wavelet_name + '_100_' + str(l) + '.png')

fig = plt.figure()
ax = fig.add_subplot(1, 2, 1)
ax.set_axis_off()
plt.imshow(original, interpolation="nearest", cmap=plt.cm.gray)
ax = fig.add_subplot(1, 2, 2)
ax.set_axis_off()
plt.imshow(r2, interpolation="nearest", cmap=plt.cm.gray)

plt.show()
```

List of Figures

Bibliography

[1] *Data Mining and Knowledge Discovery Handbook*. Springer, 2010.

[2] V. Castelli, C. Li, J. Turek, I. Kontoyiannis. *Progressive classification in the compressed domain for large EOS satellite databases*, Apr. 1996.

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

[4] Dharampal, V. Mutneja. Methods of image edge detection: A review. *Journal of Electrical & Electronic Systems*, 2015.

[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] S. Lahmiri, M. Boukadoum. Classification of brain mri using wavelet decomposition and svm. *International Journal of Computer Applications*, 154(5), Nov. 2016.

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

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

[9] C. Yuriy. Edge detection in images with wavelet transform, Nov. 2007.

[10] 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.