VILNIUS GEDIMINAS TECHNICAL UNIVERSITY

Liudas STAŠIONIS

DEVELOPMENT OF SELF-ORGANIZING METHODS FOR RADIO SPECTRUM SENSING

DOCTORAL DISSERTATION

TECHNOLOGICAL SCIENCES,
ELECTRICAL AND ELECTRONIC ENGINEERING (01T)

VGTU
LEIDYKLA TECHNIKA
Vilnius 2016

Doctoral dissertation was prepared at Vilnius Gediminas Technical University in 2011–2016.

Scientific supervisor

Assoc Prof Dr Artūras SERACKIS (Vilnius Gediminas Technical University, Electrical and Electronic Engineering – 01T).

The Dissertation Defense Council of Scientific Field of Electrical and Electronic Engineering of Vilnius Gediminas Technical University:

Chairman

Prof Dr. Šarūnas PAULIKAS (Vilnius Gediminas Technical University, Electrical and Electronic Engineering – 01T).

Members:

Prof Dr Habil Gintautas DZEMYDA (Vilnius University, Informatics Engineering – 07T),

Assoc Prof Dr Rimantas PUPEIKIS (Vilnius Gediminas Technical University, Informatics – 09P),

Dr Habil Arūnas Vytautas TAMAŠEVIČIUS (State research institute Center for Physical Sciences and Technology, Electrical and Electronic Engineering – 01T),

Dr Habil Yevhen YASHCHYSHYN (Warsaw University of Technology, Electrical and Electronic Engineering – 01T).

The dissertation will be defended at the public meeting of the Dissertation Defense Council of Electrical and Electronics Engineering in the Senate Hall of Vilnius Gediminas Technical University at **9 a. m. on 29 April 2016**.

Address: Saulėtekio al. 11, LT-10223 Vilnius, Lithuania.

Tel.: +370 5 274 4956; fax +370 5 270 0112; e-mail: doktor@vgtu.lt

A notification on the intend defending of the dissertation was send on 25 March 2016.

A copy of the doctoral dissertation is available for review at the VGTU repository: http://dspace.vgtu.lt/ and at the Library of Vilnius Gediminas Technical University (Saulėtekio al. 14, LT-10223 Vilnius, Lithuania).
VGTU leidyklos TECHNIKA 2363-M mokslo literatūros knyga .

ISBN 978-609-457-912-7

© VGTU leidykla TECHNIKA, 2016

© Liudas Stašionis, 2016

liudas.stasionis@gmail.com

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

Liudas STAŠIONIS

SAVIORGANIZUOJANČIŲ METODŲ RADIJO DAŽNIŲ JUOSTOS UŽIMTUMUI STEBĖTI KŪRIMAS

DAKTARO DISERTACIJA

TECHNOLOGIJOS MOKSLAI,
ELEKTROS IR ELEKTRONIKOS INŽINERIJA (01T)

VGTU
LEIDYKLA TECHNIKA
Vilnius 2016

Disertacija rengta 2011–2016 metais Vilniaus Gedimino technikos universitete.

Vadovas

doc. dr. Artūras SERACKIS (Vilniaus Gedimino technikos universitetas, elektros ir elektronikos inžinerija – 01T).

Vilniaus Gedimino technikos universiteto Elektros ir elektronikos inžinerijos mokslo krypties disertacijos ginimo taryba:

Pirmininkas

prof. dr. Šarūnas PAULIKAS (Vilniaus Gedimino technikos universitetas, elektros ir elektronikos inžinerija – 01T).

Nariai:

prof. habil. dr. Gintautas DZEMYDA (Vilniaus universitetas, informatikos inžinerija – 07T),

habil. dr. Yevhen YASHCHYSHYN (Varšuvos technologijos universitetas, elektros ir elektronikos inžinerija – 01T),

doc. dr. Rimantas PUPEIKIS (Vilniaus Gedimino technikos universitetas, informatika – 09P),

habil. dr. Arūnas Vytautas TAMAŠEVIČIUS (Valstybinis mokslinių tyrimų institutas Fizinių ir technologijos mokslų centras, elektros ir elektronikos inžinerija – 01T).

Disertacija bus ginama Elektros ir elektronikos inžinerijos mokslo krypties disertacijos gynimo tarybos posėdyje **2016 m. balandžio 29 d. 9 val.** Vilniaus Gedimino technikos universiteto senato posėdžių salėje.

Adresas: Saulėtekio al. 11, LT-10223 Vilnius, Lietuva.
Tel.: (8 5) 274 4956; faksas (8 5) 270 0112; el. paštas doktor@vgtu.lt

Pranešimai apie numatomą ginti disertaciją išsiųsti 2016 m. kovo 25 d.

Disertaciją galima peržiūrėti VGTU talpykloje http://dspace.vgtu.lt ir Vilniaus Gedimino technikos universiteto bibliotekoje (Saulėtekio al. 14, LT-10223 Vilnius, Lietuva).

Abstract

A problem of wide-band radio spectrum analysis in real time was solved and presented in the dissertation. The goal of the work was to develop a spectrum sensing method for primary user emission detection in radio spectrum by investigating new signal feature extraction and intelligent decision making techniques. A solution of this problem is important for application in cognitive radio systems, where radio spectrum is analyzed in real time.

In thesis there are reviewed currently suggested spectrum analysis methods, which are used for cognitive radio. The main purpose of these methods is to optimize spectrum description feature estimation in real-time systems and to select suitable classification threshold. For signal spectrum description analyzed methods used signal energy estimation, analyzed energy statistical difference in time and frequency. In addition, the review has shown that the wavelet transform can be used for signal pre-processing in spectrum sensors. For classification threshold selection in literature most common methods are based on statistical noise estimate and energy statistical change analysis. However, there are no suggested efficient methods, which let classification threshold to change adaptively, when RF environment changes.

It were suggested signal features estimation modifications, which let to increase the efficiency of algorithm implementation in embedded system, by decreasing the amount of required calculations and preserving the accuracy of spectrum analysis algorithms.

For primary signal processing it is suggested to use wavelet transform based features extraction, which are used for spectrum sensors and lets to increase accuracy of noisy signal detection. All primary user signal emissions were detected with lower than 1% false alarm ratio. In dissertation, there are suggested artificial neural network based methods, which let adaptively select classification threshold for the spectrum sensors. During experimental tests, there was achieved full signals emissions detection with false alarm ratio lower than 1%.

It was suggested self organizing map structure modification, which increases network self-training speed up to 32 times. This self-training speed is achieved due to additional inner weights, which are added in to self organizing map structure. In self-training stage network structure changes especially fast and when topology, which is suited for given task, is reached, in further self-training iterations it can be disordered. In order to avoid this over-training, self-training process monitoring algorithms must be used. There were suggested original methods for self-training process control, which let to avoid network over-training and decrease self-training iteration quantity.

Reziumė

Disertacijoje sprendžiama plačios juostos radijo spektro analizės realiuoju laiku problema. Analizės metu siekiama spektre nustatyti sritis (juostas, kanalus) kuriuose nėra jokio pirminio vartotojo (angl. *Primary User*) signalo ir nuolat stebėti, ar analizuojamoje srityje neatsiranda nežinomas pirminio vartotojo signalas. Tokio pobūdžio problemos sprendžiamos kognityvinio radijo uždaviniuose, kai radijo dažnių ruožas analizuojamas realiuoju laiku.

Disertacijoje apžvelgti šiuo metu siūlomi signalo spektro analizės metodai, skirti kognityviniam radijui. Šių metodų pagrindiniai iššūkiai – optimizuoti signalo spektro apibūdinimui naudojamų požymių skaičiavimą realiojo laiko sistemose ir parinkti tinkamą signalų klasifikatorius slenkstį. Signalų spektrui apibūdinti naudojamas signalo energijos skaičiavimas, analizuojamos statistinės energijos pokyčio laike savybės. Taip pat gali būti taikomas ir vilnelių transformacija grįstas signalų pirminis apdorojimas. Klasifikatoriaus slenksčiui parinkti literatūroje sutinkami metodai, grįsti statistiniais triukšmo įverčiais, energijos pokyčio analize, tačiau nėra pasiūlytų efektyvių metodų, leidžiančių klasifikatoriaus slenksčio reikšmes keisti adaptyviai, pakitus triukšmo lygiui analizuojamoje aplinkoje.

Buvo pasiūlytos signalų požymių skaičiavimo patobulinimai, leidžiančios padidinti įgyvendinimo įterptinėje sistemoje efektyvumą, sumažinant reikiamų atlikti skaičiavimų apimtis, ir išlaikyti šiuos požymius naudojančio spektro analizės algoritmo efektyvumą.

Pirminiam signalų apdorojimui pasiūlyti nauji vilnelių transformacija grįsti požymiai, skirti taikyti spektro jutikliuose ir leidžiantys padidinti signalų atpažinimo triukšme tikslumą. Naudojantis šiais požymiais pirminio vartotojo signalo visos emisijos aptiktos su mažesniu nei 1% klaidos santykiu. Pasiūlyti saviorganizuojančių neuronų (Kohonen) tinklų mokymusi grįsti metodai, leidžiantys spektro jutiklyje adaptyviai parinkti klasifikatoriaus slenkstį. Tinklų eksperimentuose pirminio vartotojo visos emisijos aptiktos su mažesniu nei 1% klaidos santykiu.

Pasiūlyta saviorganizuojančio neuronų tinklo struktūros patobulinimai, leidžianti paspartinti tinklo mokymąsi iki 32 kartų. Tokia tinklo mokymosi sparta pasiekta dėl papildomų vidinių ryšių, kuriais buvo papildyta Kohonen tinklo struktūra. Mokymosi metu tinklo struktūra kinta ypač sparčiai ir pasiekusi topologiją, tinkamą iškeltam uždaviniui spręsti, tolesnio mokymosi metu šią topologiją gali išardyti. Siekiant išvengti tokio Kohonen tinklo persimokymo, reikia taikyti mokymo eigos stebėjimo algoritmus. Pasiūlyti originalūs metodai saviorganizuojančio neuronų tinklo mokymui sustabdyti, leidžiantys išvengti tinklo persimokymo ir sumažinti mokymui skirtų iteracijų skaičių.

Notations

Symbols

α	log-sigmoid activation function slope coefficient;
b_{ij}	ij neuron bias;
C_f^W	spectral coherence coefficient estimated in W frequency window;
D	variance;
D_{k}	modified variance, when k: N – variance without FIFO, M – variance without square operation;
$\mathcal{D}$	desired network output;
b_{ij}	ij neuron bias;
d_{ij}	distance between ij neuron weights and and input;
$d^{\perp}$	distance between winner neuron weights and input;
Δ_{init}	initial SOM self-training change;
E	mean square error;
$e(n)$	network instant output error;
$\eta(n)$	self-training ration in n iteration, or neuron weight adjustment factor;
η_0	initial self-training ratio;
η_{nb}	neighboring neuron self-training ratio;
f	radio spectrum component frequency;

F_{C}	radio spectrum center frequency;
G_n	radio noise component;
$\gamma(n)$	neighboring Gaussian function slope parameter;
H_{k}	wavelet transform high-pass filter coefficients, when k: Haar, Deb – Daubechies and Sym – Symlet transforms;
$h_{\mathrm{H}}^{\mathrm{k}}$	wavelet transform high-pass filter output, when k: Haar, Deb – Daubechies and Sym – Symlet transforms;
$h_{\mathrm{L}}^{\mathrm{k}}$	wavelet transform low-pass filter output, when k: Haar, Deb – Daubechies and Sym – Symlet transforms;
κ	self-training depth;
L	quantity of network layers;
L_{k}	wavelet transform Low-pass filter coefficients, when k: Haar, Deb – Daubechies and Sym – Symlet transforms;
N	quantity of samples;
N_{wind}	endpoint estimation window;
$\mathcal{N}_{ij}$	ij neuron;
$\mathcal{N}^{\perp}$	winner neuron identity;
n	sample (reference) index;
nb	neighborhood function;
$\mathcal{Y}$	neuron network output;
i, j, h, l	input, neuron weight, network layer and output index;
ω_{ij}	ij neuron weight;
$s_{ij}^{(l)}$	ij neuron output before activation function;
$\tilde{s}_{ij}^{(l)}$	ij neuron in l layer output after activation function;
$\Phi_{\mathrm{k}}^{(l)}(\centerdot)$	neuron activation function in l layer, when k: S – Heaviside (threshold) or LS – Log-sigmoid;
$\mathcal{N}^{\perp}$	winner neuron identity;
ϕ_{k}	spectrum features extraction, when k: A – average, D – variance, $\mathrm{D_N}$ – variance without FIFO, D_{M} – variance without square, σ – std. deviation, σ_{N} – std. deviation without FIFO, H – Haar, Deb – Daubechies, Deb3 – Daubechies without LL branch, S – Symlet, AD – average and variance, $\mathrm{AD_N}$ average and variance without FIFO extractors;

S_f^W	discrete spectral correlation coefficient estimated in W frequency window;
$s_{ij}^{(l)}$	ij neuron output before activation function;
$\tilde{s}_{ij}^{(l)}$	ij neuron in l layer output after activation function;
σ	standard deviation;
σ_{N}	standard deviation without FIFO buffer;
T	time period;
t	sample (reference) index in time;
τ	self-training time constant;
θ_{end}	SOM self-training suspension parameter;
u_n	primary user signal component;
W	radio spectrum frequency window, or cycle frequency;
x_n	radio signal component.

Operators and Functions

$\overline{\cdot}$	average;
$\mathcal{F}(\cdot)$	discrete Fourier transform;
$\partial(\cdot)$	partial derivative;
$\|\| \cdot \|\|$	Euclidean distance;
$\min(\cdot)$	minimum;
$\cdot^{\top}, \cdot^{\perp}$	maximum and minimum.

Abbreviations

ADC	Analog Digital Converter;
ANN	Artificial Neural Network;
AWGN	Additive White Gaussian Noise;
CFE	Cyclostationary Features Extraction;
DSCF	Discrete Spectral Correlation Function;
DSP	Digital Signal Processing;
FFT	Fast Fourier Transform;
FIFO	First in First out;
FPGA	Field-Programmable Gate Array;
FSM	Finite-Stat Machine;
RAM	Random Access Memories;
RF	Radio Frequency;
ROC	Receiver operating characteristic;

SCC	Spectral Coherence Coefficient;
SDR	Software Defined Radio;
SNR	Signal to Noise Ratio;
SOM	Self Organizing Maps;
THD	Decision threshold.

Contents

Introduction

The Investigated Problem

The problem of low SNR$\sim$ 0 dB signal detection in the wide frequency band, when just few samples for features estimation are used, is addressed in this dissertation (Penna *et al.* 2009; Rasheed *et al.* 2010; Srinu *et al.* 2011; Zhuan *et al.* 2008). The exclusive attention is gained on the automatic detection of the signal without any prior knowledge about the content of the signal, neither the type of signal modulation nor the carrier frequency of the signal (Joshi *et al.* 2011a; Nair *et al.* 2010). This kind of problems usually appear in cognitive radio applications. In order to add a secondary user transmission to the radio channel already occupied by another user, the spectrum sensor should be able to detect primary user signal in order to avoid the collision (Stankevičius *et al.* 2015). Therefore spectrum sensing systems must ensure full or near to full primary user detection (Nastase *et al.* 2014).

Currently different spectrum analysis methods are investigated in order to increase the efficiency of the primary user signal (emission) detection in real-time (Chantaraskul, Moessner 2010; Kyungtae *et al.* 2010; Young, Bostian 2013). Especially problematic is the detection of the primary user signals with low energy level, comparing to channel noise level, when noise distribution is not entirely white Gaussian (Bagchi 2014; Wang, Salous 2011). Therefore sensor must continuously or periodically adjust to changing RF environment.

Importance of the Thesis

Currently there is very little RF spectrum space (up to 3 GHz), which is not appointed for service providers (Communications Regulatory Authority of the Republic of Lithuania 2015, Electronic Communications Committee within the European Conference of Postal and Telecommunications Administrations 2015). Occupancy in the very-high and ultra-high frequency bands is 10–15% even in densely populated territories. It means that more than 85% of RF spectrum is not used (Taher *et al.* 2011).However occupancy in some unlicensed radio spectrum bands like 2.4 GHz are high (Statkus, Paulikas 2012).

A number of papers in recent years focus on the effective utilization of the unused radio spectrum even if it is assigned for primary user (De Vito 2012; Jayavalan *et al.* 2014; Mehdawi *et al.* 2013; Seshukumar *et al.* 2013; Sun *et al.* 2013; Sunday *et al.* 2015; Yucek, Arslan 2009). A spectrum sensor added to a physical layer of communication systems in cognitive radio solution is responsible on the detection of primary user in noisy environment. Spectrum access should be provided for secondary users only if primary user is absent.

The sensitivity of the sensor, which detects the gap in the spectrum, directly depends on the algorithm used in cognitive radio solution. In addition, the influence of continuously changing environment should be taken into account by the sensor. Therefore, the spectrum sensor should have a possibility to adapt to current environment state.

The Object of Research

The object investigated in thesis is a spectrum sensor, able to detect all primary users signals emissions in 25 MHz spectrum band-with, minimizing the false alarm rate. The primary user emission can be populated anywhere up to 3 GHz in RF spectrum:

- in analyzed signal spectrum there may be noise component with a primary user signal or noise component only;
- there is no prior knowledge about what kind of primary user signal may be present in the analyzed spectrum band together with noise.

The Goal of the Thesis

The goal of the thesis is to develop a spectrum sensing method for primary user emission detection in radio spectrum by investigating new signal feature extraction and intelligent decision making techniques.

The Tasks of the Thesis

After the review of the current researches results, two hypothesis were formulated:

1. Wavelet transform based signal spectrogram feature extraction can increase the primary user emission detection rate in comparing with signal energy based features extractors.
2. Self-adaptation of the spectrum sensor to continuously changing radio environment can be achieved by the application of intelligent methods, modifying the self-training algorithms for implementation in embedded system.

In order to confirm both hypothesis, the following tasks were formulated in the dissertation:

1. Investigate the application of wavelet transform to increase the performance of spectrum sensor.
2. Investigation of the neural network with binary activation functions suitability for spectrum sensing applications.
3. Development of the spectrum sensing method based on a self-organizing map and investigate its modifications to decrease the duration of self-training preserving the primary user emission detection performance.

Research Methodology

There are two signal processing stages investigated in dissertation: signal feature extraction and signal detection by classification. The extraction of the signal features was performed by the use of methods for digital signal processing in time domain and methods for digital signal processing in frequency domain.

The methods for signal detection by classification were investigated by the use of artificial neural networks with supervised training algorithms and a self-organizing map with un-supervised self-training algorithm.

The offline experimental research were performed using MATLAB™, Python-based software tools. The FPGA based implementations using VHDL were performed for experimental investigation in real RF environment.

Scientific Novelty

The scientific novelty of this dissertation is following:

1. The original modifications of the signal feature extraction algorithms for efficient implementation with low latency in FPGA based systems,

preserving the primary user signal detection rate of the radio spectrum sensor.

2. Proposed spectrum features extraction technique based on wavelet transform decreases the false alarm ratio 2–9 times comparing to energy average, standard deviation or variance based spectrum sensors.
3. Proposed decision making technique, based on a self-organizing map is able adapt to radio spectrum with signals having different spectral features and preserve the false alarm ratio of 1%, estimated by receiver operating characteristics.
4. An original modification of the self-organizing map requires 32 times less self-training interations comparing to number of iterations recommended in literature according to lattice size.
5. Proposed original self-organizing map self-training endpoint selection technique reduces the numbet of iterations by 2.6–44.6% comparing to cluster quality estimation based technique.

Practical Significance of Achieved Results

In thesis are suggested new methods, which are adjust for implementation in embedded systems. These methods are implemented in MATLAB™ environment, and them are implemented in FPGA. Real time experimental researches helped to determinate and compare RF spectrum occupancy in (two different RF bands: licensed unlicensed) various areas: city, countryside and village. From experimental results was determined, that 954 MHz RF band is 38–74% occupied by primary user signals, although 433 MHz band channel capacity is used just by $8.6 \cdot 10^{-5}$–$6.275 \cdot 10^{-3}$%.

The Defended Statements

1. The application of the Daubechies wavelet transform in blind spectrum sensing application may decrease the false alarm rate below 1%, preserving full detection of primary user signals with SNR above 0.8 dB.
2. Self-adaptation of the spectrum sensor to continuously changing radio environment can be achieved by the application of self-organizing map, ensuring the false alarm rate below 1% and preserving full detection of primary user signals with SNR above 0.8 dB.
3. The modification of the self-organizing map topology during self-training phase increases the efficiency of the sensor implementation in

FPGA based embedded systems and decreases the number of self-training iterations up to 32 times comparing to number of iterations recommended in literature according to lattice size.

Approval of the Results

The main results of the doctoral dissertation were published in 10 scientific papers: 1 paper in foreign journal indexed in Thomson ISI Web of Science (Stasionis, Serackis 2015a); 3 papers in local journal indexed in Thomson ISI Web of Science (Serackis et al. 2014; Stasionis, Serackis 2011, 2014); 2 papers in journal indexed in other databases, including ICONDA and IndexCopernicus (Sledevic, Stasionis 2013; Stasionis, Sledevic 2013); 4 papers in conference proceedings (Serackis, Stasionis 2014; Stasionis, Serackis 2012, 2013, 2015b). The main results of the thesis were reported at the following scientific conferences:

- International Conference *Electronics*, 2011, 2013, 2014, Palanga, Lithuania;
- Young Scientist Conference *Science – Future of Lithuania*, 2012–2015, Vilnius, Lithuania;
- 22^{nd} International Conference *Electromagnetic Disturbances*, 2012, Vilnius, Lithuania;
- 15^{th} International Conference *EUROCON*, 2013, Zagreb, Croatia;
- 22^{nd} International Conference *Nonlinear Dynamics of Electronic Systems* (NDES), 2014, Albena, Bulgaria;
- 16^{th} International Conference *EUROCON*, 2015, Salamanca, Spain.

Structure of the Dissertation

Thesis consist of: introduction, three chapters and general conclusions. In addition in thesis are: used notations and abbreviations lists also material index. Volume of thesis is 165 pages, in which are: 45 formulas, 123 figures and 30 tables, also in thesis 157 references are used.

Acknowledgements

First and foremost I wish to thank my advisor dr. Artūras Serackis for his trust and support. I would like to thank him for giving me the freedom to find my own way in a research.

The environment Artūras Serackis and Dalius Navakauskas created at Electronic Intelligent System (EIS) group is probably the best a PhD student could ever hope for. I'm thankful to other EIS group members: Dovilė Kurpytė, Darius Plonis, Dalius Matuzevičius, Raimond Laptik, Gintautas Tamulevičius and Andrius Katkevičius for your support. I'm specially thankful to Tomyslav Sledevič, it was great pleasure to work with you.

I wish to thank to my coworkers and my superiors for giving me an opportunity to work and study my PhD.

I dedicate this thesis to my wife Giedrė Stašionė for hers constant support and unconditional love. I love you dearly.

1

Literature Survey on Spectrum Sensing Methods

In this section the overview of main current research directions in dissertation theme are presented. The purpose of spectrum sensor, analyzed in this dissertation, is to find free windows in radio environment (in time and frequency), which are not used by primary users.

1.1. Classification of Primary User Signal Detectors

In literature, the spectrum sensing detectors usually are divided in to two groups (Perera, Herath 2011; Seshukumar *et al.* 2013; Yucek, Arslan 2009):

- Spectrum sensing without prior knowledge. This type of detectors does not require prior information about primary user signal. Detectors are adapted to RF channel.
- Spectrum sensing with prior knowledge. For this type of detectors an additional information about primary user signals (e.g., modulation type, signal shape, transceiver characteristics and etc.) is needed.

To detectors without prior knowledge are assigned: energy detectors and detectors based on Wavelet transform (Fig. 1.1). Energy detectors are least complex spectrum sensing algorithms (Axell *et al.* 2012; Sun *et al.* 2013). Energy calculation can be based on estimation of spectrum average or deviation/variance. Higher processing complexity, comparing to energy estimation,

has detectors based on Wavelet transform (De Vito 2012). However, using wavelet transform more signal features can be extracted from RF spectrum.

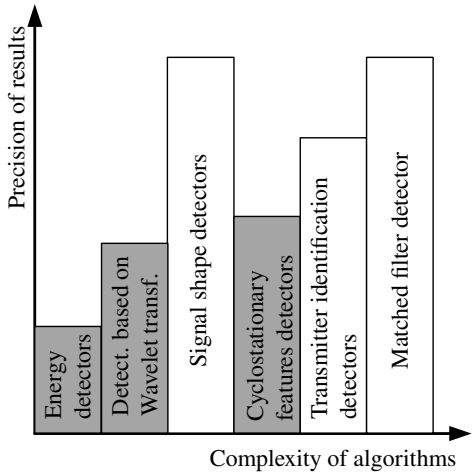


Fig. 1.1. Primary user signal detectors classification by complexity and accuracy

Other detector's group are the detectors with prior knowledge about primary user signal. To this group there are assigned: signal shape recognition based detectors, cyclostationary feature extraction based detectors, transmitter identification based detectors and detectors based on application of matched filters (Yucek, Arslan 2009). In most cases, the detector's with prior knowledge algorithms has higher complexity than those, used in detectors without prior knowledge (Fig. 1.1). However, these detectors has greater accuracy potential (Moosavi, Larsson 2014; Sun *et al.* 2013). For the signal shape and transmitter recognition algorithms there are needed huge data bases in which the signal references or transmitter parameters are stored (Perera, Herath 2011). Detectors based on application of matched filters have big collections of different filter sets, which need to match various primary user signals. The least amount of prior knowledge is used in detectors based on estimation of signal cyclostationary features (Sabat *et al.* 2010). These detectors needs to extract only the values of primary user signal carrier frequency periodicity.

Main disadvantage of the detectors, which needs prior knowledge is the lack of configuration flexibility and self-configuration capabilities. They are adapted for particular signals or signal types. Therefore, in the wide-band RF spectrum sensing applications they should have large data sets available to perform efficiently.

The cyclostationary feature estimation based detectors in comparison with other detectors, that use prior knowledge, has significantly lower amount of information about the primary user signal. Therefore, in this thesis the detectors based on extraction of cyclostationary features should be compared with energy detectors and Wavelet transform based detectors (in Fig. 1.1 marked gray). In order to compensate the lack of prior knowledge about the radio signal (e.g. no signal data base is available), the possibility to apply intelligent methods, such as neural networks and self-organizing maps, in combination with spectrum detectors should be investigated in dissertation.

1.2. Spectrum Sensors based on the Signal Energy Estimation

Detectors based on energy estimation consist of three parts (Fig. 1.2):

- Receiver with a band-pass filter. This is a part of the receiver, where there is determined, in which radio spectrum areas detector will operate. The band-pass filter also defines the width of the analyzed spectrum band.
- Energy estimator (marked as $|x|^2$ in Fig. 1.2). In this part the features (signal parameters) are estimated for selected spectrum sub-band (Bin *et al.* 2008; Imani *et al.* 2011).
- Decision maker (marked as THD in Fig. 1.2). This part is used to make a hypothesis about the spectrum occupancy or primary user presence. The decision is made by analyzing estimated energy parameters (Rasheed *et al.* 2010; Srinu, Sabat 2010).

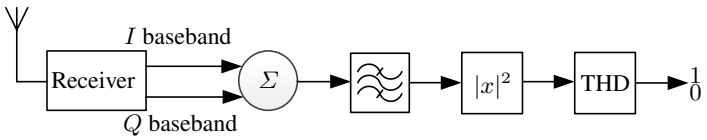


Fig. 1.2. Energy estimation from radio samples structure

The configuration of all spectrum sensor parts can make the influence to detector's performance. Therefore, all these parts must be discussed.

1.2.1. Signal Energy based Features Extraction

Usually the energy detectors are based on the estimation of one of three parameters or all of them (Elramly *et al.* 2011; Sun *et al.* 2015):

- average of the power spectrum;
- variance;
- standard deviation.

The parameters are estimated for currently analyzed sub-band. According to the estimated parameter values and their changes in time, the detection of primary user signal is performed in the later stages of the detector. Each radio sample x_n consist of primary user signal component u_n (if it is present) and noise G_n component (Arshad, Moessner 2010; Chen *et al.* 2011; Elramly *et al.* 2011; Yi *et al.* 2009):

$$x_n = u_n + G_n. \tag{1.1}$$

In most cases, the radio signal samples can be composed only from noise components $x_n = G_n$. If the statistics of radio signal noise is same or similar to additive white Gaussian noise (AWGN), then the signal feature based on calculation of average of signal spectrum components could be effectively applied in detector. Because the average of AWGN is close to zero (Chen *et al.* 2011), the average value of the spectrum components for the analyzed sub-band is estimated according to equation:

$$\overline{x_n} = \frac{1}{N} \sum_{n=0}^{N-1} x_n^2, \tag{1.2}$$

where n is the sample number, N total number of samples.

Average of the signal spectrum components gives worse detection results when the signal has a noise of different type, comparing to AWGN (Zhuan *et al.* 2008). Therefore, the detector based on this feature can be inefficient in real radio environments, where the statistics of noise can vary in time and even change it's type to impulsive (Sanchez *et al.* 2007). To overcome this limitation, additional parameters must be calculated for sub-band, such as variance and standard deviation (Bin *et al.* 2008; Zhuan *et al.* 2008). The analysis of these two estimated features can highlight changes in sub-band and also indicate the presence of primary user signal.

In order to estimate the variance D and standard deviation σ, the average value of signal spectrum components should be calculated in advance. Therefore, D or σ can be used in combination with $\overline{x_n}$:

$$D = \frac{1}{N} \sum_{n=0}^{N-1} (x_n - \overline{x_n})^2. \tag{1.3}$$

Main difference between D and σ used as features from implementation perspective is the square root operation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x_n - \overline{x_n})^2}. \tag{1.4}$$

In order to estimate σ, an additional operation is needed, which prolongs the calculation of this feature. Therefore, the application of σ has significant disadvantage comparing to application of D. In addition, D and σ estimates can be calculated only after average $\overline{x_n}$ is already calculated. Therefore, there could be investigated new modifications of σ or D calculation implementations in order to estimated these, or similar parameters in parallel with average $\overline{x_n}$ estimation.

1.2.2. Decision Threshold Selection Methods

After the calculation of signal features for energy detector, the decision about RF band occupancy must be made (Penna *et al.* 2009; Yixian *et al.* 2010). However, in most cases it is hard to decide which decision threshold THD should be selected for various RF spectrum environments (Bin *et al.* 2008; Larsson, Skoglund 2008). There are several models proposed to solve this problem and also some simulations were made to prove the proposed solution (Joshi *et al.* 2011b; Srinu, Sabat 2010; Yonghong *et al.* 2008; Zhang *et al.* 2011). However, only few experimental researches were performed taking the real measured signal data (Cabric *et al.* 2006; Kyungtae *et al.* 2010).

In order to select proper THD value, first there should be evaluated: the changes of the RF noise (Bin *et al.* 2008), the shadow fading effects (Rasheed *et al.* 2010) and the changes of location (Arshad, Moessner 2010). In order to overcome these factors, a cooperative sensing approach (Chen *et al.* 2011; Yi *et al.* 2009) and adaptive threshold selection approach were suggested (Imani *et al.* 2011; Joshi *et al.* 2011b). However, the performance of these methods was not tested in real RF environments and the efficiency of these methods application in changing RF spectrum is questionable.

One of important issues that should be taken into account when the THD selection algorithm is considered, is the complexity of algorithm implementation. Cooperative sensors systems are too complex and consists of many receivers (Yi *et al.* 2009). The implementation of these systems requires more computational resources and are less energy efficient. Therefore, further development of the spectrum sensor by adding additional signal analysis algorithms would be limited. In addition, the optimal THD selection should be oriented in to improving the accuracy and complexity of single node sensors.

1.2.3. Spectrum Sensors for Single Band Analysis

Quadrature parameters can be calculated directly after band-pass filter (Arshad, Moessner 2010; Bin *et al.* 2008; Imani *et al.* 2011). These parameters will be valid just in band-pass filter frequency range (Fig. 1.3). This approach is most common in currently most popular systems like wireless access points, bluetooth, zigbee and etc. Main disadvantages of this approach are frequency operation range and channel width accuracy inflexibility, because they are limited by filter parameters. In order to change spectrum sensing system operating bandwidth, a band-pass filter parameters should be changed. Therefore, in this approach the energy estimation should be performed in the fixed bandwidth.

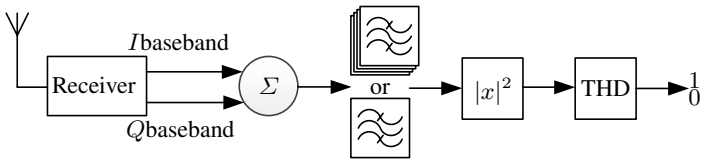


Fig. 1.3. Narrow-band energy detector structure

Bandwidth inflexibility can be overcome by using band-pass filter bank with different filter parameters (Fig. 1.3)(Farhang-Boroujeny 2008). System can switch between filters and work in different bandwidths. However, still with this filter bank energy estimation can be made just for one channel. To have more than one channel more additional filters are needed. Therefore, good bandwidth and channel selection flexibility can be achieved with large filter banks. However, large filter banks would require additional hardware and memory resources and the implementation of the filter bank based solution would be inefficient. In addition, such spectrum sensing systems have lower flexibility, because only small filter banks are used in practical applications.

1.2.4. Spectrum Sensors for Analysis of Multiple Bands

More commonly detectors calculations are implemented after Fourier transform (Fig. 1.4). Especially this way of implementation is efficient on FPGA and systems on a chip (Srinu *et al.* 2011; Wen-Bin *et al.* 2011). The selection of these chips gives a possibility to implement large FFT. After selection of the wide band-pass filter, a large FFT can help to achieve accurate RF spectrum, which can be divided in sub-channels for further analysis. Therefore, width and accuracy of sub-channels are determined by the size of FFT. For example, a 50 MHz band-width with 16 384 points FFT can be divided in to sub-channels

of approximately 20 kHz width. In comparison, in order to achieve the same sub-channel accuracy, a filter bank approach would require 250 filters.

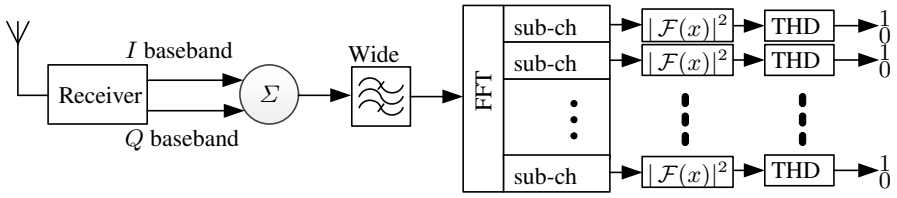


Fig. 1.4. Wide-band energy detector structure

FFT coefficients $\mathcal{F}(x_n)$ are calculated from radio signal x_n, which is filtered with wide band-pass filter (Fig. 1.4):

$$\mathcal{F}_f(x_n) = \sum_{n=0}^{N-1} x_n e^{-i2\pi fn/N}, \tag{1.5}$$

where N is the number of FFT elements, n is the number of radio signal sample. The radio spectrum coefficients from (1.5) are grouped in sub-bands, which size can be chosen from few kHz to MHz. Accuracy and size of sub-band is limited by FFT size. Therefore, this approach is more flexible than parameter calculation straight after band-pass filter.

1.3. Spectrum Sensors based on the Wavelet Transform

Discrete wavelet transform is a linear operation, which transforms data vector into two different frequency components (Fig. 1.5). Lower and higher frequency components of the signal are separated by two filters. Filter impulse response characteristics are determined by wavelets. Each wavelet transform has its unique filter coefficients. However, they properties may vary. Extensively wavelet transforms are used for image processing (Janušauskas *et al.* 2005; Valantinas *et al.* 2013; Zhao *et al.* 2014a) signal classification (Kannan, Ravi 2012), signal generation (Čitavičius, Jonavičius 2006) and also for spectrum sensing (Youn *et al.* 2007). Three different wavelet transforms are discussed in this section:

- Haar;
- Daubechies;
- Symlet.

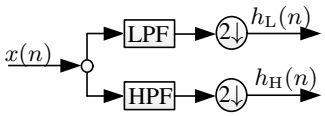


Fig. 1.5. One stage wavelet transform structure

where $2 \downarrow$ marks the downsampling by two times, LPF is a low-pass filter and HPF is a high-pass filter.

1.3.1. Wavelet Transform based Feature Extraction

Wavelet transform can extract more features from radio environment than energy estimation (Ariananda *et al.* 2009; Tian, Giannakis 2006). The time-frequency analysis of radio environment can be performed using this transform. The wavelet transform can be calculated directly after the application of the band-pass filter (Fig. 1.6) (Chu *et al.* 2014; Zhao *et al.* 2014b), or after the FFT is performed.

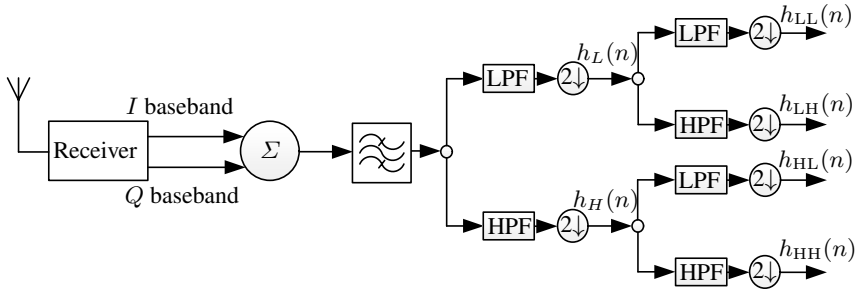


Fig. 1.6. Wavelet transform of I/Q data after band-pass filter

The structure of the spectrum sensor with wavelet transform based feature extraction, performed after application of band-pass filter, is similar to energy estimator based on a filter bank (Fig. 1.3). In order to implement wavelet transform based spectrum sensor, a large filter bank is needed (Adoum *et al.* 2010). The main disadvantage of this approach is the same as in similar energy estimation based solution. In order to obtain good sub-channel resolution (few kHz) in the wide-band, large filters banks should be used. The number of these filters in wavelet transform based spectrum sensor can be lowered by the use of the multiresolution feature (Jaspreet, Rajneet 2014). Wavelet transform filters (Fig. 1.5) can be connected into a series of filters, providing better resolution

in lower, higher or both frequency ranges (Jaspreet, Rajneet 2014; Wang *et al.* 2009). The multiresolution feature give ability to process less important RF bands by using less number of filters and more filters could be applied for the processing of the important RF bands (Fig. 1.7). However, such filter bank adjustment reduces the flexibility of the spectrum sensing system, because the filters are adjusted only for a particular RF band.

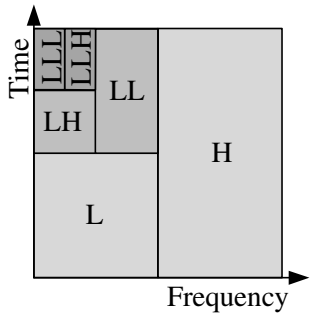


Fig. 1.7. Example of wavelet transform multiresoliution property

The spectrum sensor, based on a wavelet transform, performed after estimation of FFT, can be implemented using a small set of filters (Fig. 1.8). The wavelet transform based Smoothing, edge detection and detection of singularities are applied to RF spectrum – FFT coefficients (Chantaraskul, Moessner 2010; Mathew *et al.* 2010; Tian, Giannakis 2006). In such spectrum sensor the detection of the RF spectrum changes is performed only in frequency domain, when the FFT is already performed. Therefore, the wavelet transform is applied to the changes of FFT coefficients in time (Szadkowski 2012).

In time domain FFT coefficients are filtered by wavelet transform into 2–3 groups of fast, intermediate speed and slowly changing spectrum components (Dinesh Kumar, Thomas 2003). In this case, a RF spectrum noise can be filtered out, significantly lowered or concentrated in high-pass filter branch. In most cases noise components are quite different in neighboring time moments. The main disadvantage of this spectrum sensor structure from the implementation viewpoint is the requirement to use additional memory blocks to store the coefficients of FFT.

The application of wavelet transform give the ability to analyze the FFT spectrum in different slices. Therefore, more explicit signal features can be extracted in comparison to energy estimation based solutions.

An application of the wavelet transform after FFT gives better accuracy of feature extraction in comparison to a solution where the wavelet transform is performed after a band-pass filter. In order to receive the same accuracy, the

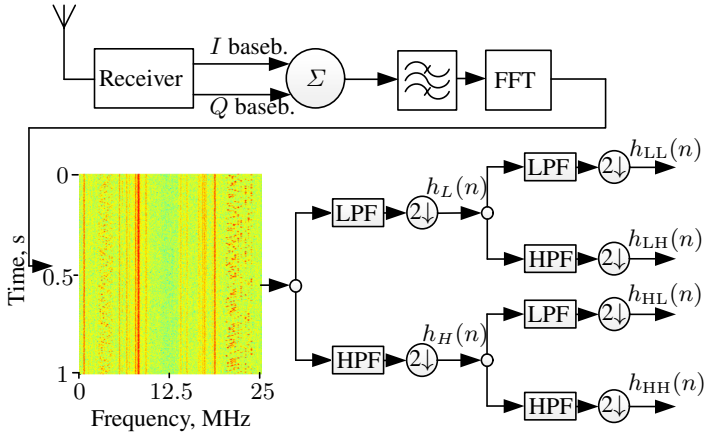


Fig. 1.8. Wavelet transform estimation structure after FFT

narrower sub-channel wavelet transform filters should be used and it significantly increases the complexity of implementation.

1.3.2. Haar Wavelet Transform

Haar wavelet is the least complex and oldest wavelet. First time it was mentioned by Haar in 1920. It consist of mother and scaling wavelets.

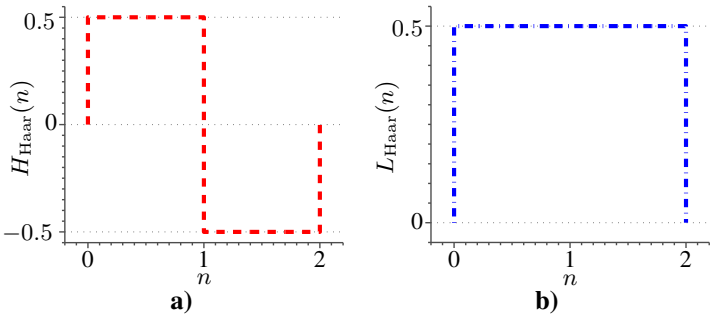


Fig. 1.9. Haar: a) mother; b) scaling wavelets

Mother wavelet compares two neighboring samples of the input signal by calculating the difference $x(n) - x(n-1)$ (Fig. 1.9a). Scaling wavelet scales two neighboring samples of the input signal by calculating average of $x(n)+x(n-1)$ (Fig. 1.9b). Therefore, the mother wavelet and scaling wavelet represents high-pass and low-pass filters in time domain.

Haar wavelets can be represented by the following expressions:

$$H_{\mathrm{Haar}}(n) = \begin{cases} 1/2, & 0 \leq n < 1; \\ -1/2, & 1 \leq n < 2; \\ 0, & \text{otherwise.} \end{cases}, \tag{1.6}$$

$$L_{\mathrm{Haar}}(n) = \begin{cases} 1/2, & 0 \leq n < 2; \\ 0, & \text{otherwise.} \end{cases}. \tag{1.7}$$

To summarize discrete Haar low-pass filter, it is expressed in frequency domain:

$$h_{\mathrm{L}}^{\mathrm{Haar}}(f) = \frac{\mathcal{F}_f(x) + \mathcal{F}_{f-1}(x)}{2}, \tag{1.8}$$

where f is the FFT coefficient frequencies. A high-pass filter is expressed in frequency domain by equation:

$$h_{\mathrm{H}}^{\mathrm{Haar}}(f) = \frac{\mathcal{F}_f(x) - \mathcal{F}_{f-1}(x))}{2}. \tag{1.9}$$

In time domain, the Haar filters are expressed by equation:

$$h_{\mathrm{L}}^{\mathrm{Haar}}(n) = \frac{x(n) + x(n-1)}{2}, \tag{1.10}$$

$$h_{\mathrm{H}}^{\mathrm{Haar}}(n) = \frac{x(n) - x(n-1)}{2}, \tag{1.11}$$

where n indicate the signal sample number.

In practical applications the Haar wavelet transform is usually used for spectrum compression (Egilmez, Ortega 2015) and spectrum edge detection (Karthik *et al.* 2011). The main motivation to apply this wavelet transform for spectrum sensing is a low computational complexity (Stojanović *et al.* 2011). However, in order to achieve better detection performance on a wavelet transform based spectrum sensor, a prior knowledge about RF spectrum is needed, unless some modifications of the spectrum sensing algorithm are performed.

1.3.3. Daubechies Wavelet Transform

Daubechies wavelet transform is the result of Ingrid Daubechies work. The wavelet transform has maximum vanishing moments of all wavelet transforms, analyzed in thesis (Valantinas *et al.* 2013). Therefore, more complex signals can be represented with less number of wavelet transform coefficients. However, the smoothness of the signal is lost (Satiyan *et al.* 2010). The mother

and scaling wavelets are shown in Fig. 1.10. The coefficients of the mother wavelets are used in a high-pass filter, according to (1.12) and (1.14) equations. The coefficients of the scaling wavelets are used in low-pass filter, according to (1.13) and (1.15) equations. The coefficients of the FFTcan be transformed by Daubechies wavelet according to the frequency and time (Fig. 1.8).

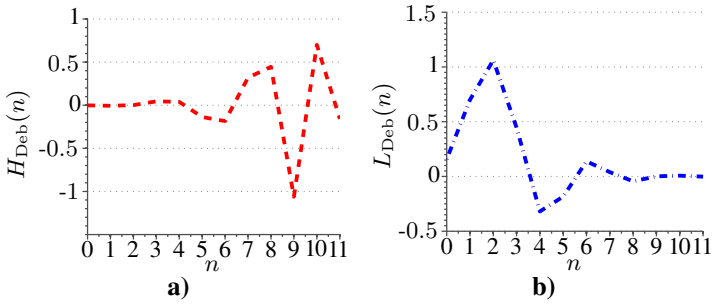


Fig. 1.10. Daubechies: a) mother; b) scaling wavelets

Daubechies transform filters in frequency domain are given in following equations:

$$h_{\mathrm{L}}^{\mathrm{Deb}}(f) = \sum_{n=0}^{W-1} H_{\mathrm{Deb}}(n)x(f-n), \tag{1.12}$$

$$h_{\mathrm{H}}^{\mathrm{Deb}}(f) = \sum_{n=0}^{W-1} L_{\mathrm{Deb}}(n)x(f-n), \tag{1.13}$$

where W is the frequency window, which is used for transform. Daubechies transform filters in time domain are given in following equations:

$$h_{\mathrm{L}}^{\mathrm{Deb}}(t) = \sum_{n=0}^{T-1} L_{\mathrm{Deb}}(n)x(t-n), \tag{1.14}$$

$$h_{\mathrm{H}}^{\mathrm{Deb}}(t) = \sum_{n=0}^{T-1} H_{\mathrm{Deb}}(n)x(t-n), \tag{1.15}$$

where T is time period, which is used for transform.

Like the Haar wavelet transform, the practical applications of the Daubechies transform includes spectrum compression and spectrum edge detection. However, the better detection performance can be achieved by using the Daubechies wavelet transform (El-Khamy *et al.* 2013; Mathew *et al.* 2010).

1.3.4. Symlet Wavelet Transform

Symlet wavelet transform is similar to Daubechies wavelets. This wavelet transform is a modification of Daubechies wavelet transform, which helps to achieve better symmetry, orthogonality and biorthogonality (Aboaba 2010; Yadav *et al.* 2015). Symlet wavelets are also suggested by Ingrid Daubechies. The mother and scaling wavelets are shown in Fig. 1.11. Like in Daubechies wavelet transform case, the coefficients are used with high-pass filter equations (1.12), (1.14) (mother wavelet) and low-pass filters equations (1.13), (1.15) (scaling wavelet). The coefficients of the FFT also can be transformed by Symlet wavelet transform according to the frequency and time (Fig. 1.8).

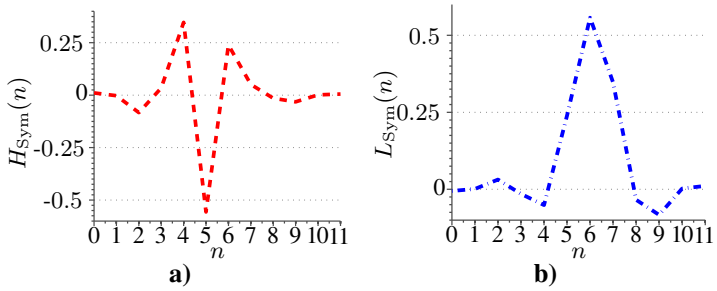


Fig. 1.11. Symlet: a) mother; b) scaling wavelets

In practical applications the Symlet wavelet transform is also used for spectrum compression and spectrum edge detection (El-Khamy *et al.* 2013; Mathew *et al.* 2010). The performance of the Symlet wavelet transform for spectrum sensing applications is similar to the performance received using Daubechies wavelet transform.

1.4. Spectrum Sensors based on the Cyclostationary Features

The cyclostationary features are present in communications signals as a side product of the transmitting systems (e.g., the signal carrier in AM). It can be detected after short observation of the RF signal (Cohen *et al.* 2011; Lin, He 2008; Yano *et al.* 2011). In addition, the spectrum sensors, based on Cyclostationary Features Extraction (CFE) may search for some periodicity in the primary user signal. Those features are extracted from signal mostly by applying the correlation after the FFT is performed (Fig. 1.12). in order to accurately extract Cyclostationary features, the results of the correlation should be

averaged over time T (Ali Tayaranian Hosseini *et al.* 2010; Du, Mow 2010; Feng, Bostian 2008). The CFE based spectrum sensing methods can make decision, which signal is in the analyzed channel. However, the prior information about signal or it's features is needed for this type of spectrum sensors (Choi *et al.* 2009; Javed, Mahmood 2010; Lin *et al.* 2009).

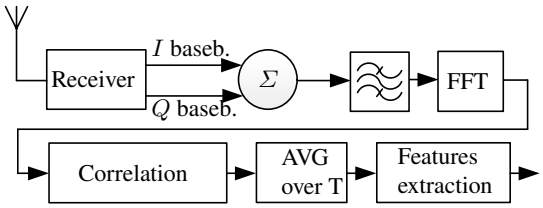


Fig. 1.12. Spectrum sensor structure based on cyclostationary features extraction

The main advantage of the CFE based spectrum sensors (comparing to energy estimation and wavelet transform based methods) is the ability to extract a signal from noisy RF spectrum band. This advantage is achieved, because in most cases the noise is not correlated with itself and the communication signal is (Umebayashi *et al.* 2011; Wang *et al.* 2010). The features from the signal spectrum are extracted by applying the Discrete Spectral Correlation Function (DSCF):

$$S_f^W = \frac{1}{N} \sum_{n=0}^{N-1} \mathcal{F}_{f+W}(x_n)\, \mathcal{F}_{f-W}(x_n)^*, \tag{1.16}$$

where $*$ in (1.16) equation marks a complex conjugate, W is the cycle frequency, f is the spectral frequency. The complexity of DSCF lays in $\frac{1}{4}N^2$ number of complex multiplications needed. In comparison, in order to calculate FFT, a $N(\log_2 N)$ complex multiplications are needed (Derakhshani *et al.* 2010). E.g. if the CFE should be calculated for 1024 FFT points, 25 complex multiplications are required.The spectral coherence coefficient (SCC) between signal components $f \pm W$ for the DSCF is calculated by equation:

$$C_f^W = \frac{S_f^W}{\sqrt{S_{f+W/2}^0 S_{f-W/2}^0}}. \tag{1.17}$$

The SCC can vary form 0 to 1. Which modulation of signal is in the channel can be decided by combining DSCF and SCC. However, for the identification, the prior knowledge about the features of the primary user signal is needed (Derakhshani *et al.* 2010; Kokkeler *et al.* 2007; Kyouwoong *et al.* 2007).

The requirement to have some prior knowledge about the primary user signal and the computational complexity are biggest disadvantages of the CFE based spectrum sensors.

1.5. Intelligent Spectrum Sensors

During the spectrum sensing application, the decision should be made about channel occupancy according to the estimated features (energy based features, wavelet transform based features or cyclostationary features). In the currently proposed spectrum sensing solutions based on artificial neural networks, the prior knowledge about radio spectrum (what signals are in RF band, what parameters they have) is required for estimation of the sensor parameters (Yu-Jie *et al.* 2010). However, such approach is not suitable enough in the radio environments where regulation of signal population is not strict and it is difficult to have a good prior knowledge about the signals in RF bands (especially in unlicensed ones). In these cases, the self learning intelligent decision-making methods may have more advantages (Fig. 1.13) (Matuzevičius *et al.* 2010; Popoola, Van Olst 2011; Tumuluru *et al.* 2010). In addition, the supervised training can be applied using features, that can be used with different types of signals in different RF environments. After the training of the neural network is performed, the intelligent spectrum sensor may successfully classify RF spectrum.

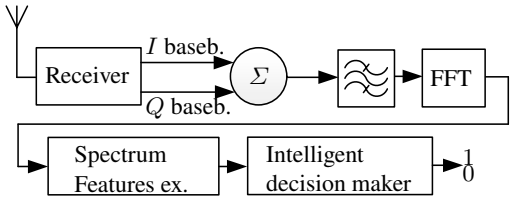


Fig. 1.13. Spectrum sensor structure based on intelligent decision maker

1.5.1. Spectrum Sensing based on Artificial Neural Network

One type of the intelligent decision-making methods is based on an artificial neural network (ANN). The way of information processing in ANN is based on biological nervous systems, like a human brain. First experiments with ANN were performed in 1943 by W. McCulloch and W. Pits. However, in that time the solution was too simple for wide spread of ANN. More significant results

in ANN research were achieved by F. Rosenbllat in 1957. After that year, more and more theoretical and practical research results were received in this area (Sledevič, Navakauskas 2015; Tamaševičiūtė *et al.* 2012).

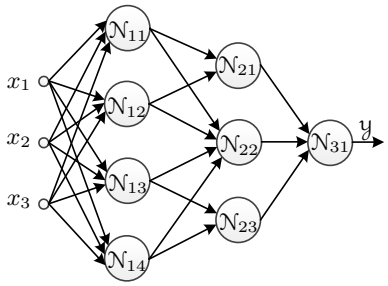


Fig. 1.14. Multilayer feedforward neural network structure

The ANN consist of interconnected processing elements – Neurons. They are working in a cooperation in order to solve specific problem or task. Mostly ANN are used for the specific types of applications: classification, pattern recognition and function approximation. The parameters of the network are adapted for a specific task through the training process. In ANN, the training state weights are adjusted for all neurons like in human brain (Paukštaitis, Dosinas 2010; Pikutis *et al.* 2014; Ramirez *et al.* 2011; Vaserevičius *et al.* 2012).

The training of the ANN is performed an an example or example set: measurement results, features extracted from images or statistical data. The successfully trained ANN have an ability to find the meaning even in inexact or corrupted data. The main requirement for training data is that it must represent a state of the measured system in all possible situations (Ivanovas, Navakauskas 2012; Laptik, Navakauskas 2005; Levinskis 2013). When the ANN is trained properly it can be used as an expert for a particular task (Plonis *et al.* 2005).

The ANN has few advantages in comparison to ordinary decision-making algorithms (Katkevičius *et al.* 2012; Serackis, Navakauskas 2008):

- the ANN has an ability to train from given data;
- some types of ANN may self-organize it's structure during training process;
- the calculation of the ANN coefficients can be accelerated by FPGA, because the mathematical operations can be implemented for processing in parallel.

Therefore, the decision-making solution based on an ANN is more superior than the threshold based or prior knowledge based algorithms. All at-

tributes of ANN gives more versatility for network to decide about the occupancy of the channel even in non-standard radio environment.

1.5.2. Architectures of the Artificial Neural Network

Two ANN architectures were analyzed in this work: single-layer and multilayer feedforward networks (Fig. 1.15). The recurrent neural networks were not considered, because the ANN architecture with recurrent connections between neurons lacks of stability in comparison to feedforward networks (Huaguang *et al.* 2014). Especially the FPGA based implementation of the recurrent neural network is a challenging task (Sicheng *et al.* 2015).

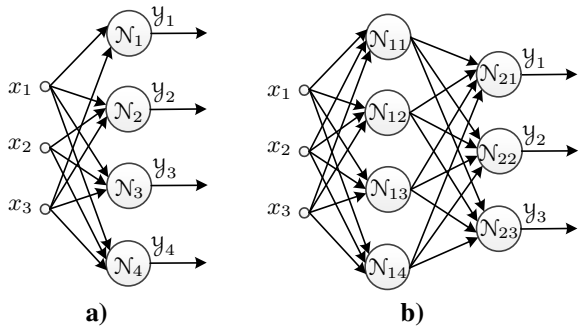


Fig. 1.15. Single layer and multilayer feedforward neural network structures

The single-layer feedforward network is a simplest form of neural network. This architecture can be used in applications, where each neuron can represent unique situation in the radio environment (Yu-Jie *et al.* 2010). Such structure of the ANN may be used where the problem do not require higher neural network complexity (Ramirez *et al.* 2011).

The structure of the Multilayer feedforward network can be divided into tree parts: input layer, output layer and the hidden layer. One or more hidden layers help network to obtain higher-order statistics from RF spectrum parameters (Liang *et al.* 2011; Xiang-lin *et al.* 2008; Zhao *et al.* 2011).

Main component of the ANN is a single artificial neuron, which can be used separately as a decision-maker or, in large networks, as a single node of the neural network. If the neuron is used as a single decision-maker, it uses a weighted sum of RF sub-channel parameters, passed to an activation function (Yu-Jie *et al.* 2010). In the larger neural network structures the purpose of the neuron can vary depending on a layer in which it is situated. The structure of a single neuron based decision-maker is shown in Fig. 1.16.

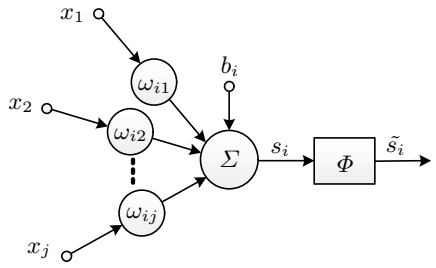


Fig. 1.16. Single neuron structure

During the estimation of the weighted sum, all inputs x_j are multiplied by a different weight variable ω_{ij}. In spectrum sensor, the inputs of the neuron can be the energy values of the RF spectrum, the outputs of the wavelet transform or the estimated cyclostationary features of the signal (Li *et al.* 2010). The multiplication of the inputs with the corresponding weights can be performed in parallel, because all multiplication operations can be made separately:

$$x_1\omega_{i1}, x_2\omega_{i2}, ..., x_n\omega_{ij}, \tag{1.18}$$

$$s_i = b_i + \sum_{j=1}^{N} x_j\omega_{ij}. \tag{1.19}$$

The summation stage s_i combines all weighed inputs and bias b_i. The bias helps for activation function to decrease or increase the input level. It gives more flexibility in interpretation of the network inputs. In the activation stage the weighted sum s_i is sent to an activation function:

$$\tilde{s_i} = \Phi(s_i). \tag{1.20}$$

The most frequently used in practical applications activation functions are: threshold (Heaviside) and sigmoid.

The threshold can take two values: 0 or 1 (Fig. 1.17) (Jun 2010):

$$\Phi_{\mathrm{T}}(s_i) = \begin{cases} 1 & s_i \geq 0; \\ 0 & s_i < 0. \end{cases} . \tag{1.21}$$

This activation function is more suitable for one neuron based decision-making solution. In addition, it could be used in the output layer of the multi-layer network to produce a final decision:

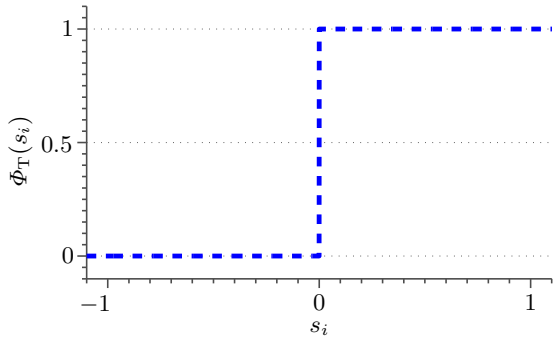


Fig. 1.17. Heaviside (threshold) function

- 1 sub-channel is used by primary user;
- 0 sub-channel is vacant.

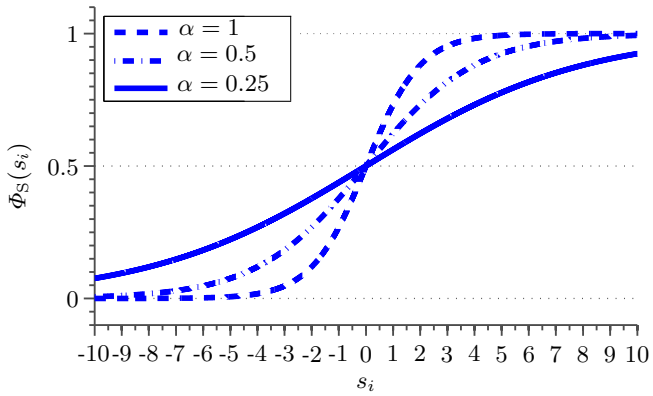


Fig. 1.18. Sigmoid function with diffrent slope coeficients

The Sigmoid function has S shape (Fig. 1.18). It is most common activation function in ANN. Sigmoid is defined as a strictly increasing function that has equilibrium between a linear and non-linear behavior (Fu *et al.* 2012). It can produce an output, which may vary from 0 to 1:

$$\Phi_{\mathrm{S}}(s_i) = \frac{1}{1 + e^{-\alpha s_i}}, \tag{1.22}$$

here α is a slope coefficient. If it reaches ∞ then the sigmoid function will act like a threshold function. In contrast to the threshold function it has a

derivative and can be used with gradient descent based training methods, such as least mean square (LMS) algorithm. Considering the properties of a sigmoid function, the function is more suitable for the inner layers of the multilayer ANN based spectrum sensor (Popoola, van Olst 2013).

1.5.3. Supervised Training of the Spectrum Sensor

The weights of the neural networks are estimated during training process. The single layer networks are usually trained by a LMS algorithm. However, the training of the multilayer network is more computationally complicated (Ibrahim 2008). One training iteration of the multilayer network consists of two stages: estimation of the output followed by calculation of the global error and error back-propagation.

The global error $e(n)$ is estimated by comparing the output of the ANN with the desired response:

$$e(n) = \mathcal{D}(n) - \Phi\left(\sum_{l=1}^{L}\sum_{i=1}^{I}\sum_{j=1}^{J}(x_{lij}(n)\omega_{lij}(n)) + b_{li}\right), \tag{1.23}$$

where $\mathcal{D}(n)$ is a desired output, l is the network layer index, i is the neuron index, j is the neuron weight index. The estimated global error is back-propagated to each neuron. Error is used for weight update:

$$\omega_{lij}(n+1) = \omega_{lij}(n) + \eta\frac{\partial e(n)}{\partial \omega_{lij}}x_{lij}(n), \tag{1.24}$$

where η is weight adjustment factor. The network convergence speed depends on η. It is clear from equation (1.24) that the implementation of the training algorithm should be adjusted for each network configuration, because the error expression will be different for each layer. The goal of the backpropagation training algorithm is to minimize the mean square error (MSE):

$$E = \frac{1}{N}\sum_{n=0}^{N-1} e^2(n), \tag{1.25}$$

when the algorithm reaches defined E boundary, the network weight updates are stopped.

The most challenging part in error back-propagation stage is calculation of the activation function derivative. For example, the derivative of the sigmoid function, used in inner ANN layers, is estimated according to the formulas:

$$\Phi_{\mathrm{S}}(s_i) = \frac{1}{1 + e^{-\alpha s_i}}, \tag{1.26}$$

$$\frac{\partial \Phi_{\mathrm{S}}(s_i)}{\partial s_i} = \Phi_{\mathrm{S}}(s_i)(1 - \Phi_{\mathrm{S}}(s_i)). \tag{1.27}$$

In order to implement (1.26) and (1.27) equations in the hardware, the large lookup tables or approximations of these should be used (Aliaga *et al.* 2008). The implementation of the whole backpropogation algorithm in most cases should be adapted to particular ANN architecture and task, solved by the network.

1.5.4. Unsupervised Training of the Spectrum Sensor

Self organizing map (SOM) is a neural network, for which the weights are estimated during a self-training process. Usually no initial knowledge about the desired response of the network is used. This is main difference between SOM and the ANN trained by the supervised algorithms (Stefanovic, Kurasova 2011; Villmann *et al.* 2011). SOM is widely used in classification or clustering tasks (Ivanikovas *et al.* 2008; Serackis *et al.* 2010; Stankevičius 2001; Stefanovič, Kurasova 2014). In addition, there are several examples of SOM application for prediction (Merkevičius, Garšva 2007; Merkevičius *et al.* 2004) and reduction of data dimensionality (Kurasova, Molytė 2011). The self-training algorithm of the SOM is based on competition (Pateritsas *et al.* 2004). First, all neurons compete with each other in order to be selected as a winner neuron. Next, the weights of the winner neuron and it's neighbors are updated in order to increase the outputs for these neurons.

SOM has an advantage against classical ANN in RF environments where the situation is unpredictable, especially in noisy and unlicensed spectrum areas (Cai *et al.* 2010; Khozeimeh, Haykin 2012). In spectrum sensing applications, the SOM can perform an interpretation of the statistical parameters during self-training process, without knowledge that some input features indicates the presence or absence of the primary user. Therefore, this type of network can be self-trained even for harsh radio surroundings (Yang *et al.* 2012, 2014).

The most common SOM structures used in practical applications are one dimensional and two dimensional (Fig. 1.19). The practical applications of the higher dimension SOM are rare. There are two strategies for self-training of the SOM: the winner-takes all and the neighborhood. In the first strategy only the winner neuron weights are adjusted during self-training process (Nascimento *et al.* 2013). In the second strategy, the weights of the winner and some neighboring neurons, selected according to the on neighborhood function, are

updated (Herrmann, Ultsch 2007; Sharma, Dey 2013). In this case more than one neuron are updated during one self-training iteration.

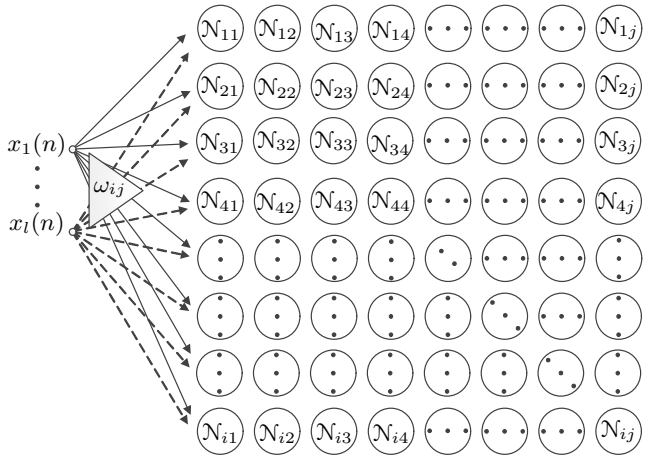


Fig. 1.19. Self organizing map 2 dimensional structure

The winner neuron is selected by estimating Euclidean distance between the input and current SOM weights:

$$\mathcal{N}^{\perp}(\boldsymbol{x}) = \arg\min_{ij} ||\boldsymbol{x} - \boldsymbol{\omega}_{ij}||. \tag{1.28}$$

The winner neuron is selected according to the minimal Euclidean distance (Gunes Kayacik *et al.* 2007). the weights of the neuron are updated with self-training ratio η, which is exponentially decreased during the self-training process. The current value of η depends on two parameters: initial self-training ratio η_0 (usually $\eta_0 \leq 1$) and time constant τ, which controls the slope of the exponent:

$$\boldsymbol{\omega}_{ij}(n+1) = \boldsymbol{\omega}_{ij}(n) + \eta(n)(\boldsymbol{x}(n) - \boldsymbol{\omega}_{ij}(n)), \tag{1.29}$$

$$\eta(n) = \eta_0 e^{-\frac{n}{\tau}}. \tag{1.30}$$

In summary, SOM self-training algorithm can be divided into four steps:

1. Initialization of the weights. Small random values (different) are assigned to all neurons' weights.
2. Competition. The Euclidean distance is estimated for each neuron according to equation (1.28).
3. The update of the weights. The weights of the winner neuron are updated according to equation (1.29).

4. Repetition of the 2nd and 32rd step. The Self-training ratio is updated and algorithm is started from second step over again.

The main advantage of SOM in comparison to alternative ANN is a low complexity of the digital implementation (Dlugosz *et al.* 2011; Tamukoh *et al.* 2004). SOM doesn't have complex activation functions (Appiah *et al.* 2012; Oba *et al.* 2011) and the estimation of neuron output can be performed easier, in parallel to output estimation for other neurons (Caner *et al.* 2008; Franzmeier *et al.* 2004).

1.5.5. Unsupervised Training using Neighborhood Neurons

In neighborhood self-training strategy more neurons are involved in comparison to winner-takes all strategy (Stankevičius 2001). Therefore, the strategy with neighboring neurons has a greater potential to converge faster. Topology based structure, for which in the center is a winning neuron, is created and adjusted during the self-training process (Fig. 1.20 and 1.21). Winning neuron and it's neighbors are self-trained using different self-training ratio η. The more distant neighbors have lower initial value of η (e.g., η_1, η_2, η_3). Therefore, the changes of neuron weights during the self-training iteration is less significant (Zhang *et al.* 2013).

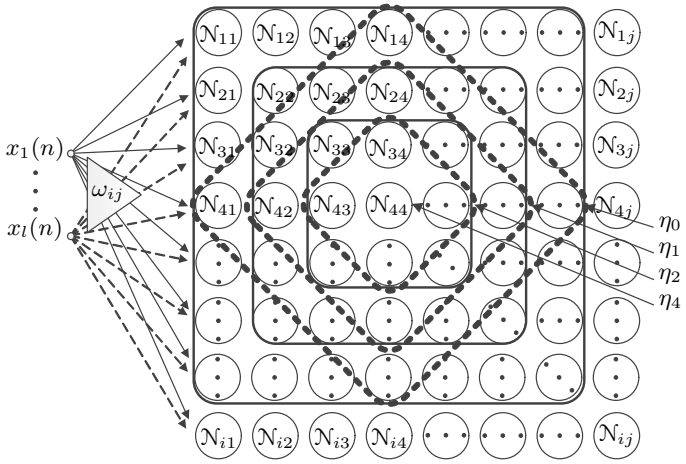


Fig. 1.20. Self organizing map square and rhombus neighborhood topologies

The weights of the neurons in self-training process with neighbors are updated according to the equation (2.4). The initial value of the self-training ratio $\eta(n)$ is changed for neighboring neurons to $\eta_{\mathrm{nb}}(n)$, according to equation

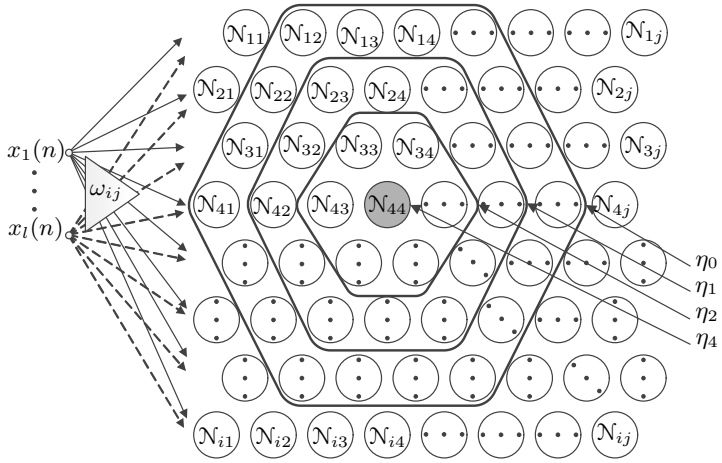


Fig. 1.21. Self organizing map hexagonal neighborhood topology

(1.31). Neighboring topologies can be square, rhombus (Fig. 1.20) and hexagonal (Fig. 1.21). Index nb designates the distance from the winner. When the distance is 0, it marks a winning neuron. The higher index is, the lower is self-training ratio is used. Therefore, in most cases $\eta_0 > \eta_1 > \eta_2 > \cdots > \eta_{\mathrm{nb}}$:

$$\boldsymbol{\omega}_{ij}(n+1) = \boldsymbol{\omega}_{ij}(n) + \eta_{\mathrm{nb}}(n)(\boldsymbol{x}(n) - \boldsymbol{\omega}_{ij}(n)). \tag{1.31}$$

Topologies differ from each other by the number of neurons taken into neighborhood. Most neighbors have square topology and least has rhombus. Hexagonal topology is nearest to radial neighborhood.

$$\eta_{\mathrm{nb}}(n) = \eta(n) h_{ij,\mathcal{N}^{\perp}(\boldsymbol{x})}. \tag{1.32}$$

Self-training ratio η_{nb} for the neighboring neurons can be decreased using Gaussian function, according to equations (1.32) and (1.33) (Gorunes cu *et al.* 2010). The self-training ratio η_{nb} depends on a distance estimate $d^2_{ij,\mathcal{N}^{\perp}(\boldsymbol{x})}$ exponentially. In addition, the convergence of the self-training depends on the parameter γ, which is also changing exponentially during self-training process. The slope of the changes depends on the chosen τ_{nb}:

$$h_{ij,\mathcal{N}^{\perp}(\boldsymbol{x})}(n) = \exp\Big(-\frac{d^2_{ij,\mathcal{N}^{\perp}(\boldsymbol{x})}}{2\gamma^2(n)}\Big), \tag{1.33}$$

$$\gamma(n) = \gamma_0 \exp\Big(-\frac{n}{\tau_{\mathrm{nb}}}\Big). \tag{1.34}$$

The application of neighborhood functions can increase the self-training speed of the SOM. However, the implementation of these functions requires additional algorithm blocks, in comparison to winner-takes all strategy (Manola kos, Logaras 2007; Yamamoto *et al.* 2011). Therefore, during the implementation, all SOM topologies should be optimized, especially the Gaussian function (Caner *et al.* 2008).

1.6. Conclusions of Chapter 1 and Formulation of the Tasks

1. Spectrum sensors are used to make a decision: do the primary user signal is present in the analyzed signal frequency band or not. Two types of spectrum analysis methods are most dominant in reviewed literature: methods based on the estimation of the channel noise level, methods based on the specific feature search in the received signal.
2. In the first type of methods, the channel noise level in the spectrum sensors is measured by estimating the received signal energy. A threshold of the detector is selected manually or calculated accordingly to the current transmission channel noise characteristics. The main advantage of the spectrum sensors based on signal energy estimation is a low computational cost, comparing to alternative spectrum sensing methods. The main disadvantage of the spectrum sensors based on signal energy estimation is the selection of the decision threshold.
3. The second type of methods uses different types of additionally estimated features during the signal pre-processing stage. The features are estimated by application of the specific signal transform (e.g. wavelet transform) or by the analysis of the signal cyclostationary features. For both types of the spectrum sensors, the threshold of the detector is used for the estimated values of each feature in order to make a final decision: do the primary user signal is present or not.
4. Wavelet transform based signal spectrogram feature extraction can increase the primary user emission detection rate in comparing with signal energy based features extractors.
5. Decision about radio spectrum occupancy can be made by applying pre-estimated threshold, or by using self-adapting intelligent methods like artificial neural networks or self-organizing maps. Self-adaption property of self-organizing maps can be efficiently used for various signal types detection, about which there is no prior knowledge.

Two hypotheses were formulated as a result of the performed literature review:

1. Wavelet transform based signal spectrogram feature extraction can increase the primary user emission detection rate in comparing with signal energy based features extractors.
2. Self-adaptation of the spectrum sensor to continuously changing radio environment can be achieved by the application of intelligent methods, modifying the training algorithms for implementation in embedded system.

In order to confirm both hypothesis, three tasks should be completed:

1. Investigate the application of wavelet transform to increase the performance of spectrum sensor.
2. Investigation of the neural network with binary activation functions suitability for spectrum sensing applications.
3. Development of the spectrum sensing method based on a self-organizing map and investigate its modifications to decrease the duration of self-training preserving the primary user emission detection performance.

2

Spectrum Sensing Methods Theoretical Researches

In this chapter the theoretical research results are presented. Two approaches are proposed for automatic selection of the spectrum sensor's threshold. In addition, the investigation results of the signal feature extractor modifications for efficient implementation and modification of the classical SOM topology to increase the self-training speed are also presented in this chapter.

The results of investigations, presented in this chapter are published in five papers (Stašionis, Serackis 2013; Stašionis, Serackis 2014; Serackis, Stašionis 2014; Stašionis, Serackis 2015a; Stašionis, Serackis 2015b) and presented at four international conferences (ELECTRONICS'2013; EUROCON'2013; NDES'2014 and EUROCON'2015).

2.1. Architecture of Spectrum Sensing System

The spectrum sensing system can be divided into two parts: signal receiver and digital signal processing (DSP) unit. There are two main components in the signal receiver part: IQ signal receiver and a band-pass filter. The characteristics of the IQ signal receiver defines the RF spectrum part, in which the spectrum sensor can operate. The DSP unit performs FFT, signal feature extraction and performs signal analysis based on estimated features. The DSP part may have different complexity of the signal processing algorithms and also may introduce different latency to the spectrum sensor output.

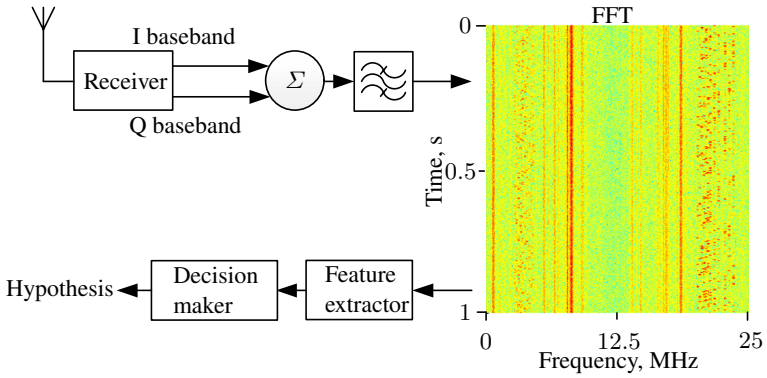


Fig. 2.1. Spectrum sensing system structure

In this dissertation the attention was focused to the development of the new and more effective signal processing methods for spectrum sensing applications. In addition, the methods, proposed in this dissertation were specially adapted for FPGA based implementation.

DSP part will be implemented in FPGA. Therefore, implementation of all components must be adapted to these technology. From these components depends:

- FFT block. On it depends accuracy of radio component estimation.
- Feature extraction. On this component depends quality of features, which will be extracted from radio spectrum. Main features which will be used in research is energy – variance estimates and wavelet transform.
- Decision maker. On this module depends accuracy of decision. Hypothesis about channel occupancy will be made by neural network and SOM.

As base of DSP part is used Zynq 7020 system on chip, which is a combination of ARM and FPGA. All implementations would be tested in this chip, therefore, all limitations which will be mentioned in design is dictated by Zynq 7020.

2.2. Radio Spectrum Feature Extractors in Field Programmable Gate Array

This section focuses on the implementation of signal feature extraction methods using field programmable gate array based embedded systems. Two al-

ternative FPGA based implementations of the FFT are compared. In order to efficiently implement selected signal spectrum feature extractors (reduce the number of calculations performed by a hardware), the modifications of the calculation formulas are proposed. A wavelet transform was considered as a feature extractor. Therefore, the FPGA based implementation of the wavelet transform is also analyzed in this section.

2.2.1. Analysis of the Different Implementations of Fast Fourier Transform

The FPGA based implementation of the wide-band FFT (Fig. 2.1) may have a high influence to the performance of the spectrum sensor. The amount of hardware resources, dedicated to wide-band FFT block, should be small enough to leave room form the implementation of the rest signal processing applications (Bury *et al.* 2008; Pupeikis 2015). Two types of FFT implementation are available in intellectual property (IP) core, found in Xilinx environment standard library:

- Pipelined implementation. In this implementation there are applied several Radix-2 butterfly processing engines. Data stream is pipelined through these processing engines and it allows to perform a continuous processing of the radio signals.
- Radix implementation. In this implementation only one Radix-4 butterfly engine is used. There is a possibility to use a Radix-2 butterfly engine, which is more suitable for less powerful FPGA based systems. In this implementation the radio signals cannot be processed continuously. The first signal frame should be fully loaded into the block input RAM. The coefficients of the FFT are available for further processing only after the complete estimation of FFT for the whole selected signal frame is performed.

Table 2.1. FPGA resource utilization dependency on FFT length

FFT length	DSP slice	Block RAM	Latency, μs	20 MHz BW,Hz	15 MHz BW,Hz
4096	45	11	27	9765	7324
8192	52	20	55	4882	3662
16 384	54	37	109	2441	1831
32 768	61	69	218	1220	915
65 536	63	134	437	610	457

The Pipelined implementation of FFT requires two times more resources. However, the latency of FFT estimation is two times lower than received using

Radix implementation. The main disadvantage of Radix implementation is that the input of the FFT block should have additional buffer to load the signal frame. Therefore, in case of higher input rates, the Radix implementation may suffer from data loss. Whereas, the pipelined implementation is able to process an unbuffered data streams taken directly from ADC.

Taking into account the ability to process an unbuffered data streams and the lower latency the pipelined FFT implementation was selected for spectrum sensor design. The dependency on the demand for hardware resources to the number of estimated FFT samples are shown in Table 2.1.

The latency, received in the FFT block, is directly proportional to the number FFT samples. For signal frame with 65 536 samples the latency of the FFT block reached $\sim$0.5ms. However, for signal frame with 4096 samples, the latency was just 27 μs (clock rate if the block was 300 MHz). The number of DSP slices did not vary so much like latency. The number of DSP slices, needed for FFT implementation, varied in the range from 45 to 63. The memory dependency increases proportionally to the length of FFT. The maximum number of 18 kb memory blocks, equal to 134, was required for the biggest analyzed signal frame.

The resolution of the estimated spectrum depends on the sampling frequency, which is selected accordingly to the analyzed bandwidth. Two bandwidths were compared during theoretical investigation: 20 MHz and 15 MHz. In larger bandwidths, such like 20 MHz and 15 MHz, more signal frame samples should be used. E.g. 16 384 samples provides signal spectrum resolution of 2440 Hz for 20 MHz frame and 1831 Hz for 15 MHz frame.

The maximum number of DSP slices (28%) and 18 kb block RAM (49%) was required to implement the FFT for 65 536 signal samples. The minimum suggested number of samples (16 384) will require 24% of DSP slices and 13% block RAM. The amount of hardware resources was calculated according to the FPGA based system, used in experimental environment.

If the FPGA based embedded system has enough block RAM components and the latency of $\sim$0.5 ms is acceptable for application, it is possible to use 65 536 signal samples for FFT. However, the FFT window can be reduced up to 16 384 in order to optimize RAM utilization and reduce the latency caused by calculation.

2.2.2. Implementation of the Energy based Feature Extraction

Energy based spectrum features extractors also called quadrature extractors are based on channel power calculation. Main task of these extractors is to highlight signals from noisy environments like shown in Fig. 2.2.

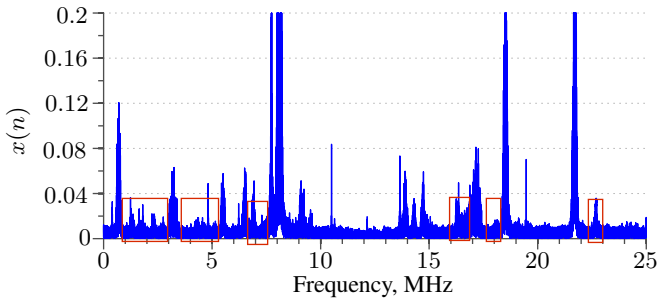


Fig. 2.2. 25 MHz bandwidth radio spectrum record

In presented radio spectrum are many signals, which are covered by random nature noise. Therefore, all energy extractors implementations will be tested with this radio spectrum sample. In this section will be tested on channel power average, variance, standard deviation and their modifications based extractors, which will be implemented in FPGA equations (1.2), (1.3), (1.4), (2.1), (2.2).

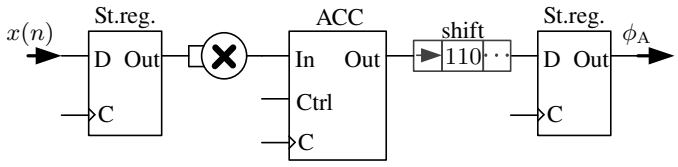


Fig. 2.3. Channel power average – ϕ_{A} implementation in FPGA

In Fig. 2.3 is shown coarse structure of power average extractor – ϕ_{A} FPGA implementation (1.2). In its beginning and its end are 2 state registers – St. reg. These registers set input and output in each rising edge of synchronization (clock) impulse. These registers are mandatory in each design, because they prevent systems from uncertain conditions. Another important element of this design is multiplier, which is implemented by DSP slice. This component quantity is very limited in hardware, therefore, all multiplication operations must be used carefully. After multiplication operation signal is accumulated in ACC component. Accumulator collects input samples for predefined window, which is equal to channel size. Channel length is iterative to 2^n, because in this case, there is no need of more DSP slice to calculate average. Division can be implemented by bit shifting operation. So if channel length is set to 8 spectrum samples, then to calculate its average value, accumulated sum must be shifted by 3 bits. This implementation latency is proportional to channel

length. If channel length is 8, then latency will be 12 clock cycles. 4 cycles are used for multiplication and signal setting, other 8 for accumulation.

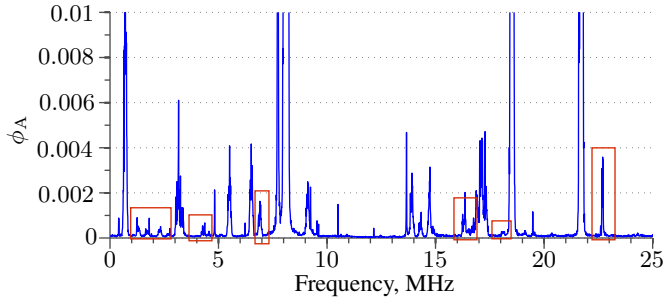


Fig. 2.4. 25 MHz bandwidth radio spectrum record after ϕ_A extraction

In test RF spectrum sample 6 areas are marked (Fig. 2.2), where signals level is to low. SNR for these areas vary from 1–5 dB. After ϕ_A (Fig. 2.4) noise level is lowered. In first marked spectrum case are 3 signals and for them SNR is raised up to 7 dB. In second is just one signal and for it SNR is achieved 5.5 dB. In 3,4 and 6 radio areas SNR lifted significantly up to 12 dB, 10 dB and 16 dB. Lowest SNR 5 dB is achieved in 5 area. This implementation of ϕ_A increased low level signals SNR by 4–11 dB.

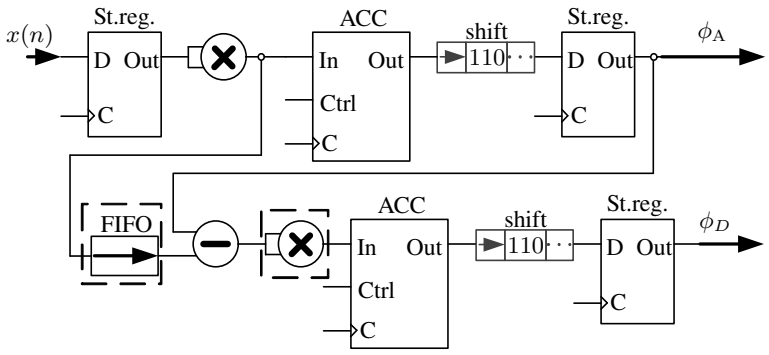


Fig. 2.5. Chanel power variance – ϕ_D implementation in FPGA

To ϕ_A (Fig. 2.3) can be added variance calculation part – ϕ_D (Fig. 2.5). It additional has First in first out – FIFO buffer and subtraction element. For channel variance calculation first mean value of same area must be estimated (equation (1.3)). Therefore, time gap forms between these calculations, which

is equal to average calculation window. FIFO buffer is used to delay variance calculation. This buffer length is a same as channel length. Through FIFO passed powered spectrum samples are delayed. Until average value for new channel is calculated, those delayed samples are subtracted from same channel average. So ϕ_{A} and ϕ_D calculations are paralleled. After subtraction ϕ_D implementation is similar to average structure. Main disadvantages of this variance implementation are:

- It has longer delay than average calculation. For 8 samples variance is calculated after 21 clock cycles. Additional delay is caused by FIFO buffer 8 cycles and subtraction operation 1 cycle.
- It uses additional multiplication operation. This operation can utilize more than 1 DSP slice.

By ϕ_D noise level was lowered even more, see Fig. 2.6. In 1, 2 and 5 marked areas low level signals SNR is lifted to 8–11 dB. And SNR is up to 4 dB more, than it was achieved with ϕ_{A}. Significantly better results are achieved in 3 and 4 areas. There SNR achieved 19 dB and 20 dB accordingly. In 6 signal case ϕ_D shown result worse by 1dB, than it was in average estimate. Yet overall ϕ_D performed better than ϕ_{A}.

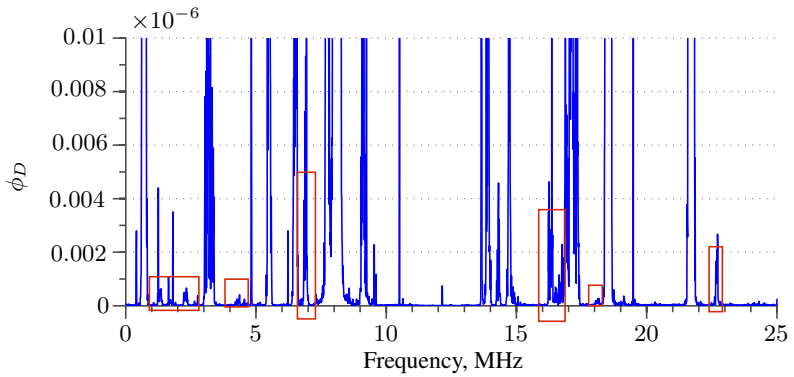


Fig. 2.6. 25 MHz bandwidth radio spectrum record after ϕ_D extraction

ϕ_D calculation is almost 2 times longer than ϕ_{A}, because FIFO buffer brings additional delay. This disadvantage can be corrected with one assumption, that the neighboring channel doesn't change much. With this idea variance – $\phi_{D_{\mathrm{N}}}$ or std. $\phi_{\sigma_{\mathrm{N}}}$ deviation can be calculated by using old or neighboring channel average value. If average value is used of neighboring channel $n-1$, then D_N will be:

$$D_N = \frac{1}{W}\sum_{f=0}^{W-1}(\mathcal{F}_{f,n}(x) - \overline{\mathcal{F}_{f,n-1}(x)})^2, \tag{2.1}$$

and standard deviation σ_{N}:

$$\sigma_{\mathrm{N}} = \sqrt{\frac{1}{W}\sum_{f=0}^{W-1}(\mathcal{F}_{f,n}(x) - \overline{\mathcal{F}_{f,n-1}(x)})^2}. \tag{2.2}$$

These two variance and std. deviation expressions can be implemented without additional FIFO buffer.

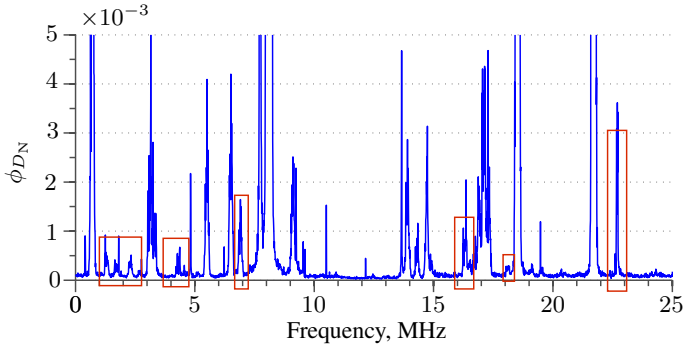


Fig. 2.7. 25 MHz bandwidth radio spectrum record after $\phi_{D_{\mathrm{N}}}$ extraction

When neighboring channel mean value is used for variance calculation – $\phi_{D_{\mathrm{N}}}$, noise level of processed radio spectrum sample slightly increases in comparison with original estimation – ϕ_D (Fig. 2.7). With this modification SNR for most signals is lower too. In areas 1, 2 and 5 low level signal SNR ratio is by 1.2–2 dB lower, than it was with ϕ_D. Biggest difference between results is in 3 and 4 cases, where SNR is dropped to 13 dB and 12 dB. In 6 case SNR is 15 dB and it is same as in original calculation.

Another modification (2.3) for variance can be made by removing second multiplication element (Fig. 2.5):

$$D_{\mathrm{M}} = \frac{1}{W}\sum_{f=0}^{W-1}|\mathcal{F}_{f,n}(x) - \overline{\mathcal{F}_{f,n-1}(x)}|, \tag{2.3}$$

in this modification – $\phi_{D_{\mathrm{M}}}$ comparison between mean value and spectrum component must be made before subtraction in order to receive positive distance. Lower value mus be always subtracted from bigger.

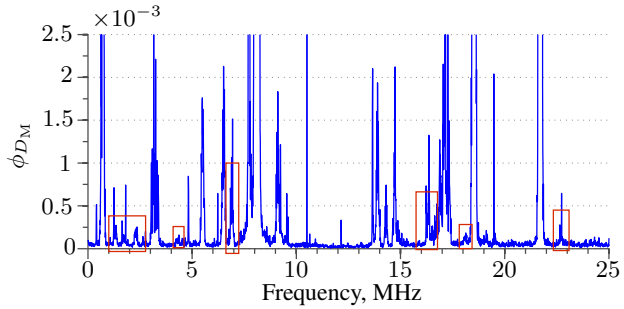


Fig. 2.8. 25 MHz bandwidth radio spectrum record procesed by variance without square – ϕ_{D_M} extractor

In general ϕ_{D_M} performed worse than ϕ_A calculation see Fig. 2.8. For some low level signals SNR was 5–8 dB (1, 2 and 3 marked areas), and these results are similar to ϕ_A. Yet for 6 case SNR is achieved by 10 dB lower. In 3 area signal SNR was 10.6 dB and it was by 1.4 dB lower than in ϕ_A case.

In order to implement standard deviation extractor – ϕ_σ, Cordic square root component must be added into ϕ_D chain (Fig. 2.9). Cordic is coordinate rotation digital computer. This component is based on addition, subtraction and bit shift operations. Therefore, it has long latency, it needs 26 clock cycles to calculate square root. Then ϕ_σ system will use 47 cycles to calculate parameter for 8 samples.

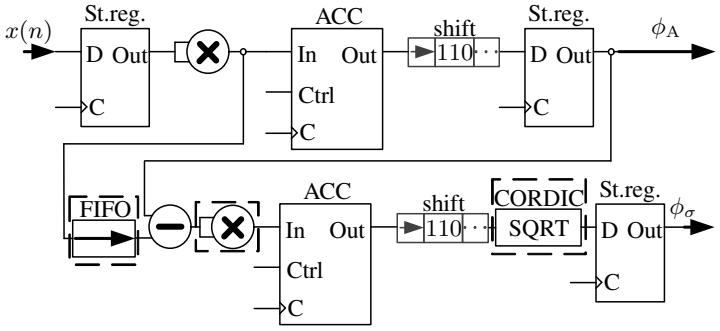


Fig. 2.9. Chanel power std. deviation – ϕ_σ implementation in FPGA

Worst results from all extractors, tested in this section, were achieved with ϕ_σ implementation (Fig. 2.10). In 1 and 2 areas signals SNR vary from 3.2 dB to 6 dB. In 6 case SNR is lower by 9 dB, than result achieved with ϕ_A. With others signals (3, 4, and marked areas) extraction is by $\sim$ 2 dB lower too, in comparing with ϕ_A.

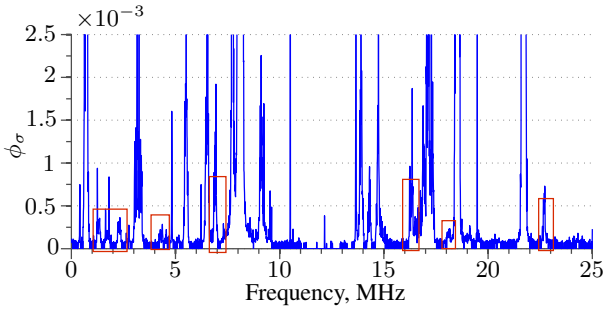


Fig. 2.10. 25 MHz bandwidth radio spectrum record after ϕ_σ extraction

In Table 2.2 is shown how much FPGA resources utilize various feature extractors, when channel consist of 16 spectrum samples. ϕ_{A} uses least FPGA resources and it has lowest delay, just 20 clock cycles. ϕ_D uses 3 times more resources and its latency is about 2 times longer than average. Latency was lowered to 21 clock cycle with $\phi_{D_{\mathrm{N}}}$, which used neighboring channel average value. It uses slightly less FPGA resources than ϕ_D, because FIFO buffer was removed from structure. Another variance optimization ϕ_{DM} was made by removing second multiplication component. This modification helped to save 4 DSP slices. Most FPGA resources were used by ϕ_σ implementation. This significant increase was caused by Cordic square root component. And this component caused additional 26 clock cycles latency. ϕ_σ estimation can be modified as well as variance. It can use neighboring channel average value. Then latency of ϕ_σ can be reduced to 47 clock cycles.

Table 2.2. FPGA resource utilization for 16 samples channel features calculation

Extractor	Slice registers	LUTs	DSP slice	Block RAM	Latency cycles
ϕ_{A}	99	78	1	0	20
ϕ_D	261	293	5	1	37
$\phi_{D_{\mathrm{N}}}$	227	240	5	0	21
$\phi_{D_{\mathrm{M}}}$	256	290	1	1	36
ϕ_σ	504	553	5	1	63
$\phi_{\sigma_{\mathrm{N}}}$	470	500	5	0	47

From these implementation researches it is clear that ϕ_σ is least efficient and it uses most FPGA resources. Best results was achieved with ϕ_D, however it has longer delay than ϕ_{A}. Therefore, for latency reduction can be used $\phi_{D_{\mathrm{N}}}$, which is slightly less efficient than original. In both variance cases aver-

age value is accessible too. In conclusion ϕ_D is most cost efficient extractor, because it has best performance and an additional parameter (average) is calculated as a side product.

2.2.3. Wavelet Transform Implementation in Field Programmable Gate Array

Wavelet transforms can be used to filter spectrum components in time. By these transforms can be highlighted spectrum energy and significant changes in FFT coefficients. 3 transforms Haar ϕ_{Haar}, Daubechies ϕ_{Deb} and Symlet ϕ_{Sym} are implemented (equations (1.10), (1.11), (1.14) and (1.15)). Daubechies and Symlet wavelet transforms are implemented as filter banks (Fig. 1.8). Haar transform have different implementation approach, because it consist just from summation and differentiation.

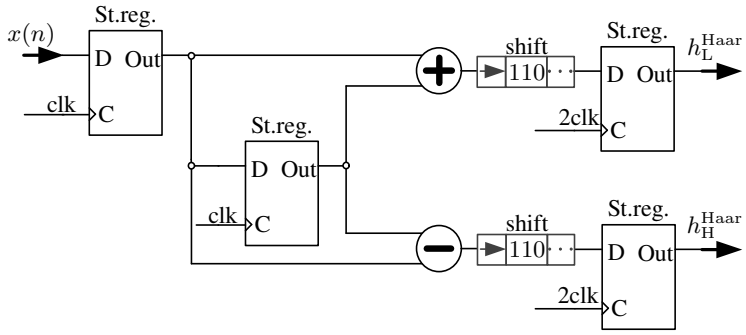


Fig. 2.11. One stage Haar transform imlementation in FPGA

Haar mother and scale wavelets sum and subtract two neighboring samples (Fig. 2.11). To implement these two operations delay must be added in to design, because actions are made between input and its older value. For this case FIFO buffer isn't appropriate. It's too big for one sample delay. Therefore, this operation is implemented by second state register. It delays input sample by one clock cycle. After summation and differentiation bit shift by one bit is made in order to implement division by 2. Decimation is made by synchronizing design output with clock, which period is 2 times longer than used for hole system. In Fig. 2.11 upper part are shown mother wavelet and lower part scale wavelet implementations. By this design 1 level wavelet transform is implemented . To calculate transform output are used 4 clock cycles.

Other transforms are implemented as filter banks. In their designs differ just filter coefficients, however structures are similar. Therefore, Daubechies

Table 2.3. FPGA resource utilization by wavelet transforms

Transform	Slice registers	LUTs	DSP slice	Latency cycles
Haar 2lvl.	291	554	0	8
2lvl wav. 6-4	1417	871	16	33
2lvl wav. 4-6	1423	876	16	33
2lvl wav. 8-4	1424	888	18	40
2lvl wav. 4-8	1428	891	18	40
2lvl wav. 8-6	1468	943	20	48
2lvl wav. 6-8	1469	946	20	48

– ϕ_{Deb} and Symlet – ϕ_{Sym} similar structures will use same FPGA resources. 2 level wavelet transforms are implemented with filter lengths 6–4, 4–6, 8–4, 4–8, 8–6 and 6–8. First digit marks how much coefficients are used in structure 1 level, second - in 2 level. In comparing with ϕ_{Haar} all structures use DSP slices and they latency is about ~ 5 times longer. Depending from filters lengths structures use from 16 to 20 DSP slices. And delay vary from 33 to 48 clock cycles. Logic units doesn't vary much from filters length. Yet they are utilized by 2–3 times more than in ϕ_{Haar} case. Chose of transform filter lengths must be done according to DSP slice and latency budgets.

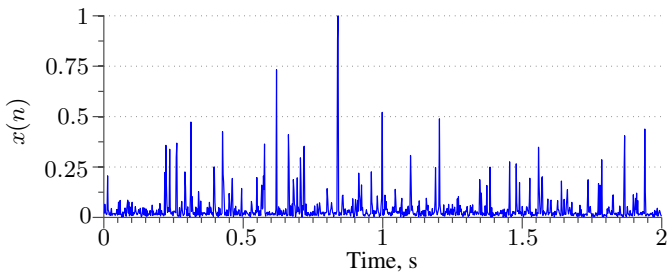


Fig. 2.12. FFT coefficients of radio record in time

Wavelet transforms test signal is taken from previous section RF record. It shows how spectrum components vary during two seconds interval. From this signal by wavelet transforms must be extracted signals spikes energy, highlighted spectrum changes and lowered noise.

After 2 level ϕ_{Haar} transform extracted 4 signals, they are shown in Fig. 2.13:

- $h_{\mathrm{LL}}^{\mathrm{Haar}}$. In test signal until 0.2 s was mostly noise, therefore, it was lowered by 2 stages of low-pass filters. After this interval ϕ_{Haar} summations summed neighboring samples in one spike, which shows energy of short period. Average value of spikes in period 0.2–0.75 s is 0.25.

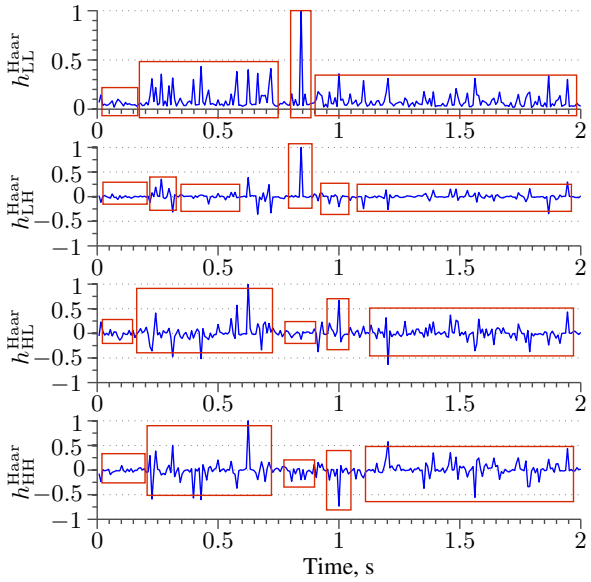


Fig. 2.13. FFT coefficients in time after 2 stage Haar transform

In 0.84 s spike, which has biggest amplitude, is highlighted. In interval 0.9–2 s lower power pulses are combined, where average value is 0.15.

- $h_{\mathrm{LH}}^{\mathrm{Haar}}$. In signal beginning noise level was efficiently lowered. In another 0.2–0.34 s interval it marked more frequent burst. However if intensity of burst drops, so drops $h_{\mathrm{LH}}^{\mathrm{Haar}}$ signal amplitude. It's obvious by comparing $h_{\mathrm{LL}}^{\mathrm{Haar}}$ and $h_{\mathrm{LH}}^{\mathrm{Haar}}$ signals 0.34–0.6 s time intervals. In 0.84 s it marked same signal as highlighted in $h_{\mathrm{LL}}^{\mathrm{Haar}}$.
- $h_{\mathrm{HL}}^{\mathrm{Haar}}$. In this signal changes of spectrum components is more highlighted than in $h_{\mathrm{LL}}^{\mathrm{Haar}}$ and $h_{\mathrm{LH}}^{\mathrm{Haar}}$ cases. Therefore, in intervals 0.2–0.75 s, 0.9–1.05 s and 1.1–2 s spectrum bursts are clearly shown. Bursts are sharpen by intensity of signal change from positive to negative part. In $h_{\mathrm{HL}}^{\mathrm{Haar}}$ high amplitude burst isn't marked, which was clear in $h_{\mathrm{LL}}^{\mathrm{Haar}}$ in $h_{\mathrm{LH}}^{\mathrm{Haar}}$ cases.
- $h_{\mathrm{HH}}^{\mathrm{Haar}}$. This signal is similar to $h_{\mathrm{HL}}^{\mathrm{Haar}}$, yet changes in spectrum are more clear. Amplitudes of spikes are on average by 0.1 greater than in $h_{\mathrm{HL}}^{\mathrm{Haar}}$ case. It has the same problem to highlight high amplitude signal in 0.84 s moment.

Haar transform $h_{\mathrm{LL}}^{\mathrm{Haar}}$ signal combines energy from neighboring spectrum components. It can efficiently mark short period power. $h_{\mathrm{LH}}^{\mathrm{Haar}}$ signal can be used to mark noise periods. $h_{\mathrm{HL}}^{\mathrm{Haar}}$ and $h_{\mathrm{HH}}^{\mathrm{Haar}}$ signals can be used to highlight changes in radio spectrum.

After 2 level 6-8 ϕ_{Deb} 4 signals are shown in Fig. 2.14:

- $h_{\text{LL}}^{\text{Deb}}$. ϕ_{Deb} transform $h_{\text{LL}}^{\text{Deb}}$ signal quite similar to ϕ_{Haar}. Just in ϕ_{Deb} case most spikes amplitude is lower by 8–10% .
- $h_{\text{LH}}^{\text{Deb}}$. In this case changes in radio spectrum is better highlighted than in ϕ_{Haar}. Specially it is clear in 0.2–0.75 s and 0.91–2 s intervals. Yet in this case $h_{\text{LH}}^{\text{Deb}}$ signal isn't useful for noise extraction.
- $h_{\text{HL}}^{\text{Deb}}$. Otherwise than in ϕ_{Haar} $h_{\text{HL}}^{\text{Deb}}$ signal is quite similar to $h_{\text{LH}}^{\text{Deb}}$, however not to $h_{\text{HH}}^{\text{Deb}}$. They both marks changes in spectrum. In this signal is highlighted high amplitude signal, which was not marked in ϕ_{Haar}.
- $h_{\text{HH}}^{\text{Deb}}$. In this signal noise is more suppressed than $h_{\text{LL}}^{\text{Deb}}$, $h_{\text{LH}}^{\text{Deb}}$ and $h_{\text{HH}}^{\text{Deb}}$. It's clear from intervals 0–0.2 s, 0.35–0.6 s and 0.9–2 s. In $h_{\text{HH}}^{\text{Deb}}$ high amplitude signal, which is in 0.84 s, is marked too.

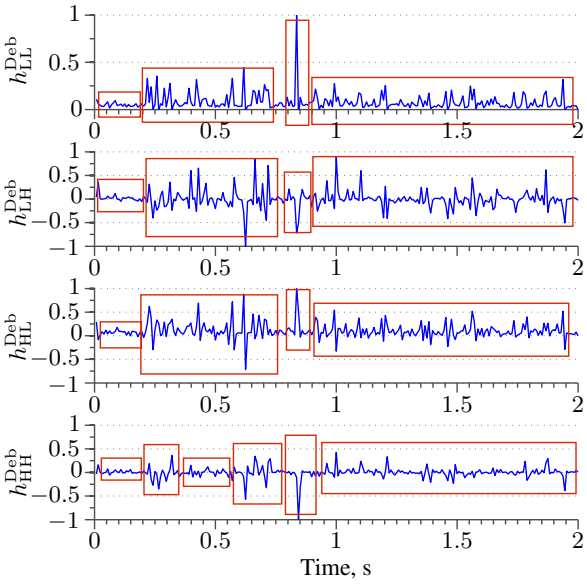


Fig. 2.14. FFT coefficients in time after 2 stage Daubechies transform

As in ϕ_{Haar} case to mark short period power $h_{\text{LL}}^{\text{Deb}}$ signal is suitable. In ϕ_{Deb} to track spectrum changes better to use $h_{\text{LH}}^{\text{Deb}}$ and $h_{\text{HL}}^{\text{Deb}}$ signals. And for noise extraction is better $h_{\text{HH}}^{\text{Deb}}$.

ϕ_{Sym} 2 level 6–8 transform 4 signals are shown in Fig. 2.15:

- $h_{\text{LL}}^{\text{Sym}}$. As in previous transforms (ϕ_{Haar} and ϕ_{Deb}) Low-pass filters chain extracts spectrum energy. Amplitudes of spikes is quite similar to ϕ_{Deb} $h_{\text{LL}}^{\text{Sym}}$ signal.

- $h_{\mathrm{LH}}^{\mathrm{Sym}}$. In ϕ_{Sym} transform as in ϕ_{Deb} $h_{\mathrm{LH}}^{\mathrm{Sym}}$ signal is suitable to track spectrum changes.
- $h_{\mathrm{HL}}^{\mathrm{Sym}}$. In this case all components is heavily suppressed, except component in 0.84 s. Therefore, it's more suited for noise marking.
- $h_{\mathrm{HH}}^{\mathrm{Sym}}$. Same as $h_{\mathrm{LL}}^{\mathrm{Sym}}$ and $h_{\mathrm{LH}}^{\mathrm{Sym}}$ signals $h_{\mathrm{HH}}^{\mathrm{Sym}}$ is very similar to ϕ_{Deb}, therefore, it saves the same purpose as in previous transformation.

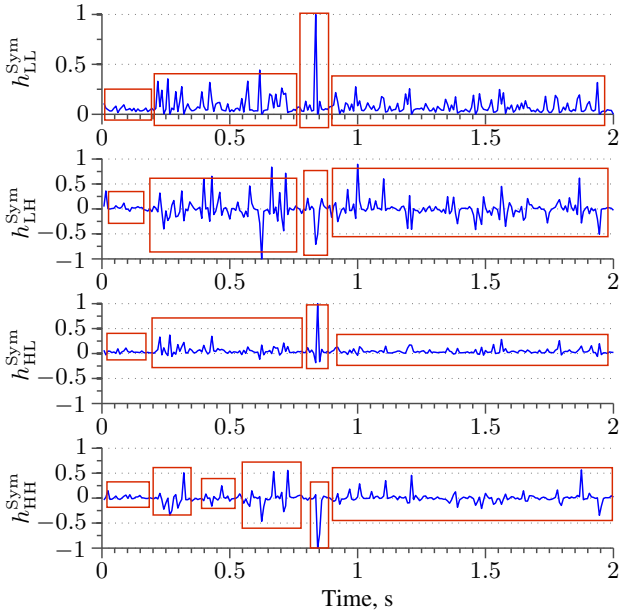


Fig. 2.15. FFT coefficients in time after 2 stage Symlet transform

With ϕ_{Sym} transformation $h_{\mathrm{LL}}^{\mathrm{Sym}}$, $h_{\mathrm{LH}}^{\mathrm{Sym}}$ and $h_{\mathrm{HH}}^{\mathrm{Sym}}$ signals were similar to ϕ_{Deb}. Just $h_{\mathrm{HL}}^{\mathrm{Sym}}$ signal was significantly different. Therefore, in transform all signals saved they purposes as was in ϕ_{Deb} except $h_{\mathrm{HL}}^{\mathrm{Sym}}$, which is more suited for noise determination.

From wavelet transform research it is clear that in all transformations h_{HL} signal can be removed from designs, because it always doubles some feature extraction. With this removal about 15% FPGA resources can be saved, just latency would be the same. By using ϕ_{Haar} transform DSP slices and up to 3 times logic cells can be saved. Though with ϕ_{Haar} some emissions can be lost, as it was in $h_{\mathrm{HH}}^{\mathrm{Haar}}$ case with high amplitude signal. With ϕ_{Deb} and ϕ_{Sym} transforms similar results can be achieved. Especially when h_{HL} signal is removed.

2.3. Self Organizing Maps Self-training and Structure Optimization

2.3.1. Self-training Process Endpoint Detection

The convergence of SOM depends on three main aspects: selection of initial weights, self-training rate and neighborhood size. Therefore, limiting the SOM self-training process by maximum number of iterations is not reasonable, because the SOM self-training performance may be not very sensitive to the number of self-training steps (Brugger *et al.* 2008). The selection of the SOM self-training endpoint can be made by monitoring the estimated mean value of the cost function (Kohonen 1991; Lampinen, Oja 1992). As an alternative, the tracking of the $\eta(n) \simeq \eta(n-1)$ can be used as indication to stop the self-training, if the changes of the self-training rate becomes insignificant (Vegas-Azcarate *et al.* 2005). The third alternative method for endpoint selection is based on the SOM cluster quality measure (Herbert, Yao 2007).

The preliminary experimental tests of the currently available SOM self-training endpoint selection methods showed that the automatic selection of the endpoint was made too early (the sensitivity of the SOM based spectrum sensor could ne improved by adding additional self-training iterations) or too late (the sensitivity of the SOM based spectrum sensor did not change).

The self-training performance of the SOM depends on many factors, including the size of the lattice. SOM with three different lattice size were tested:

- 2×2 size lattice (see Fig. 2.16);
- 5×5 size lattice (see Fig. 2.17);
- 8×8 size lattice (see Fig. 2.19).

Two SOMs with higher number of neurons in the lattice were organized in square topology. The smallest SOM had only four neurons, therefore, it was too small to apply various self-training strategies. A signal with uniformly distributed random values was used as an input to all tested SOMs. The initial weights for each SOM were also selected randomly, ensuring there are no equal weights.

The initial self-training rate $\eta_0 = 1$ was assigned for 2×2 size SOM (see (2.5) equation). In addition, the time constant $\tau = 2500$ was assigned in (2.5) equation. Because of the small SOM lattice, there were no neighbors used during SOM self-training.

In Fig. 2.16a, the position of SOM weights is shown right after the first self-training cycle. Therefore only the winner neuron weights were updated. Other neurons were at the same positions as in the initial state. The SOM weight positions after 1000, 2500 and 5000 self-training iterations are shown in

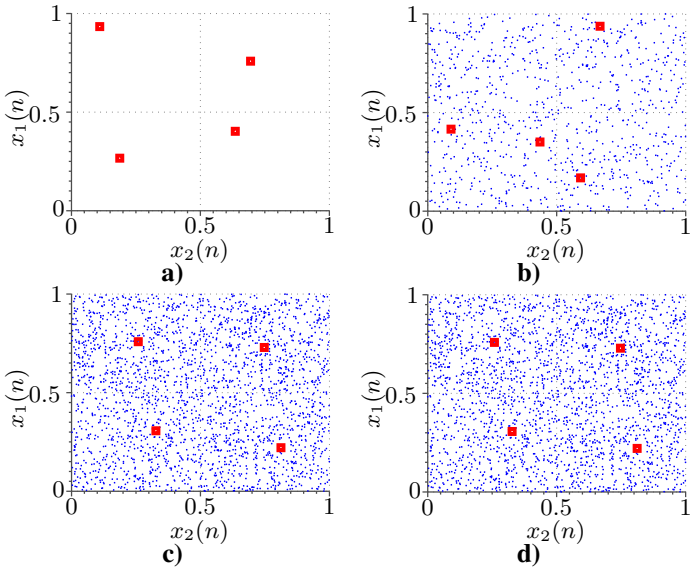


Fig. 2.16. 2×2 size SOM after different number of self-training iterations: a) 1; b) 1000; c) 2500; d) 5000

Figs. 2.16b–d. However, the result in Fig. 2.16c shows, that there were enough 2500 iterations to reach the convergence of the SOM. In addition, 2500 iterations were selected manually, yet the number of self-training iterations could be selected between 2500 and 5000 in order to ensure the stable convergence for different initial weights.

The intermediate self-training results for 5×5 size SOM are shown in Fig. 2.17. The square topology was used for this map. Only one set of neighbors was selected during self-training. The self-training ratio was changed according to (2.5) equation by setting different initial self-training ratio for the winner neuron ($\eta_0 = 1$) and for the neighboring neurons ($\eta_1 = 0.3$).

Nine neurons in total were updated after the first iteration of the SOM self-training process. As it is seen in Fig. 2.17a, the SOM weights were distributed randomly. The more regular distribution of the SOM weights was noticed only after 1000–2500 self-training iterations (see Fig. 2.17b and Fig. 2.17c). However, the number of iterations was not sufficient for the convergence of the SOM. The alignment of SOM weights have not been fully completed even after 5000 self-training iterations (see Fig. 2.17d). The additional tests showed, that additional 500 iterations were needed for complete SOM convergence (see Fig. 2.18). The alignment of SOM weights, shown in Fig. 2.17d and in Fig. 2.18 is similar. However, in order to ensure the stable convergence of SOM, the higher number of iterations should be used.

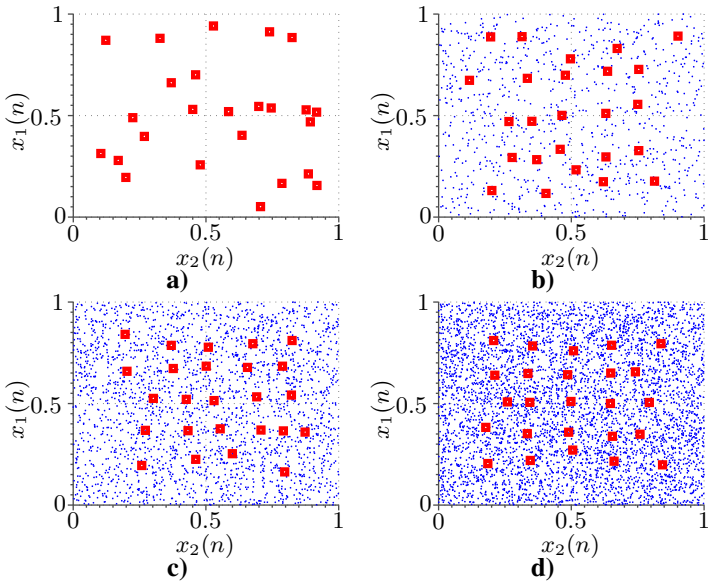


Fig. 2.17. 5×5 size SOM after different number of self-training iterations: a) 1; b) 1000; c) 2500; d) 5000

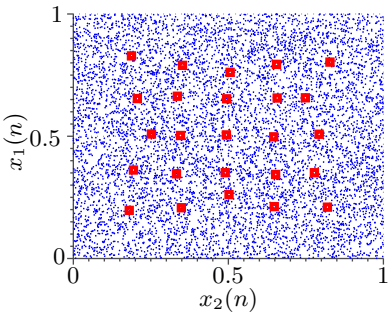


Fig. 2.18. 5×5 SOM after 5500 self-training iterations

The intermediate self-training results for 5×5 size SOM are shown in Fig. 2.17. The square topology was used for this map. Two sets of neighboring neurons were selected during self-training. The self-training ratio was changed according to (2.5) equation by setting different initial self-training ratio for the winner neuron ($\eta_0 = 1$), for the nearest neighboring neurons ($\eta_1 = 0.5$) and for the rest of the neighboring neurons ($\eta_2 = 0.2$).

For the 8×8 size SOM, 25 neurons were updated in each self-training cycle. After first iteration, despite some neurons have been updated, weights

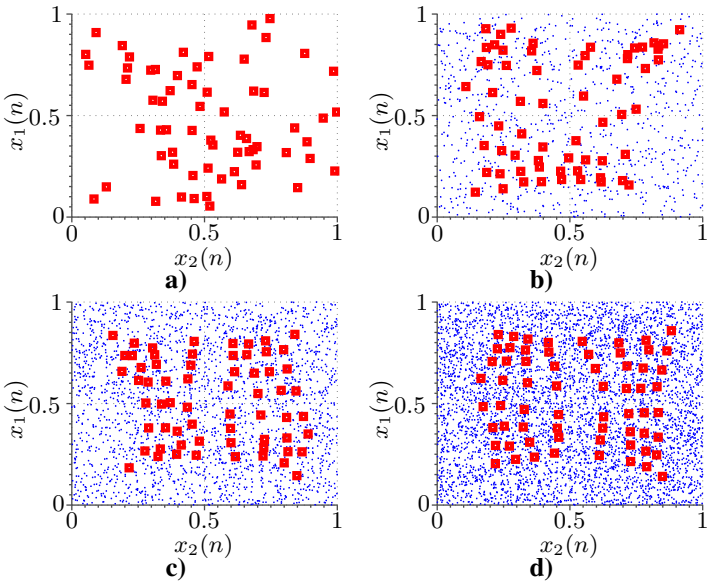


Fig. 2.19. 8×8 size SOM after different number of self-training iterations: a) 1; b) 1000; c) 2500; d) 5000

were still distributed randomly (see Fig. 2.19a. Even after 1000 iterations, the structure of SOM weights was mostly irregular (see Fig. 2.19b. After 2500–5000 iterations SOM weights did not have regular structure (see Figs. 2.19c, 2.19d). Therefore, the self-training of SOM had to be continued. The complete convergence of the SOM was reached after 8000 iterations (Fig. 2.20a). As a result, all SOM weights were distributed among input plane in four groups.

Looking at the self-training results of the tested SOM with different number of neurons it was clear, that the bigger SOM needs more self-training iterations. In addition, the quality of self-training highly depended on the distribution of the SOM inputs, that were used for each self-training iteration. Therefore, the order of the inputs may increase or decrease the minimum number of iterations, that are needed to reach the self-training convergence of the SOM. In order to minimize the number of SOM self-training iterations, an algorithm for endpoint selection is needed.

There are two approaches available for SOM self-training: supervised SOM (Pateritsas *et al.* 2004) and un-supervised SOM (Fritzke 1994). The supervised SOM self-training requires additional external data during SOM self-training procedure. There are various algorithms available for supervised self-training: methods based on measurement of generalization (Koikkalainen, Oja 1990; Lampinen, Kostiainen 1999), self-training associations by self

organization (Carpenter, Grossberg 1992; Fessant *et al.* 2001), label propagation and others (Herrmann, Ultsch 2007).

During unsupervised SOM self-training, the network status or the changes of node weights are monitored. In addition, the representation of input data by SOM can be analyzed. The changes of SOM node weights reflects the changes in map structure. If the weights changes are negligible during self-training, the SOM self-training process is terminated. However, negligible changes of the SOM node weighs during self-training does not guarantee that the obtained topology properly represents input data.

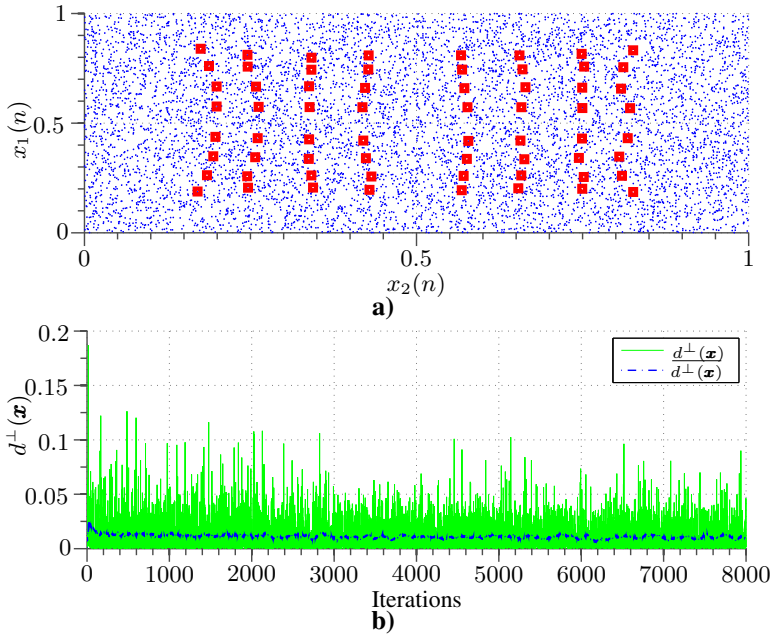


Fig. 2.20. Illustration of SOM self-training results: a) 8×8 SOM after 8000 self-training iterations; b) winner neuron distance from input

The endpoint selection during self-training of SOM requires continuous monitoring of the self-training process. It is possible to classify currently proposed approaches into two types: continuous monitoring of the current SOM during self-training process or analysis of the input data representation by the SOM.

Using the SOM monitoring approach (results of this approach application are shown in Figs. 2.16–2.20), the SOM self-training process is suspended, when the changes of neuron weights become insignificant (Vegas-Azcarate *et al.* 2005). The weight update of the winner neuron Δw highly depends on the

selected adaptive self-training rate η according to the following expression:

$$\boldsymbol{\omega}_{ij}(n+1) = \boldsymbol{\omega}_{ij}(n) + \eta(n)(\boldsymbol{x}(n) - \boldsymbol{\omega}_{ij}(n)). \tag{2.4}$$

The self-training rate η changes adaptively and is being estimated accordingly to the following expression:

$$\eta(n) = \eta_0 e^{-\frac{n}{\tau}}, \tag{2.5}$$

here η_0 is the initial value of the self-training rate.

During SOM self-training process, the η decreases exponentially. If the self-training rate changes are insignificant and $\eta(n) \simeq \eta(n-1)$, the further update of the neuron weights is not reasonable. However, low value of the self-training rate η does not mean that the SOM represents input data well.

In order to monitor the input data representation by the SOM, distance changes between the input $\boldsymbol{x}$ and the winner neuron weight vector $\boldsymbol{\omega}_{ij}$ should be estimated (Hulle 2000; Moreira, Fiesler 1995; Solodov, Svaiter 2000). During such approach, the endpoint is initiated when the distance between inputs and winner neuron weights reach their minimum $d^{\perp}(\boldsymbol{x})$ (Fig. 2.20):

$$d^{\perp}(\boldsymbol{x}) = \min_{ij} \|\boldsymbol{x} - \boldsymbol{\omega}_{ij}\|. \tag{2.6}$$

As an alternative to the estimation of the distance between input and neuron weights, the quality of the input data clustering by a set of neurons may be used. However, these alternative approaches require performing additional analysis using the higher order statistical data (Gunes Kayacik *et al.* 2007; Herbert, Yao 2007).

The SOM self-training endpoint selection method, proposed in this dissertation is based on the idea of monitoring the input data representation by the SOM. During self-training the estimated value of $d^{\perp}(\boldsymbol{x})$ is compared to the threshold θ_{end} and used to initiate the endpoint.

Fig. 2.20b represents the momentary and averaged distances between SOM inputs and weights. It is possible to notice in Fig. 2.20b, that the θ_{end} has a tendency to decrease. Taking into account, that $\overline{d^{\perp}(\boldsymbol{x})} = 0.0065$ was received after 6200 iterations (Fig. 2.21) and the final value $\overline{d^{\perp}(\boldsymbol{x})} = 0.00955$ was received after 8000 iterations was higher, the self-training endpoint should be somehow selected earlier. Therefore, there is a demand for an additional parameter θ_{end} for threshold selection at the point, when the value of $\overline{d^{\perp}(\boldsymbol{x})}$ becomes acceptable.

In order to select an appropriate threshold θ_{end} for the endpoint, the attention must be drawn to the first cycles of the SOM self-training (see Fig. 2.22a).

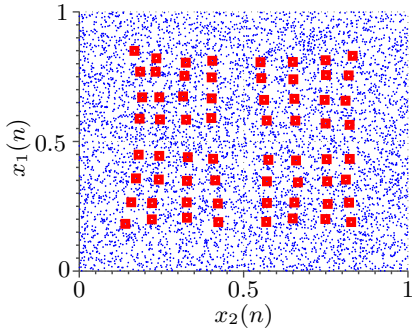


Fig. 2.21. 8×8 SOM after 6200 self-training iterations

From the Fig. 2.22 it is clear that the signal, obtained by continuously estimating $d^{\perp}(\boldsymbol{x})$, has a spiking nature. However, during SOM self-training the magnitude of the spikes decreased. The monotonic decrease of the distance between input and SOM neuron weights can be seen also in the characteristics of the averaged $d^{\perp}(\boldsymbol{x})$. In the given example, the maximum of $d^{\perp}(\boldsymbol{x})$ was achieved at the 16^{th} self-training iteration. The current input-to-weights distance comparison with the maximum, achieved in the first cycles of SOM self-training, was used to make the final decision in the threshold θ_{end} selection phase.

To better understand tendencies of $d^{\perp}(\boldsymbol{x})$ changes, spikes of signal must be filtered with high pass filter. For this effect can be used differentiator Fig. 2.22b, like it has been used in Haar transform. Dynamics of $d^{\perp}(\boldsymbol{x})$ signal change, which represent changes in SOM weights, are more clear after filtration. From equations (2.6) and (2.4) it's clear that greater distance between neuron and input provokes significant changes in its weights. To highlight more significant changes of $d^{\perp}(\boldsymbol{x})$ and move negative signal part to positive, it must be raised squares Fig. 2.22c. Now in this signal it is clear, that SOM coarse changes have been made in first 80 self-training cycles. From 100 iteration it has started adjustments. Coarse alteration phase must be assessed by calculating its mean square root value. Square root operation is needed, because new parameter must be the same dimension like $d^{\perp}(\boldsymbol{x})$ (to undone square operation). Now SOM has new parameter – initial change Δ_{init} equation (2.7), where N_{wind} is number of samples, how much would be taken from the start of self-training:

$$\Delta_{\text{init}} = \sqrt{\frac{1}{N_{\text{wind}}} \sum_{n=0}^{N_{\text{wind}}-1} \left(d^{\perp}_{n+1}(\boldsymbol{x}) - d^{\perp}_{n}(\boldsymbol{x}) \right)^2}. \tag{2.7}$$

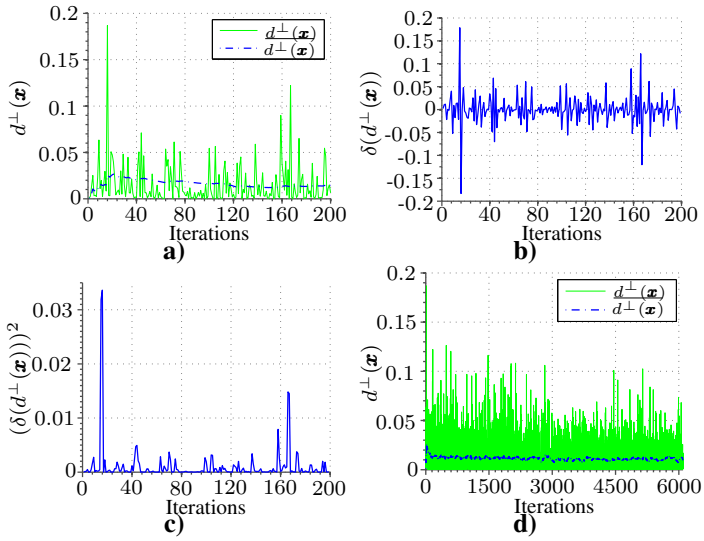


Fig. 2.22. Illustration of input-to-weights distance estimate $d^{\perp}(\boldsymbol{x})$ changes during SOM self-training: a) $d^{\perp}(\boldsymbol{x})$ continuous estimate and it's average $\overline{d^{\perp}(\boldsymbol{x})}$ over 200 iterations; b) $d^{\perp}(\boldsymbol{x})$ continuous estimate after differentiation; c) squared differential signal; d) $d^{\perp}(\boldsymbol{x})$ continuous estimate and it's average $\overline{d^{\perp}(\boldsymbol{x})}$ over 6200 iterations

To estimate θ_{end} value, in Δ_{init} expression must be embedded coefficient κ equation (2.8). Because if θ_{end} is assigned directly to Δ_{init}, self-training cycle will be stopped just after coarse weight alignment. κ marks depth of self-training and it can vary from 1 to 0. It depends on how well SOM must progress from initial self-training moment:

$$\theta_{\text{end}} = \kappa\Delta_{\text{init}} = \kappa\sqrt{\frac{1}{N_{\text{wind}}}\sum_{n=0}^{N_{\text{wind}}-1}\left(d_{n+1}^{\perp}(\boldsymbol{x}) - d_{n}^{\perp}(\boldsymbol{x})\right)^2}. \tag{2.8}$$

Because $d^{\perp}(\boldsymbol{x})$ signal is not stable, θ_{end} must be compared not directly with signal although with its averaged form. Averaged signal is shown in Fig. 2.22a and 2.22b. Then if $\theta_{\text{end}} < \overline{d^{\perp}(\boldsymbol{x})}$ – self-training process continues, otherwise it's suspended.

In θ_{end} expression are two components N_{wind} and κ, which can be chosen by SOM builder. Therefore, it must be tested how these parameters influence the self-training process.

To investigate Δ_{init} dependency on various N_{wind}, 4 nets 2×2, 4×4, 6×6 and 8×8 were self-trained Fig. 2.23. In Figure are shown first 500 iterations

of $d^{\perp}(\boldsymbol{x})$ signal after differentiation and square operations. These 2 operations were done, because, as mentioned before, there is the need to extract coarse changes in the SOM. For each network the Δ_{init} value was calculated with different N_{wind} Table 2.4.

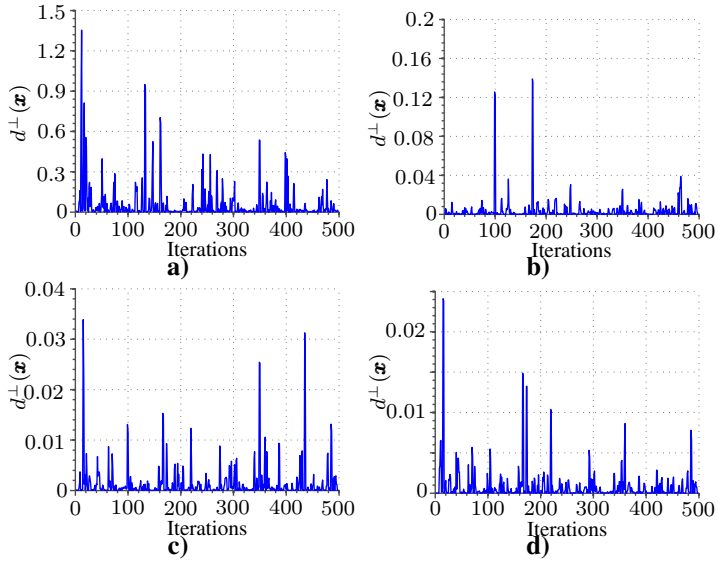


Fig. 2.23. Squared $d^{\perp}(\boldsymbol{x})$ signal after differentiation for different SOM size: a) 2×2; b) 4×4; c) 6×6; d) 8×8

Table 2.4. Δ_{init} value for various windows

size	25	50	75	100	150	200	250	300
2×2	**0.469**	0.359	0.324	0.296	0.297	0.277	0.263	0.258
4×4	0.040	0.039	0.042	0.054	0.06	**0.069**	0.067	0.064
6×6	**0.057**	0.048	0.043	0.04	0.036	0.038	0.036	0.036
8×8	**0.054**	0.045	0.04	0.036	0.032	0.035	0.033	0.032

- 2×2 structure. With $N_{\text{wind}} = 25$ Δ_{init} has reached maximum. Between 100–200 iterations are group of spikes, which must be included into the Δ_{init} estimate, because they show significant changes in the SOM Fig. 2.23a. Therefore, for $N_{\text{wind}} = 200$ $\Delta_{\text{init}} = 0.297$ and it is 36% less than in $N_{\text{wind}} = 25$.
- 4×4 structure. Main 2 spikes are at 99 and 178 self-training cycles Fig. 2.23b, therefore, to assess them must be used $N_{\text{wind}} = 200$. With this window Δ_{init} has reached its maximum value.

- 6×6 structure. In this case significant changes of SOM occurred in the 15, 350 and 435 iterations Fig. 2.23c, however to evaluate them all must be used $N_{\text{wind}} = 450$. Δ_{init} for this window is 0.038. To use this long window is insignificant, specially when the same value estimated with $N_{\text{wind}} = 200$. Δ_{init} maximum was reached with $N_{\text{wind}} = 25$.
- 8×8 structure. Despite that maximum Δ_{init} is achieved with $N_{\text{wind}} = 25$, near 200 iteration are 4 significant spikes, which must be assessed Fig. 2.23d. With window $N_{\text{wind}} = 200$ Δ_{init} is 0.035.

In 3 cases 2×2, 4×4 and 8×8 maximum value of Δ_{init} was achieved with $N_{\text{wind}} = 25$, yet in all structures were more SOM significant changes up to 200 self-training cycle Fig. 2.23. Therefore, by choosing to narrow N_{wind}, important moments of SOM self-training can be not evaluated. To correctly assess initial change N_{wind} must be used from 100 to 200 Table 2.4.

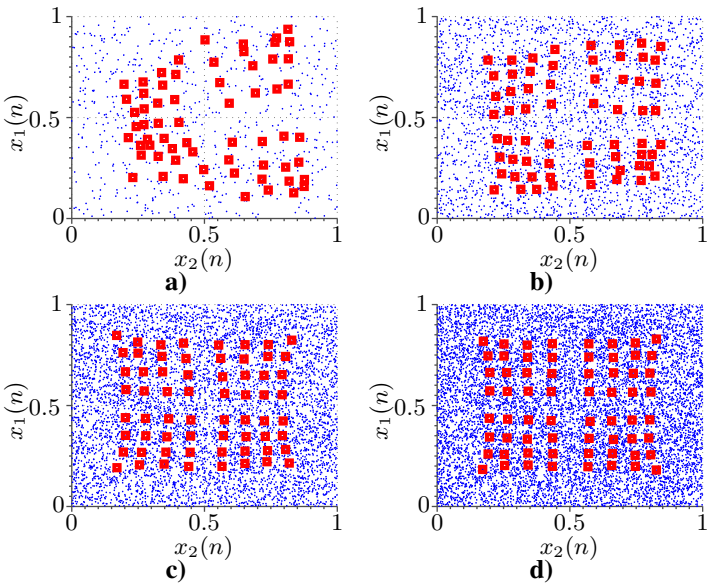


Fig. 2.24. SOM after self-training: a) 8×8 SOM after 811 iterations with $\kappa = 0.3$; b) 8×8 SOM after 3120 iterations with $\kappa = 0.25$; c) 8×8 SOM after 6146 iterations with $\kappa = 0.2$; d) 8×8 SOM after 10 000 iterations with $\kappa = 0.15$

SOM self-training process suspension of 8×8 net was tested with 4 various κ, and self-training had stopped in 4 different development episodes:

- $\kappa = 0.3$. With this parameter $\theta_{\text{end}} = 0.01065$ was estimated. It was reached in 811 self-training cycle. It is clear, that SOM hasn't reached

regular structure, however in upper and lower right quarters weights has started to adjust in right order.

- $\kappa = 0.25$. θ_{end} was reached in 3120 iteration and it was 0.008875. Network is more regular, than in $\kappa = 0.3$ case, yet all SOM weights needs more adjustment.
- $\kappa = 0.2$. Self-training process has stopped at 6146 self-training cycle. Suspension parameter estimated $\theta_{\text{end}} = 0.0071$. In this case SOM is regular, and neurons weights are distributed in 4 groups, like in Fig. 2.21.
- $\kappa = 0.15$. With this parameter θ_{end} has been never reached. Self-training process was suspended, because max quantity of self-training cycles was exceeded. Network hasn't changed significantly in comparing with 6146 self-training moment.

SOM regular was achieved with $\kappa = 0.20$, however with different input signal (not random) higher value can be chosen. Therefore, this parameter can be chosen from interval $0.2 \leq \kappa \leq 0.25$.

This SOM self-training suspension approach is unsupervised manner, therefore, it must be compared with similar algorithms Fig. 2.25 (comparison was done with 8×8 SOM). For this purpose 3 additional methods were chosen:

- Mean value of cost function (Kiang 2002) Fig. 2.25 **mark 2**. Suspension point is determined in beginning of Self-training by mean value of cost function. This algorithm complexity is similar like Δ_{init}, because additional estimations are done in the same moment.
- $\eta(n) \simeq \eta(n-1)$ insignificant change of self-training ratio (Vegas-Azcarate *et al.* 2005) Fig. 2.25 **mark 3**. SOM weights dynamic is monitored in this case. If SOM weights practically doesn't change, then SOM self-training is suspended. This method is least complex, because no additional calculation is needed.
- Measuring SOM quality (Herbert, Yao 2007) Fig. 2.25 **mark 4**. This method is based on measurement how well SOM classify input and how each neuron represents its data like cluster. This rating is done in each self-training cycle, therefore, it's most calculation intense method from this test. The suspension decision is made with some delay, because quality of SOM must be verified for various neurons with various input vectors.

8×8 SOM self-training suspension was done by Δ_{init} estimate in iteration 6265 Fig. 2.25 **mark 1**. From Fig. 2.21 it's clear, that SOM in this point is well self-trained. By SOM quality measurement suspension was done in 6527 iteration **mark 4**. It took 262 self-training cycles more than Δ_{init} case. As mentioned before this delay is caused by SOM quality verification. Net weights change was insignificant in 7151 iteration **mark 3**, and this decision, based on

low $\Delta\eta$ was made at latest in this test. Suspension point established by mean value of cost function was on 4606 self-training moment **mark 2**. At this point in SOM are some irregularities Fig. 2.19, therefore, self-training process by this method was stopped too early.

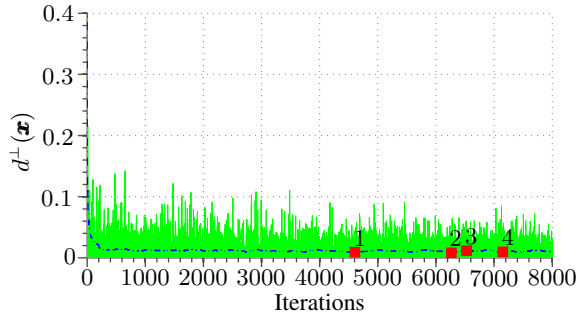


Fig. 2.25. Illustration of $d^{\perp}(\boldsymbol{x})$ changes at every self-training iteration with marked self-training endpoints, obtained during experimental investigation

Like shown in Fig. 2.25, 6 smaller than 8×8 SOM structures were self-trained Table 2.5. Form these results it's clear that all self-training suspension methods save some self-training iterations. Method based on SOM cluster quality has $\backsim$ 250 self-training cycles delay in comparing with Δ_{init}. Suspension made by mean value of cost function was always earliest, yet from SOM cluster quality, Δ_{init} results and Fig. 2.16, 2.17, 2.19 is clear, that SOM in these moments hasn't been fully adjusted.

Table 2.5. Self-training duration for various SOM sizes

Self-training	2×2	3×3	4×4	5×5	6×6	7×7
Mean value	1301	2646	3271	4237	4475	4701
Cluster quality	1921	3119	3663	4871	5517	6415
Δ_{init}	1627	2868	3405	4622	4276	6194

Δ_{init} based self-training suspension method has shown similar results like SOM quality measurements, even so its complexity is considerably lower. Therefore, this estimate can be effectively used for determination of SOM convergence. This method is unsupervised, therefore, it fits for applications like unknown RF environments spectrum sensing. However some consideration of using this method must be made. In SOM design beginning must be chosen 2 parameters equation (2.8):

- N_{wind}, which must be from 100 to 200;
- κ, which must be from 0.2 to 0.25.

2.3.2. Self-training Assistant

Convergence speed and quality of SOM depends on initial conditions and self-training samples (Kiang 2002). It is hard to change inputs samples, because SOM must be adjusted to it. Therefore, input can be just rearranged (Kohonen 1990). More optimization options in self-training process beginning can be achieved with self-training ratio η, because self-training speed depends on it (2.5). $\eta(n)$ depends on initial parameters η_0, τ and self-training iteration there SOM is. $\eta(n)$ decreases exponentially, for slope is responsible τ and for start point η_0. To show how $\eta(n)$ changes during self-training process in Fig. 2.26 it was compared with $d^{\perp}(\boldsymbol{x})$. For this case $\tau = 3500$ and initial $\eta_0 = 1$ was chosen. Up to 6000 iteration $\eta(n)$ induced significant changes in the SOM. $d^{\perp}(\boldsymbol{x})$) value also significantly dropped from initial cycles. After 6000 to 10^4 self-training iteration $\Delta\overline{d^{\perp}(\boldsymbol{x})}$ was only 0.03 and further self-training was pointless as it was explained in Section 2.3.1. From Fig. 2.26 it is clear that $\eta(n)$ during half self-training process doesn't have influence to SOM adjustment. Therefore, $\eta(n)$ value must not only be adjusted before self-training beginning, yet it must be more bonded with self-training result.

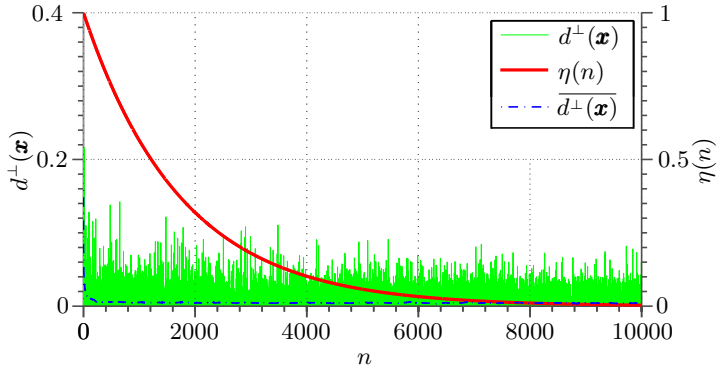


Fig. 2.26. η slope comparison with $d^{\perp}(\boldsymbol{x})$ and $\overline{d^{\perp}(\boldsymbol{x})}$

Numerous self-training ratio adjustment algorithms like adaptation based on the angle between gradient directions in consecutive iterations (Nawi *et al.* 2008), prediction of new values of η (Hwang, Li 2009), calculation of optimal fixed value (Atanassov *et al.* 2008) and etc. are used in neural networks. All of them increase computational load for self-training algorithm and in some cases their performance can be poorer than is shown in Fig. 2.26. Therefore, there is the need of algorithm which increases speed of SOM convergence, although doesn't increases complexity of algorithm.

From Section 2.3.1 it's clear that $\eta(n)$ equation (2.5) leads SOM to convergence. However in some cases, which were mentioned in this section, SOM self-training process can be accelerated or on the contrary slowed down. Decision how to change self-training course must depend on self-training process success ratio. As success ratio parameter can be chosen $d^{\perp}(\boldsymbol{x})$ or its averaged value, as it was done in Section 2.3.1. From these assumptions can be made coarse Algorithm 1., which assists SOM in self-training process.

Algorithm 1. Coarse SOM self-training assistant algorithm

```
1  Procedure: Self-training assistant
2    d⊥(x) ← d⊥(x)
3    if ψ0 < d⊥(x)
4      train. par. ← set/reset
5      …
6      train. par. ← set/reset
7    else if ψ1 < d⊥(x) < ψ2
8      train. par. ← set/reset
9      …
10     train. par. ← set/reset
11   else if …
12     …
13   else
14     train. par. ← train. par.
15     …
16     train. par. ← train. par.
```

This skeleton of Algorithm 1. consists of rules, which make changes for some self-training parameters. Decision depends on $\overline{d^{\perp}(\boldsymbol{x})}$ value. If this value satisfies one of conditions, then changes will be made, otherwise parameters will be unchanged. For this algorithm ψ values and parameters are missing. Parameters will be changing in various cases. ψ must serve like threshold for algorithm to decide how well self-training process is proceeding. As base of ψ value can be used Δ_{init} equation (2.7), because decision about SOM self-training progress can be made from it. For different situations ψ values can be:

- $\psi_0 = \Delta_{\text{init}}$, SOM is far from global minimum, therefore, it need *very hard* reset;
- $\psi_1 = 0.99\Delta_{\text{init}}and\psi_2 = 0.75\Delta_{\text{init}}$, SOM made some progress, however results are unsuitable, it need *hard* reset;
- $\psi_3 = 0.749\Delta_{\text{init}}and\psi_4 = 0.5\Delta_{\text{init}}$, SOM is in the middle of self-training, therefore, it need *soft* changes to accelerate self-training;

- $\psi_5 = 0.49\Delta_{\text{init}}and\psi_6 = 0.3\Delta_{\text{init}}$, SOM is near final convergence, network need just *light* corrections.

Now SOM possible self-training process situations are defined, therefore, there is need for parameters, which would be changed. Efficient way to control self-training process is to control $\eta(n)$. As was mentioned in this section to influence $\eta(n)$ must be changed η_0, τ or self-training iteration number n. Therefore, all these parameters must be controlled by assisting algorithm. Although if after few thousand self-training cycles SOM signal $\overline{d^{\perp}(\boldsymbol{x})}$ rates just ψ_0 and ψ_1–ψ_2 zones, change of $\eta(n)$ will not be enough. Because it's possible, that network weights were tuned in wrong direction and only with $\eta(n)$ they wouldn't be rearranged. Therefore, SOM weights resets in assistant algorithm for these 2 self-training process situations must be provided.

Table 2.6. Self-training ratio slope dependency on τ

τ	1000	1500	2000	2500	3000	3500	4000	4500
iterations	3546	5185	6913	8642	10371	12100	13828	15557

To correctly choose values of η_0 and τ, which are regulated in assistant algorithm, first must be defined how $\eta(n)$ changes when parameters values varies. In Tables 2.6 and 2.7 is shown how much $\eta(l)$ self-training iterations network needs to reach 0.001 mark, with various η_0 and τ values. This point of $\eta(n)$ is chosen, because after this mark SOM weights will not change considerably. And further self-training would be not purposeful.

In Table 2.6 are shown results of slope dependency from τ. In this case $\eta_0 = 1$. From it is clear that higher τ value provokes longer $\eta(n)$ slope. Yet too high and so too low τ values can cause overfitting or SOM will suffer from lack of effective self-training iterations. How is shown in Fig. 2.20 even 8×8 network for convergence take less than 10^4 self-training cycles. Therefore, using higher τ values than 3000 can cause unwanted oscillations. Low value of τ as 1000 can be used only for small network structures like 2×2, because for larger SOM 3546 effective self-training cycles is not enough. From these assumptions for assistant algorithm situations control τ must be chosen in interval from 1000 to 3000.

Table 2.7. η slope dependency on η_0

η_0	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
iterations	10036	9835	9604	9330	8995	8563	7954	6912

To calculate these Table 2.7 results, τ was chosen 3000. In table isn't shown self-training iteration quantity, which was needed to achieve 0.001 with

$\eta_0 = 1$, because it was shown in Table 2.6 and it was 10371. $\eta(n)$ endpoint is proportional to η_0, therefore, these initial ratios can be arranged in 4 equal groups. From these groups η_0 will be chosen for assistant algorithms situations.

As mentioned in this section, in situations there are a need of *hard* and *very hard* resets, only manipulation of $\eta(n)$ is not enough. Therefore, for assistant algorithm must be provided weight full re-initiation in *very hard* and partial in *hard* cases.

Algorithm 2. SOM self-training assistant algorithm

Procedure:Self-training assistant
  $\overline{d^{\perp}(\boldsymbol{x})}) \leftarrow d^{\perp}(\boldsymbol{x}))$
  if $\Delta_{\text{init}} < \overline{d^{\perp}(\boldsymbol{x})})$
    $\tau \leftarrow 3000$
    $\eta_0 \leftarrow 1$
    $n \leftarrow 1$
    $\boldsymbol{\omega} \leftarrow reset$
  else if $0.99\Delta_{\text{init}} < \overline{d^{\perp}(\boldsymbol{x})}) < 0.75\Delta_{\text{init}}$
    $\tau \leftarrow 2500$
    $\eta_0 \leftarrow 0.7$
    $n \leftarrow 1$
    $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \boldsymbol{\omega} * 0.5\text{rand}$
  else if $0.749\Delta_{\text{init}} < \overline{d^{\perp}(\boldsymbol{x})}) < 0.5\Delta_{\text{init}}$
    $\tau \leftarrow 2000$
    $\eta_0 \leftarrow 0.3$
    $n \leftarrow 1$
  else if $0.49\Delta_{\text{init}} < \overline{d^{\perp}(\boldsymbol{x})}) < 0.3\Delta_{\text{init}}$
    $\tau \leftarrow 1500$
    $\eta_0 \leftarrow 0.1$
    $n \leftarrow 1$
  else
    $\tau \leftarrow \tau$
    $\eta_0 \leftarrow \eta_0$

Assistant Algorithm 2. consist of 5 situations:

- Unsatisfactory SOM. In this situation network is dangerously far from representing given input vectors sets, therefore, it must be fully re-initiated. τ and η_0 are fixed in to values, which ensure more than 10^4 cycles of SOM evolution. Network weights reassigned in to new initial values, because older weights don't represent input entirely.

- Regressive SOM. Network improvement is insignificant, therefore, it must be pushed out of this situation. For this purpose SOM weights will be not re-initiated like in previous state, although each neuron weight gets some random value. This operation is done by multiplying half weight value of random number, which vary from 0 to 1 and has uniform distribution. τ and η_0 combination helps SOM to self-train up to 8000 iterations.
- Progressive SOM. Neurons weights started arranging in to right direction and they don't need any readjustment. For this situation stable weight change must be ensured, therefore, these τ and η_0 values are chosen. This set of parameters lets SOM to self-train 5700 iterations. Higher self-training cycle can cause unnecessary weight oscillations.
- Satisfying performance of SOM. Network is going to final self-training stage. For this moment big changes in SOM are not necessary. Network needs max. few thousand self-training cycles for convergence, therefore, τ and η_0 is fixed to these low values.
- Stable SOM. In this state network is near final convergence, therefore, any changes would be more noxious than positive.

In all assistant situations, except the last one, point of $\eta(n)$ was reset $n \leftarrow 1$. This readjustment was done for that self-training ratio would have predictable slope. Otherwise after more than 2000 iterations τ and η_0 change would not have significant influence to self-training process.

To run this assistant self-training algorithm in each iteration will be inappropriate. Because, just after initiation stage, algorithm would change parameters of SOM and network would not have chance to adapt. Therefore, this revision of network self-training quality must be made each few hundred or even each thousand iteration. Then SOM will have enough iterations to adjust its weights in new conditions, and decision about its progress will be more accurate.

To investigate SOM self-training ratio dependency on frequency of algorithm interventions, 8×8 network was self-trained with various periods of assistant checks Fig. 2.27. Interventions period was changed from 200 to 1000. When assistant was used in each 200 self-training iteration, two adjustments were done. First adjustment was done in the first iteration of algorithm, and η_0 with τ was fixed to 0.3 and 2000 accordingly. By $\overline{d^{\perp}(\boldsymbol{x})})$ algorithm decided, that SOM is in progressive state. Second adjustment was done in 1800 iteration and in this case network self-training reached satisfying results. After 3751 iteration from self-training beginning $\overline{d^{\perp}(\boldsymbol{x})})$ reached $0.2\Delta_{\text{init}}$ mark, and self-training process was suspended. At 300 check periodicity, self-training parameters were adjusted 18 times. At each adjustment assistant decided from $\overline{d^{\perp}(\boldsymbol{x})})$ that SOM has satisfying self-training performance and parameters were

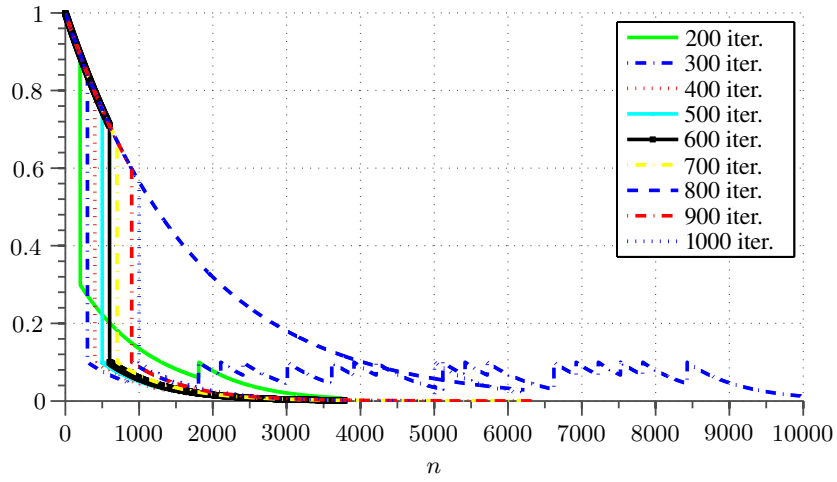


Fig. 2.27. Self-training ratio dependency from assistant interventions

set $\eta_0 = 0.1$, $\tau = 1500$. Yet with these all alterations SOM hasn't reached endpoint even after 10^4 iterations. Network convergence has failed, because $\eta(n)$ readjustment was to frequent. In 400, 500 and 600 cases self-training parameters adjusted just one time. Assisting algorithm in all situations decided that self-training process is satisfying, therefore, parameters were adjusted accordingly. Self-training process for these 3 cases was suspended around 3780 iteration. When assistant checks periodicity were 700 and 900, adjustments were similar like in previous 3 situations, however self-training took 2540 iterations more. In 800 periodicity checks case no adjustments for self-training process were done. Assistant algorithm decided, that SOM self-training is in stable condition, therefore, no corrections were required. Self-raining process was suspended in 6320 iteration. With rarest checks algorithm changed self-training process two times. One was after first revision (1000 iteration), another after 5000 self-training iteration. In both adjustments were decided, that SOM self-training process performance is satisfying and parameters were set $\eta_0 = 0.1$, $\tau = 1500$. In this case self-training suspended in 6320 iteration.

From all results of assistant periodicity checks can be made decision, that too frequent revisions like 200 and 300 can make SOM self-training process unstable. Especially it was clear in 300 period case. Longer periods between revisions than 700 were ineffective, because self-training process was 2540 iterations longer than in 400, 500 and 600 cases. Therefore, assistant algorithm must be used for each 400–600 self-training iteration.

Assistant was tested with 4 SOM structures, to investigate how well algorithm helps self-train network (Fig. 2.28). In Table 2.8 is shown how much

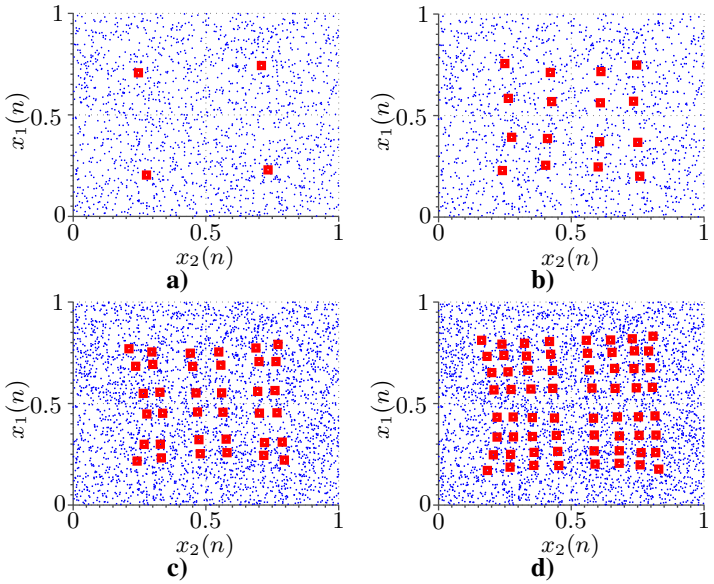


Fig. 2.28. SOM structures self-trained with assistant a) 2×2; b) 4×4; c) 6×6; d) 8×8

iterations were used to self-train SOM with assistant and without it. Periodicity of algorithm checks was set to 500. Self-training process was suspended, when $\overline{d^{\perp}(\boldsymbol{x})})$ reached $0.2\Delta_{\text{init}}$ value. Revisions of self-training process for small structure 2×2 increased self-training process iteration quantity by 134, however SOM has fully adjusted its weights Fig. 2.28a. Assistant algorithm showed better results with larger structures 4×4, 6×6, 8×8. Self-training of 4×4 SOM was faster by 16 iterations than it was for 2×2 network with assistant. Self-training process with revisions for 4×4, 6×6 and 8×8 structures was shorter by $\simeq$ 1500–2000 iterations. From Fig. 2.28bcd it's clear, that with assistant these quantities are enough for SOMs convergence.

Table 2.8. Self-training duration for various SOM sizes

Self-training	2×2	4×4	6×6	8×8
Δ_{init}	1627	3405	4267	6194
With assistant	1761	1745	3282	3783

From all results it is clear that assistant algorithm helps to speed up self-training process. Stable adjustments of SOM self-training can be achieved, when revisions periodicity is from 400 to 600 iterations. As optimization parameter assistant algorithm can use $\overline{d^{\perp}(\boldsymbol{x})})$. In algorithm ranges for this value

can be determinate by Δ_{init}. Assistant is ineffective with small networks like 2×2, therefore, it must be used with larger SOMs.

2.3.3. Self Organizing Map with Inner Weights

SOM can be used efficiently for decision making in unknown radio environments. Yet main disadvantage of these networks is self-training process, specially for larger structures. Because by general rule self-training iteration quantity must be at least 500 times number of neurons in network. For example 6×6 SOM must be self-trained for $1.8 \cdot 10^4$ iterations.

In Sections 2.3.1 and 2.3.2 were researched opportunities to optimize and accelerate self-training process. With endpoint detection self-training phase took 4276 cycles for 6×6 network structure, and it were 119 iterations for one neuron. With assistant self-training process for one neuron took 92 cycles for similar structure (Table 2.8). Even with these improvements for larger structures of SOM, weight adjustment can take too much computational cycles.

Another limiting factors of SOM are $\eta(n)$ and neighborhood functions. These functions decrease exponentially. And to efficiently implement them many multiplication operations or large quantity of memory must be used. Both implementations require more additional resources from hardware. Therefore, in realization of SOM for real-time applications, like radio spectrum sensing, there are problems of latency and implementation complexity.

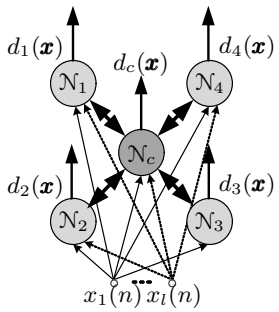


Fig. 2.29. SOM inner weights direction determination

These problems challenge to search and develop different network structure than was mentioned in Sections 1.5.4, 1.5.5. Key task of this research is to find different neurons interactions than neighborhood functions. Instead of this function can be used weights between neurons Fig. 2.29. These weights can serve as link between closest neurons of network. By this connection one neuron output $\mathcal{Y}$ has influence to another. If one neuron has many relationships

with other neurons it can be overwhelmed. Therefore, with large quantity of connections is a great risk. In consideration of this danger, in beginning will be used just few inner weights ω_{inn}.

In Fig. 2.30 network with four inner weights is shown. By these connections center neuron is connected with four neighbors, which are in the outer ring. In this network must be determined influence direction. Inner weights there are pointed to the center or from the center. In the first case four weighted neurons outputs would be used in center neuron output calculation. Otherwise center neuron output can be used in outer ring neurons calculations. In the case when weights are pointed from center, one neuron can have too great influence to other. Therefore, both influence directions must be tested.

Table 2.9. Neurons shots dependency on weights direction

Weight direction	Center	1	2	3	4
To the center	89	218	227	230	236
From the center	393	3	21	570	13

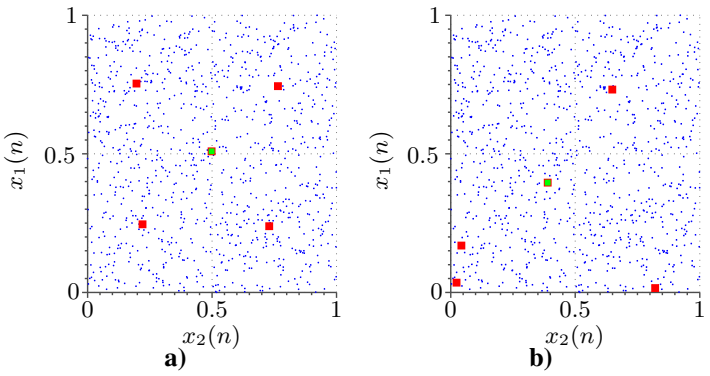


Fig. 2.30. SOM with inner weights pointed a) to center; b) from center

Tests of both structures are done as in Sections 2.3.1 and 2.3.2. During tests was looked how networks can preserve regular structure, when inputs are random sequences. In Fig. 2.30 are shown SOM structures after 1000 self-training iterations and in Table 2.9 is represented quantity of neuron wins. From results it's clear that better performance was achieved with structure, when weights were pointed to center. In this case SOM weights are distributed correctly. Center neuron is in the center and others are in four angles. In competition process more wins have outer ring neurons. They had more than 2 times higher success ratio than center. When weights were pointed to outer

ring, mostly 2 neurons were updated. For another 3 neurons change was insignificant. This situation emerged because for these neurons quantity of competition wins was just 3, 21 and 13. From Table 2.9 and Fig. 2.30 it is obvious, that topology when relations are forced from center is unstable.

From test it is found that SOM inner weights direction must be pointed in to the center. Now more neurons can be added into this structure. Lets take into network 4 more neurons. They marked in Fig. 2.31 by indexes: $\mathcal{N}_{11}$, $\mathcal{N}_{12}$, $\mathcal{N}_{13}$, $\mathcal{N}_{14}$. These neurons must be connected with others components of network. Center neuron can't have more connections, because it would have too significant influence from neighbors. And it can become inactive. Therefore, these 4 additional neurons have relations not with center neuron, although with outer ring neurons. New components of network are positioned between outer neurons (Fig. 2.31). This spacing of nodes in structure dictates, that each neuron must have connections with its 2 neighbors. Therefore, relations will be:

- $\mathcal{N}_{11}$ with $\mathcal{N}_1$ and $\mathcal{N}_2$;
- $\mathcal{N}_{12}$ with $\mathcal{N}_2$ and $\mathcal{N}_3$;
- $\mathcal{N}_{13}$ with $\mathcal{N}_3$ and $\mathcal{N}_4$;
- $\mathcal{N}_{14}$ with $\mathcal{N}_4$ and $\mathcal{N}_1$.

Direction of these connections will be as same as was found in previous research. Weights will be pointed to inward of network. Each additional set of 4 neurons $\mathcal{N}_{x1}$, $\mathcal{N}_{x2}$, $\mathcal{N}_{x3}$, $\mathcal{N}_{x4}$ will be positioned and connected to network likewise.

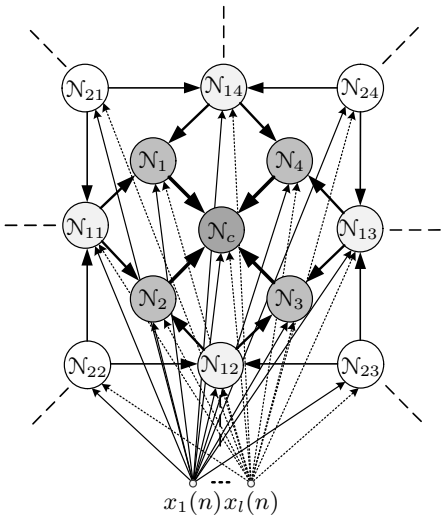


Fig. 2.31. SOM with inner weights structure

Network will consist of these layers:

- center neuron;
- closest to center neurons – they will have strong connection to center node;
- external neurons – each new layer will have relations with previous layer.

Algorithm 3. SOMwith inner weights output calculation

```
1  Procedure: SOM output calculation and winner determination
2    while external layers >0
3      for each neuron in layer
4        Calculate 𝔶
5        if this distance is lower than previous winner
6          declare temporary Winner
7    for each closest to center neuron
8      Calculate 𝔶
9      if this distance is lower than previous winner
10       declare temporary Winner
11   Calculate center neuron 𝔶
12   if this distance is lowest
13     declare Winner ℕ^⊥
14   else
15     Winner ℕ^⊥ temporary Winner
```

In Fig. 2.31 is shown final SOM with inner weights structure. From it can be described network output estimation Algorithm 3.. All neurons inner weights are directed to inward of network. Therefore, outputs of neurons must be calculated from last external layer Algorithm 3.. Because in neuron output calculation ere included neighboring neurons outputs:

$$||\boldsymbol{x} - \boldsymbol{\omega}_{ij}||, \tag{2.9}$$

$$||\boldsymbol{x} - \boldsymbol{\omega}_{ij}|| + ||\mathfrak{y} - \boldsymbol{\omega}_{inn}||. \tag{2.10}$$

And estimation of these outputs must be started from neurons which don't have pointed connections to them equation (2.9). After each output estimation, this value is compared with temporary winner distance. If it's lower, then new temporary winner is declared. After external layers outputs are calculated, closest to center layer can be calculated. In this stage output is calculated just for 4 neurons. Nearest external layer outputs are used in estimation. Center neuron will be calculated last, because closest to center layer outputs are used

for its estimation. In this last stage can be identified final winner neuron. If Center neuron output is lower than temporary winner, then center neuron is declared as final winner. Else temporary winner gets this title.

Self-training process of this structure is different than original SOM structures. In self-training process must be adopted not only input weights, although inner relations. During this SOM structure self-training is used main principal, that winner input weights are updated. Inner relations from winner node perspective are considered as input weights. Therefore, how is shown in Algorithm 4., depending on which neuron wins, quantity of updated weights can vary. When center neuron is the winner, 4 inner and input weights are updated. If winner is in the last external layer only input weights are changed. In other cases 2 inner and input weights are updated. Inner weights change is slightly different than input weights:

$$\boldsymbol{\omega}_{inn}(n+1) = \boldsymbol{\omega}_{inn}(n) + \eta(n)(\mathbf{y}(n) - \boldsymbol{\omega}_{inn}(n)). \tag{2.11}$$

These weights adopted not to SOM input, although to neighboring neuron output. Other self-training parameters like self-training ratio can be used as same as in simple SOM structure.

Algorithm 4. SOM with inner weights winner takes all self-training

```
1  Procedure: Self-training
2    if Winner is center neuron
3      Center neuron input weights ← update
4      Inner weights form N_1, N_2, N_3, N_4 ← update
5    else if Winner is in first layer from center
6      Winner neuron input weights ← update
7      Inner weights form neighbors ← update
8    else if Winner is in external layer
9      Winner neuron input weights ← update
10     Inner weights form neighbors ← update
11   else Winner is in external last layer
12     Winner neuron input weights ← update
```

Four SOM structures with inner weights were tested Fig. 2.32. Test was used same as in this section beginning and Sections 2.3.1, 2.3.2. Quantity of external layers was different between these structure. It was changed from 1 to 4. In Table 2.10 is shown quantity of self-training cycles, which were used to self-train network. From table it is obvious, that to self-train network were used significantly less self-training iterations than marks general rule. In structure with 1 and 2 external layers were used just 21 and 23 self-training iterations

for 1 neuron. In larger network with 3 external layers were used 60 cycles for 1 neuron. In the last structure for 1 node were used 476 self-training cycles, however it hasn't achieved satisfactory result.

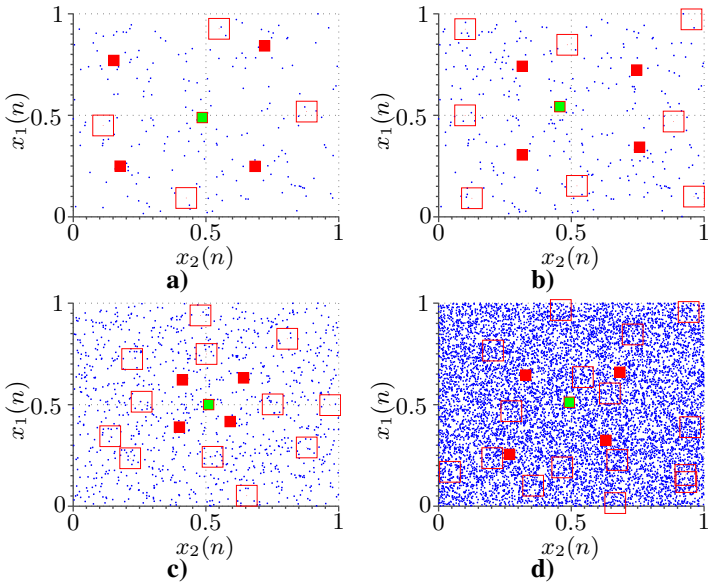


Fig. 2.32. SOM structure after self-training, when used: a) 1 external layer; b) 2 external layers; 3) 3 external layers; d) 4 external layers

Table 2.10. Self-training duration for various SOM structures

SOM structure	1 ex. layer	2 ex. layers	3 ex. layers	4 ex. layers
Self-training cycles	196	304	1026	10000

This network has a high self-training speed, specially in first 2 cases. However quality of self-training results must be determined before making final conclusions about this structure:

- SOM with 1 external layer Fig. 2.32a. In this case all neurons take their provided positions, therefore, can be concluded that this network is self-trained well.
- SOM with 2 external layers Fig. 2.32b. With additional external layer network keeps its structure. Just closest to center layer is pushed more to center.
- SOM with 3 external layers Fig. 2.32c. In this case network doesn't achieved full structural integrity. Few neurons in last external layer are

slightly shifted form provided positions. Yet other neurons took their places.

- SOM with 4 external layers Fig. 2.32d. This network even after 10^4 self-training cycles doesn't have regular structure. Just center and neighboring neurons took close positions to appointed spaces.

SOM structure with inner relations can significantly increase speed of self-training process. This achievement can reduce latency and complexity of network self-training. With this structure self-training ratio $\eta(n)$ can be implemented with lookup table, because in memory can be stored just few hundreds, although not thousands, values of exponent function. This structure has main disadvantage – with larger structures it's unstable. Therefore, it can't be used for task where high dimensionality is needed.

2.4. Self-training Implementation in Field Programmable Gate Array

SOM network in spectrum sensing system (Fig. 2.1) is marked as a decision maker. Its inputs are quadrature features or wavelets transform outputs. From these spectrum features SOM makes decision about spectrum occupancy.

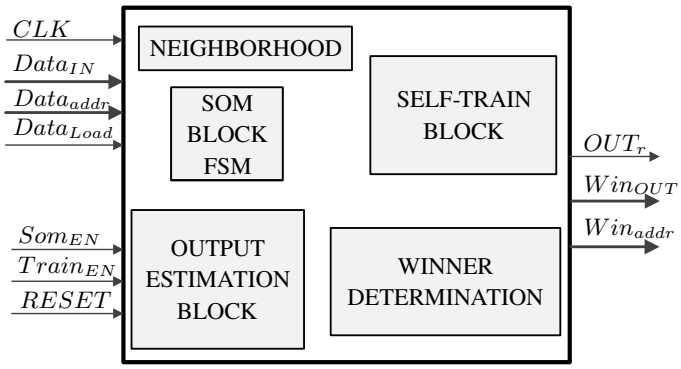


Fig. 2.33. SOM implementation based on FSM structure

This implementation has 7 inputs and 3 outputs. Inputs can be divided in 3 main groups. One input is responsible for block synchronization CLK. Second group consist of binary signals Som_{EN}, $Train_{EN}$ and $RESET$. These signals are used for SOM block mode setting:

- Som_{EN} – this signal enables SOM. If it is logical `0`, network is inactive;

- $Train_{EN}$ – it enables self-training mode in network;
- $RESET$ – global block reset. All components will be set to initial state after it.

Third group consist of $Data_{IN}$, $Data_{addr}$ and $Data_{Load}$ signals. This group is used to load input vector:

- $Data_{IN}$ – input vector;
- $Data_{addr}$ – input RAM address signal;
- $Data_{Load}$ – data load bit signal. If this signal is set to logical `1`, then input vector can be loaded to RAM.

SOM block has 3 outputs:

- OUT_r – if this bit signal is logical `1`, then SOM output is ready for reading;
- Win_{OUT} – winner distance from input;
- Win_{addr} – this signal shows position of winner neuron.

This network structure consist of SOM block finite-state machine (FSM) and 4 main blocks:

- output estimation block Fig. 2.34, it makes comparison between input and weights vectors;
- self-train block is responsible for weights update, which is made to winner neuron and its neighbors Fig. 2.38;
- winner determination Fig. 2.35, this block finds winner neuron from estimated distance vector;
- neighborhood Fig. 2.36, this block sets neighborhood bit signal according to winner position in network.

SOM block FSM is responsible for data exchange between blocks mediation and SOM modes setting. Main inputs of this block are Som_{EN}, $Train_{EN}$, $RESET$ and $Data_{Load}$. According to these signals FSM sets SOM modes:

- Global `RESET`. It's active, when $RESET$ signal is set to `1`. In this mode reset is passed to all RAM blocks and state registers.
- `Initialization`. This condition is activated after $RESET$ signal level changes form `1` to `0`. In this mode initial weights are set, by filing weight RAM with random numbers (Fig. 2.38).
- `Output calculation`. This mode is activated, when $Train_{EN}$ signal is logical `1` and others is `0`. In this condition just output estimation and winner determination blocks are active.
- `Self-train`. To this condition SOM is set by FSM, when $Train_{EN}$ is logical `1`. In this mode all SOM blocks are active, because all parts are needed in self-training process.
- `Inactive`. When $Train_{EN}$ is logical `0`, all other SOM conditions except global $RESET$ are unavailable.

Output estimation block, as mentioned in this section, compares input and weight vectors (Fig. 2.34). This block is controlled by FSM too. First mode is `idle`. This mode is active, when Som_{EN} is logical `0`, after `RESET` and before data is loaded to SOM. `idle` is initial condition of FSM. Another mode is `Data_Load`. In this case from outside this block is enabled Input and or Weight RAMs. All operations in block are halted, until new vectors is loaded to memory. During `Data_Load` mode all internal address generators are reinitialized. When RAMs are released FSM starts third output calculation mode.

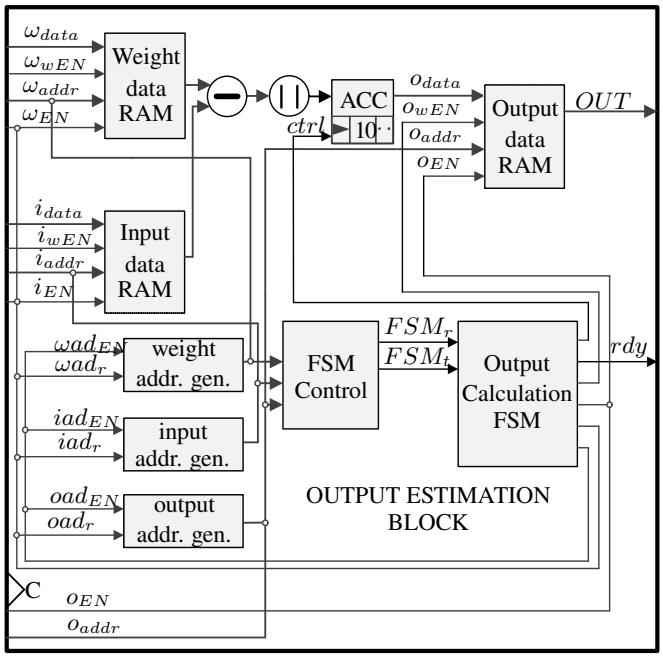


Fig. 2.34. SOM output estimation block implementation in FPGA

Third mode is called `Manhattan`. This stage is used for Manhattan distance calculation. This distance is more cost efficient than Euclidean, because Manhattan doesn't uses square and square root operations, which use important resources and have long latency. To implement Manhattan estimation, comparison between operands must be made before each subtraction. In order to get positive distance lower value must be subtracted from greater. This difference is accumulated and divided in to ACC accumulator. To calculate distance for neuron, weight w_{addr} and input i_{addr} addresses are incremented, until input vector boundary is reached. Then `output_save` FSM mode is initialized, in which accumulated distance is passed to output RAM. After `output_save`

begins `Next_output` mode, in which Input addr. gen. is reseted and o_addr is incremented for next neuron output calculation. And this Manhattan distance calculation is cycled, until weight address boundary is reached. When all outputs are calculated FSM goes to `idle` mode, in which rdy output signal is set to logical `1`. This change can be detected by outer logic. SOM output can be read from RAM by using o_{EN}, o_{addr} and OUT signals. To calculate 9 neurons outputs by using this implementation were used 189 clock cycles, 62 slice registers, 153 logical units and 3×18 kb block RAMs.

To implement SOM output calculation with inner weights (Fig. 2.31) into the structure (Fig. 2.34) must be added 2 RAMs and 2 address generators, which will be used for inner weights and temporary output storage. In FSM must be added one mode, which will be started after `Manhattan`. In new state additional distances to neuron output sum will be added. This modification for 9 neurons structure will use additional 42 clock cycles, 14 slice registers, 21 logical unit and 2×18 kb block RAMs.

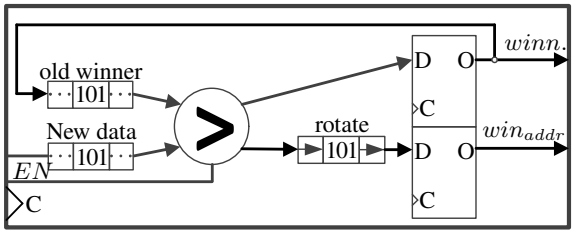


Fig. 2.35. SOM winner neuron determination block implemetatin in FPGA

Winner determination block is activated by main FSM after SOM output estimation Fig. 2.35. Winner is found from output vector. When EN is logical `1`, `New_data` is presented for block. This output vector sample is compered to Old_winner value. If new value is lower than older, New_data is declared as winner. And then value is passed to state register. Winner neuron address is marked in win_{addr} signal by bit rotation. If Old_winner is 2 neuron and new winner is 5, rotation is made by 3 bits. Bit length of win_addr signal is equal to neuron quantity. Therefore, if in network are 9 neurons, then win_{addr} signal can be `000010000`. Winner determination is small block and for 9 neuron comparison it would use 22 clock cycle, 31 slice registers and 46 logical units. When all samples of output vector ae compared main FSM enables neighborhood block.

In neighborhood block win_{addr} signal is converted to $neig$ (Fig. 2.36). $neig$ signal is used in self-training process. By it winner neuron neighbors are set. Block output signal bit length is 2 times longer than input, because for

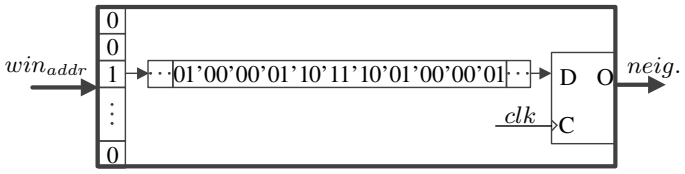


Fig. 2.36. SOM neighborhood block implementation in FPGA

each neuron are used 2 bits. These bits mark neuron neighborhood to winner neuron:

- `00` – not related with winner neuron;
- `01` – distant winner neighbor;
- `10` – close winner neighbor;
- `11` – winner neuron.

In Figure is shown neighborhood example, in which just closest neighbors are used. Then 32 bit $neig$ signal for this situation will be:

`00'00'00'00 | 10'10'10'00 | 10'11'10'00 | 10'10'10'00.`

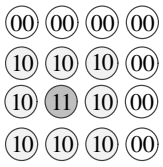


Fig. 2.37. SOM neurons neighborhood example

To prepare this (Fig. 2.37) $neig$ signal were used 6 clock cycles, 171 slice registers and 228 logical units. For winner takes all self-training strategy just wining neuron is marked in $neig$ bit signal. Therefore, for SOMs, which update just one winner neuron, neighborhood block is unnecessary.

Generated $neig$ signal is passed to self-train block. Self-train block is similar by structure and FSM modes to output calculation block. Main differences in structure are:

- less memory blocks;
- additional self-training ratio lookup table and random generator;
- different calculation engine.

To update SOM weights just input and old weight vectors are needed. Therefore, self-train block uses one RAM less, than output estimation. Old and new weights use same RAM, because weight calculation makes adequate delay for reading old and writing new values.

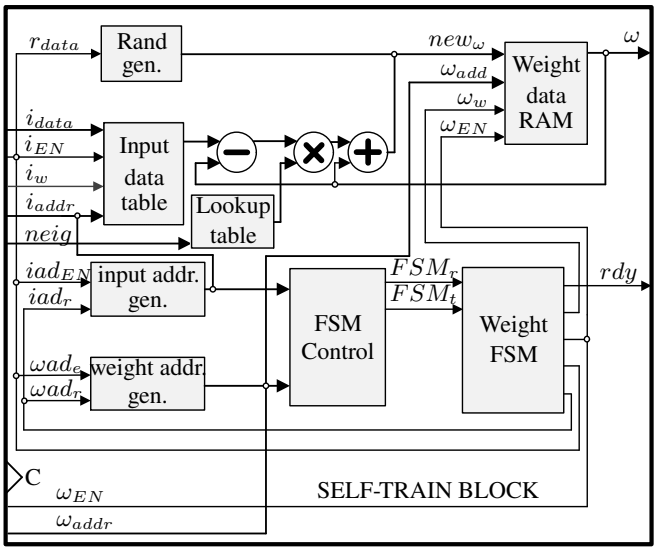


Fig. 2.38. SOM self-training block implementation in FPGA

Random generator is enabled once for initial weight load in to weight RAM. Generator is activated, when Global `RESET` mode is active in SOM. During this mode wad_{EN} and r_{EN} is set to logical `1`, until weight RAM is filed with initial random weights.

Another important component is self-training ratio $\eta(n)$ lookup table. Self-training ration is exponentially decreasing function equation (2.5), therefore, for implementation of this function are used many FPGA resources. If implementation is made by multiplication operations 28 DSP slices are needed. If it is implemented by Lookup table RAM is needed to store $\eta(n)$ values. To store 10^4 $\eta(n)$ values with 16 bits resolution 10×18 kb bock RAMs will be used. With greater accuracy or greater $\eta(n)$ quantity proportionally increases demand of block RAM. Lookup table is less resources demanding than multiplication operations, therefore, table approach is chosen for $\eta(n)$ function implementation.

For weight update must be used one multiplication operation. This operation is used to multiply self-training ratio with difference between input and weight. Then with this multiplication result old weight value is updated and passed to RAM for storing.

Self-train block FSM has some differences in comparing with output estimation too:

- `idle` – in this mode self-train block stays until $Train_{EN}$ is logical `0`;
- `data_Load` – just input vector can be loaded in to block;

- `Manhattan` mode is changed by `weight_update`, this mode latency is lower, because operation is done just with winner weights and its neighbors;
- `weight_save` – W_w is set to logical `1` for weight record;
- `Next_weight` – this weight update is cycled until weight vector boundary is reached.

To update 9 weights with this structure are used 201 clock cycles, 59 slice registers, 174 logical units, 2 DSP slice and 2×18 kb block RAMs.

As in output estimation block to implement inner weight update must be added 2 RAMs and 2 address generators for additional weights and outputs. Additional FSM mode for inner weights update must be added too. This mode will be started after input weight vector boundary is reached. This modification for 9 neurons weights update uses additional 247 clock cycles, 19 slice registers, 24 logical unit and 2×18 kb block RAMs.

In Sections 2.3.1 and 2.3.2 additional improvements to SOM, which shorten self-training period, were researched. First improvement was self-training process endtpoint tracking. This tracker implementation is shown in Fig. 2.39. It is used externally from SOM main structure Fig. 2.33. Tracker controls $Train_{EN}$ signal. By default this signal is set to logical `1`, until self-training process endpoint is reached. Tracker input is SOM winner neuron distance from output Win_{OUT}. To save FPGA resources input signal processing chain is optimized as in Manhattan distance calculation. Then multiplication and square root operations are saved (equation (2.7)). Processed signal is accumulated in ACC. Accumulator output is bitwise shifted to implement division. Counter sets set register output, when N_{wind} is reached. Then in register output is set self-training stop criteria. Stop criteria is compared with input average value, which is calculated in lower part of structure. If AVG_{IN} value becomes lower than $stop$, than EN signal will be set to logical `0`. This tracker uses 113 slice registers, 217 logical units and 1×18 kb block RAM. Latency of this block mostly depends on N_{wind}.

Another self-training process improvement must be embedded in to self-training block FSM Fig. 2.38. Assistant must be in block control chain, because it adjusts self-training process. For decision making assistant must have information about self-training process progress. Therefore, outputs Avg_{IN} and $stop$ from tracker must be added in to self-train block inputs. These 2 inputs in FSM `Assistant` mode will periodically control multiplexer. This multiplexer will be switched in to conditions, which are same as in Algorithm 2.. During these 5 conditions in most cases lookup table address will be changed. If self-training process is unsuccessful, some random value for SOM weights will be added. This SOM self-train block change will use additional 25 slice registers and 37 logical units.

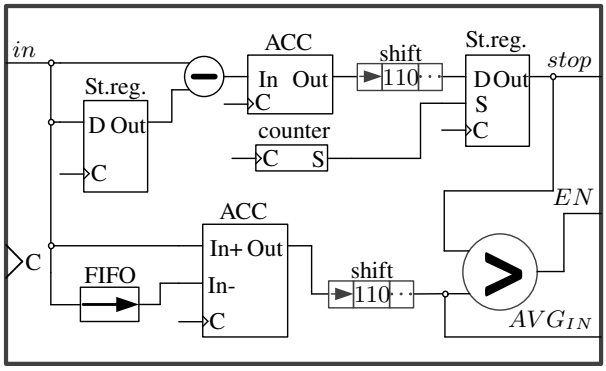


Fig. 2.39. SOM self-training process endpoint tracker implementation in FPGA

Table 2.11. FPGA resource utilization by SOM modifications

SOM modification	Slice registers	LUTs	DSP slice	Block RAM	Latency cycles
8000 self-train iterations	344	624	2	14	5704000
With endpoint detection	405	654	2	9	2523105
With assistant	431	693	2	7	1298280
With inner weights	383	675	2	6	175951

In Table 2.11 are four SOMs modifications, which are rated by self-training latency. Most delay have regular 16 neuron structure with rectangular neighborhood function. It used 5704000 clock cycles for 8000 self-training iterations. If this SOM structure is synchronized with 50 MHz clock, for self-training it will use 114.08 ms. 18 kb block RAMs mostly are used by regular structure too, because more self-train ratio values must be stored in lookup table then in others SOMs. When endpoint detector was added to SOM structure, self-training delay was shorten more than 2 times and self-training process took 50,46 ms. Self-training endpoint detection used more slice registers and logical units, however it helped to save 5×18 kb block RAMs. With assistant to self-train 16 neuron SOM network used 25,97 ms,. And assistant used just 7 block RAMs, because it needed even less self-training ratio values in lookup table than previous modification. Yet this improvement used 87 slice registers and 69 logical units more than regular SOM. SOM with inner weights has lowest self-training delay. To self-train SOM with 1 external layer were used 3.5 ms. Block RAM utilization by this structure was lowest too. With inner weights structure used 39 slice registers and 51 logic unit more than regular SOM network.

All SOM improvements helped to significantly shorten network self-training delay. With inner weights self-training latency was 32 times shorter than in regular SOM structure. In last modification block RAM was utilized more efficiently too. In Zynq 7020 SoC can be implemented more than 40 parallel SOMs with inner weights, when in regular network case can be implemented 2 times less. Other two SOM modifications save self-training time and block RAM too, however they are not such cost efficient as network with inner weights.

2.5. Conclusions of Chapter 2

1. The modification of the signal spectrum variance estimation formula decreases the total latency, received in the feature extraction phase and can be successfully used as a feature vector for the spectrum sensor.
2. The implementation of Haar wavelet transform, which can be used as a feature vector for the spectrum sensor, is three times more efficient comparing to Daubechies and Simlet wavelet transforms. In addition, near 15% FPGA resources can be save by removing the h_{HL} signal from feature extraction algorithm, keeping the same primary user signal detection rate.
3. An algorithm for SOM self-training endpoint selection, proposed in this dissertation, may efficiently stop the self-training process adaptively, ensuring the required SOM lattice structure organization and less number of required iterations, comparing to the methods, proposed in literature.
4. The SOM training assistant, proposed in this dissertation, helps to shorten self-training duration up to $\sim 50\%$ in comparing with endpoint selection algorithm. However assistant algorithm is less efficient with small SOM structures like 2×2 than endpoint detection algorithm.
5. The implementation efficiency of the SOM self-training algorithm can be increased by adding additional inner weights for the neurons in the lattice during SOM self-training process. This modification decreases up to 2.3 times memory cells used by network in FPGA based embedded systems, in comparing to memory required for methods, proposed in literature.

3

Spectrum Sensing Methods Experimental Research

In this chapter the experimental setup and results of experimental investigation are presented. The Experimental tests were performed in order to test the performance of the spectrum sensors when various alternative signal spectrum energy based feature vectors are used as an input to the detection system. In addition, a spectrum sensors based on the self-organizing map with different topology, lattice size and self-training algorithm modifications are tested experimentally. The experimental tests were performed in order to propose the most reliable features for spectrum sensor and to propose the most efficient structure and self-training algorithm for the implementation of the SOM based spectrum sensor in FPGA based embedded system.

The results of experiments, presented in this chapter are published in five papers (Stašionis, Serackis 2013; Stašionis, Serackis 2014; Serackis, Stašionis 2014; Stašionis, Serackis 2015a; Stašionis, Serackis 2015b) and presented at four international conferences (ELECTRONICS'2013; EUROCON'2013; NDES'2014 and EUROCON'2015).

3.1. Experimental Setup

The experimental tests were performed using the real radio signal recordings. The most of the tests were performed in offline mode, when the same previously recorded data was sent to embedded system with various configurations,

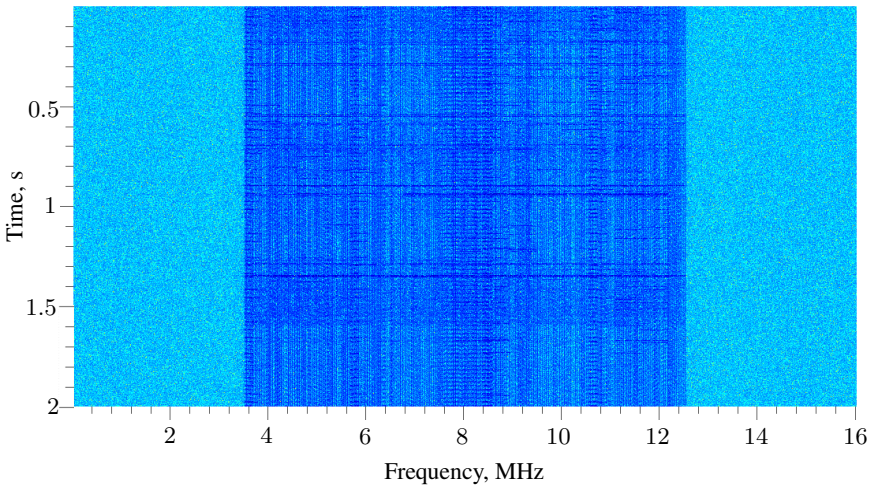


Fig. 3.1. RF spectrum record with center frequency $F_{\mathrm{C}} = 778\,\mathrm{MHz}$, bandwith BW$= 16\,\mathrm{MHz}$

with different feature extraction methods, different decision threshold selection techniques and their modifications. In addition, the spectrum sensor prototype was tested in online mode, when the radio spectrum was analyzed during continuous real-time sensing experiments in three different public areas.

3.1.1. Preparation of Experimental Data Records

To objectively test spectrum detection capabilities of various methods, these methods must be tested with four most common signal types in radio environment:

- wide-band (Fig. 3.1);
- narrow-band (Fig. 3.2);
- frequency hoping (Fig. 3.2);
- burst (Fig. 3.2).

In first example is shown record Fig. 3.1, which is taken with center frequency $F_{\mathrm{C}} = 778\,\mathrm{MHz}$. In this record is 8 MHz bandwidth DTV signal. DTV signals can be detected in wide range of RF spectrum from 470 MHz to 790 MHz. Therefore Radio spectrum detector must have capability to detect DTV signals presence.

In second record $F_{\mathrm{C}} = 391\,\mathrm{MHz}$ are few narrow-band signals BW $=$ 30 kHz Fig. 3.2. This bandwidth or narrower signals can be emitted in various bands of RF spectrum. Spectrum detectors must detect not just wide-band yet narrow-band emissions too.

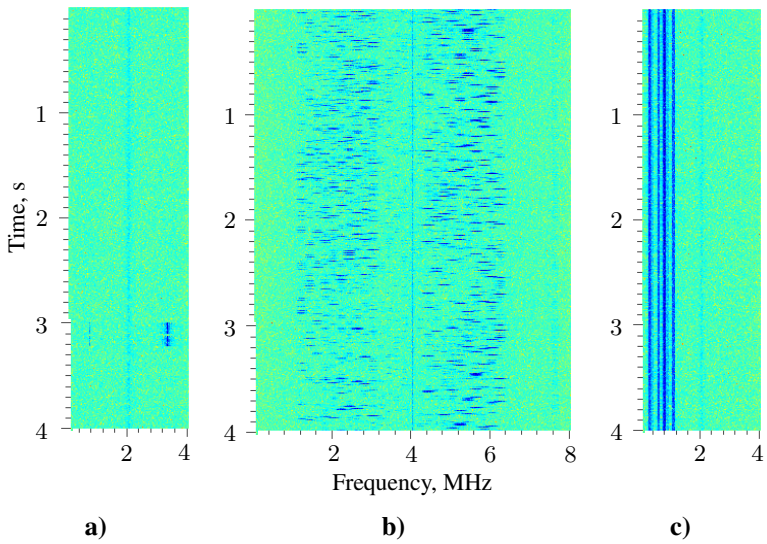


Fig. 3.2. Three RF spectrum records with: a) $F_{\mathrm{C}} = 391\,\mathrm{MHz}$, BW$= 1\,\mathrm{MHz}$; b) $F_{\mathrm{C}} = 928\,\mathrm{MHz}$, BW$= 8\,\mathrm{MHz}$; c) $F_{\mathrm{C}} = 433\,\mathrm{MHz}$, BW$= 1\,\mathrm{MHz}$

Another case of radio spectrum is shown in record $F_{\mathrm{C}} = 928\,\mathrm{MHz}$. In this record are signals with changing carrier frequency (frequency hopping, burst time few milliseconds). Bandwidth of these signals are few hundred kHz. More and more customer device can emit various modes of these type signals. Therefore algorithms must be capable to interact with frequency hopping signals.

In the last record is one-shot narrow-band burst signal with BW$= 20\,\mathrm{kHz}$. It can be caused by various devices like house and car alarm systems, portable radios, modems and etc. Because in RF environment, as mentioned, are variety of short active devices, spectrum detector must locate this type emissions too.

RF spectrum detector efficiency will be determined by how well it can detect various types signals. Therefore detector must be evenly performant with wide-band, narrow-band, frequency hoping and burst signals.

3.1.2. Selection and Configuration of the Hardware Tools

Requirements for RF spectrum detectors algorithms are also mandatory for equipment, which is used with these methods. Therefore RF hardware must operate in wide spectrum range, to be capable to detect vacant spectrum parts in various RF bands. To identify wide-band primary user signals equipment

Fig. 3.3. Wide band antenna

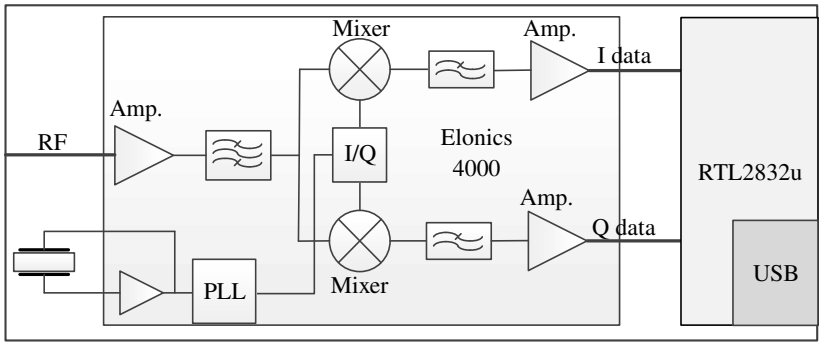


Fig. 3.4. Low cost SDR structure

must have wide real time bandwidth. Hardware RF part must have good sensitivity, because it must detect even distant emissions. ADC must be chosen with high sampling rate, that it would be capable to sample short pulses. All these requirements were taken into account when equipment for experiments was selected.

Wide-band omnidirectional antenna (Fig. 3.3) was chosen for investigation of RF spectrum detection algorithms. It works from 0.4–2.6 GHz with nominal 5 dBi gain. Therefore this antenna is right to use for wide range RF spectrum surveys in open and closed environments.

3 hardware sets were made and used for experiments:

- low cost SDR (Fig. 3.4);
- SDR based on high performance receiver (Fig. 3.5);
- wide-band SDR (Fig. 3.6).

For the first experimental investigations a modified low cost USB DVB-T receiver was used. This USB dongle is based on two chips: RTL2832u demodulator and Elonics E4000 receiver. The RF bandwidth up to 2.8 MHz can be processed by using this receiver. Together with DVB-T USB dongle, freeware HDSDR (High Definition Software Defined Radio) and GNU-Radio software

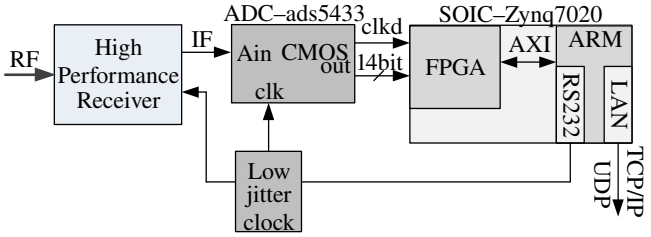


Fig. 3.5. Structure of SDR based on high performance receiver

were used to record and process I/Q samples of RF spectrum. Elonics E4000 is a wide-band receiver, with input frequency range from 64 to 1700 MHz. Declared by documentation sensitivity is up to -105 dBm, however after measurement established sensitivity was -97.7 dBm. Receivers Input referenced third interception point – IIP3 is 5 dBm. The simplified structure of the E4000 is shown in Fig. 3.4. The receiver has two analog outputs in-phase "I" and quadrature "Q", which are sampled and streamed through USB by the use of RTL2832u demodulator. In order to keep I/Q data not demodulated the dynamic-link library of demodulator was changed. Main disadvantages of this hardware solution were low bandwidth (less than 8 MHz) and low measured sensitivity.

Therefore to overcome these limitations SDR, which is based on high performance receiver (Fig. 3.5), was developed. Receiver intermediate frequency – IF is sampled with external ADC. These samples are passed to SOIC – Zynq7020 FPGA part. By FPGA processed data is transferred to ARM, which is responsible for receiver control and data streaming.

With this high performance receiver measured sensitivity of experimental hardware was increased to -113.4 dBm, bandwidth to 8 MHz and IIP3 9.8 dBm. Input frequency range is from 25 to 2000 MHz. To sample receiver IF was chosen ads5433 ADC, which has 14 bit resolution and up to 80 MSPS sampling rate. ADC is synced by low jitter clock 8 ppm. Quality of conversion was ensured by this clock. Differently than in low cost SDR hardware implementation some or all RF signal processing can be done in internal Zynq7020 FPGA (Section 2.2). All implementations of thesis were tested in this hardware platform.

Main disadvantage of this SDR implementation (Fig. 3.5) is limited bandwidth by 8 MHz. Therefore detection of wider communications is complicated. This limitation emerges from used high performance receiver. It's IF output is just 8 MHz width, when ADC can sample 40 MHz. An expensive update must be done to improve this platform. High performance receiver must be

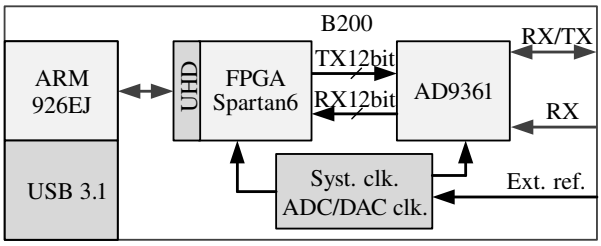


Fig. 3.6. Wide-band SDR B200 structure

changed in to wider version. Therefore further development of this platform was stopped.

New SDR development possibilities emerge with new Analog devices RF transceiver AD9361 chip. This single chip has 2 transceivers, which can work from 70 to 6000 MHz with bandwidth up to 56 MHz. It can work in 3 times longer frequency range with 7 times wider bandwidth than high performance receiver. Even so AD9361 has lower measured sensitivity -106.1 dBm, IIP3 8.2 dBm and internal converters resolution is 12 bit, it is a good choice for experimental hardware development.

USRP B200 SDR development kit (Fig. 3.6) was used for wide-band experiments. In this SDR is embedded AD9361 transceiver. As in previous hardware platform in B200 RX samples are passed to Spartan6 FPGA where it can be processed. Data from SDR is streamed trough USB3.1. Therefore to achieve maximum throughput to SDR, it must be used with USB3.0 or better interface.

Low cost SDR was efficient in the beginning of experimental researches, although for further development its parameters were too poor. With second SDR were implemented all spectrum extraction and detection algorithms, which were mentioned in this thesis. Unfortunately new receiver for it was too expensive. For further wide-band algorithm development and research was used USRP B200 SDR kit, which have superior real time bandwidth and frequency range. All experiments were made with same wide-band antenna (Fig. 3.3).

3.2. Experiments of the Sensors Using Different Features Extraction Methods

In this section all RF spectrum features extractors and decision makers will be tested with USRP B200 SDR development kit and wide-band antenna. Experiments will be made in real-time and with RF records (as is in Fig. 3.1 and 3.2).

At first all methods will be tested in non real-time meaner, because to test all feature extractors and decision makers combinations in same RF environment there is not enough equipment resources. Combinations, which will be tested in real-time for several days in different locations and bands, will be selected after non real-time experiments.

In experiments will be tested features extractors, which were discussed in Sections 2.2.2 and 2.2.3. Receiver operating characteristic (ROC) will be measured for these extractors. For each extractor in ROC will be marked how emission detection probability depends on false alarm probability. In each case 4 ROC will be measured. These curves will represent extractor capabilities to highlight in RF spectrum narrow-band, burst, frequency hoping and wide-band signals. ROC points will be measured by applying various thresholds, which will mark decision about RF spectrum occupancy boundary. Verification of features extraction methods results will be made by more accurate cyclostationary features extractor. By cyclostationary method will be determined positions in frequency, moments and quantity of emissions in RF records. These reference results will be compared with results, which will be achieved with researched extractors.

In second experiments stage from extractors will be selected most performant RF spectrum highlight methods, which will be used in networks tests. Test will be made with the same signals (burst, wide-band, narrow-band and freq. hopping), as it were made in extractors case. In networks experiments will be tested 10 unsupervised (SOM) and 1 supervised algorithms. Performances of SOM networks will be compared with supervised method and ROC experiments results. These 10 SOMs can be divided into 4 groups by training algorithms:

- general rule based self-training (neighborhood square, rhombus, hexagonal) Section 1.5.4;
- endpoint detection based self-training (neighborhood square, rhombus, hexagonal) Section 2.3.1;
- assistant based self-training (neighborhood square, rhombus, hexagonal) Section 2.3.2;
- SOM with inner weights Section 2.3.3.

Endpoint detection, assistant and SOM with inner weights are SOM modifications which help to save quantity of self-training iterations in comparing with general rule based self-training. Therefore the focus will be drawn to comparison of modified SOMs experiments results with general SOMs too. To find quantity of neurons needed for optimal detection performance for each network in experiments will be tested 8 different structure sizes. For all networks except SOM with inner weights will be tested same structure sizes 9, 16, 20, 25, 30, 36, 40, 45 $\mathcal{N}$ (SOM with inner weights 9, 17, 21, 25, 29, 37, 41, 45).

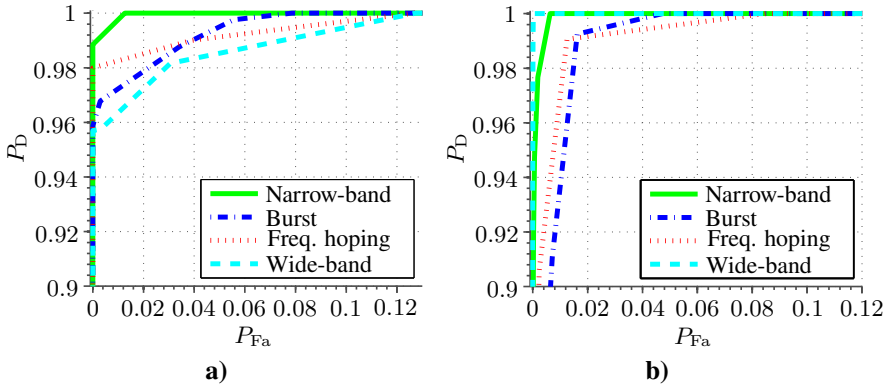


Fig. 3.7. ROC of: a) average – ϕ_A; b) variance ϕ_D extractors

In the third stage from previous two experiments will be selected most performant networks and extractors combinations, which will be tested in online experiments. Main aims of these test are to show networks performance in real time and RF spectrum occupancy of various bands. Two RF bands will be selected for experiments. One RF band will be in unlicensed range, other – in densely occupied RF band. Results from these bands will show how networks works in differently occupied ranges and will serve as RF spectrum resource exploitation example. These experiments will be made in 3 different locations:

- city – densely populated area;
- countryside – intermediate populated area;
- village – rarely populated area.

These locations will show how RF spectrum resources can be optimized by SOM based spectrum sensors in differently populated areas.

Results of average and variance implementations are shown in (Fig. 3.7). Feature extractors are tested with 4 signal types. Extractors performance for various signal types is different.

Average extraction – ϕ_A (Fig. 3.7):

- Wide-band signal detection probability $P_D = 0.96$ is achieved with low false alarm ratio $P_{Fa} = 0.0032$. Higher detection rates $P_D = 0.98$ and 1 are achieved with $P_{Fa} = 0.0306$ and 0.12 accordingly.
- Narrow-band signals detected $P_D = 0.9885$, when false alarm ratio was 0. With small increase of $P_{Fa} = 0.0126$, detection probability increases to $P_D = 1$.
- Frequency hoping signals detected $P_D = 0.98$ with $P_{Fa} = 0$. Higher detection rates 0.99 and 1 are achieved with more significant $P_{Fa} = 0.039$ and 0.127 accordingly.

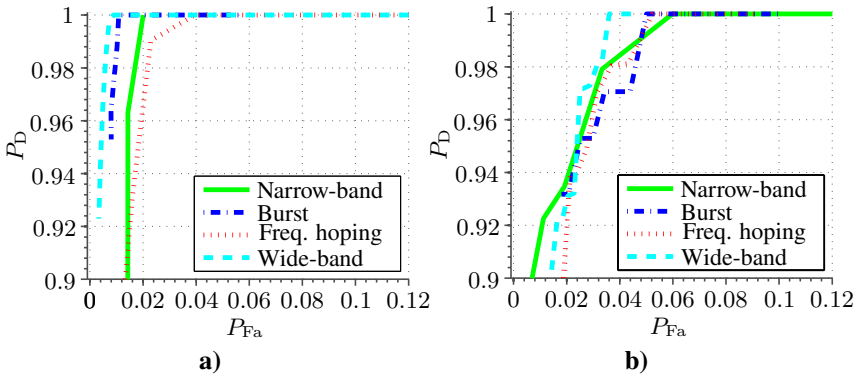


Fig. 3.8. ROC of variance without: a) FIFO – ϕ_{D_N}; b) multiplication – ϕ_{D_M} extractions

- Burst signals detection probability was P_D = 0.97 with P_{Fa} = 0.0029. And P_D = 1 was achieved, when false alarm ratio increased to 0.078.

ϕ_A is most effective with narrow-band signals. Worst results were achieved with wide-band signal extraction. Higher P_D for burst signals was achieved with significant P_{Fa} increase.

Variance extraction – ϕ_D (Fig. 3.7):

- Wide-band signal detection probability increased to P_D = 1, when false alarm ratio was just P_{Fa} = 0.00017.
- Narrow-band signals P_D was 0.9767 with false alarm ratio 0.0018. Full detection probability was achieved, when P_{Fa} reached 0.0056.
- Frequency hoping signals detection probability increased to 0.99 with P_{Fa} = 0.0124. P_D was 1, when false alarm ratio was P_{Fa} = 0.084.
- Burst signals detected with P_D = 0.9925 when P_{Fa} = 0.0161. All emissions were detected when P_{Fa} was 0.0473.

In comparing with ϕ_A, ϕ_D was more performant in all signals types extractions. ϕ_D achieved higher P_D rates with lower P_{Fa}.

In Section 2.2.2 two variance implementation modifications were discussed (Fig. 2.5). ROC performance of variance implementations without FIFO and multiplication was also different for various signals types.

Variance without FIFO – ϕ_{D_N} (Fig. 3.8):

- Wide-band signal detection probability reached P_D = 0.9614 with P_{Fa} = 0.00465. Full detection was achieved when P_{Fa} was 0.0073.
- Narrow-band signals detected with P_D = 0.9631 when P_{Fa} = 0.0143. Detection probability achieved P_D = 1 with higher P_{Fa} = 0.02 rate than false alarm probability was with ϕ_A and ϕ_D extractors.

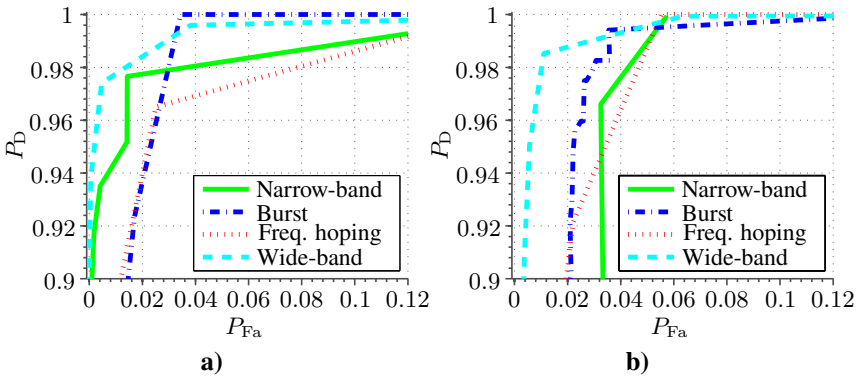


Fig. 3.9. ROC of Std. deviation: a) with FIFO – ϕ_σ; b) without FIFO – ϕ_{σ_N} extractions

- Frequency hoping signals detected more accurate (than with ϕ_A and ϕ_D), because $P_D = 1$ achieved with significantly lower $P_{Fa} = 0.03944$.
- Burst signals detection ROC was more performant too. P_D reached 1 with low $P_{Fa} = 0.011$.

ϕ_{D_N} implementation showed better extraction performance than ϕ_A and ϕ_D in frequency hoping and burst signal cases. In wide-band signal extraction this modification was less performant than ϕ_D, however it was better than ϕ_A.

Variance without multiplication – ϕ_{D_M} (Fig. 3.8):

- Wide-band signal detection probability in comparing with previous ϕ_D and ϕ_{D_N} extractors dropped. P_D reached 1 with $P_{Fa} = 0.036$.
- Narrow-band signals fully detected with significantly higher false alarm probability $P_{Fa} = 0.059$.
- Frequency hoping signals extraction ROC is worse than in ϕ_{D_N}. $P_D = 1$ was reached when $P_{Fa} = 0.05227$.
- In burst signals case this ϕ_{D_M} has less accuracy than ϕ_{D_N}. Detection probability reached 0.97 and 1 with $P_{Fa} = 0.0314$ and 0.05227 accordingly.

ϕ_{D_M} ROC in all signal type cases was worse than ϕ_{D_N}.

Two more quadrature extractors implementations were discussed in Section 2.2.2 std. deviation – ϕ_σ and std. deviation without FIFO – ϕ_{σ_N}. Overall these two extractors ROC are significantly worse then it were in ϕ_A, ϕ_D, ϕ_{D_N} and ϕ_{D_M} cases. Therefore ROC of these two implementations will be discussed jointly:

- Wide-band signal ROC with both ϕ_σ, ϕ_{σ_N} are worse than curves was in ϕ_D and its modification cases. By ϕ_σ $P_D = 1$ was reached with $P_{Fa} = 0.2$. ϕ_{σ_N} full emission interception was achieved with $P_{Fa} = 0.064$.

- Narrow-band signals full detection was achieved by ϕ_σ with high false alarm ratio 0.167. Better results were reached with modification $P_{\mathrm{D}} = 1$ when $P_{\mathrm{Fa}} = 0.0574$.
- Frequency hoping signals were detected by $\phi_{\sigma_{\mathrm{N}}}$ more precise than with unmodified implementation too. $\phi_{\sigma_{\mathrm{N}}}$ full interception reached with $P_{\mathrm{Fa}} = 0.05526$, when this result achieved ϕ_σ with $P_{\mathrm{Fa}} = 0.1495$.
- With burst signals better performance was achieved by ϕ_σ $P_{\mathrm{D}} = 1$ was reached with significantly low false alarm ratio 0.0344. In $\phi_{\sigma_{\mathrm{N}}}$ case this boundary was achieved with $P_{\mathrm{Fa}} = 0.1467$.

From ROC results it's clear that these two RF spectrum extractions implementations in all signal types are outperformed by less complex solutions.

In Section 2.2.3 were introduced three wavelet transforms implementations, which were used as RF spectrum features extractors. Therefore for these transforms Haar – ϕ_{Haar}, Daubechies – ϕ_{Deb} and Symlet – ϕ_{Sym} ROC is measured too. Two stage transforms were used for experiments. In ϕ_{Deb} and ϕ_{Sym} cases for tests 6–8 filter bank structures were implemented. For each transform output (h_{HH}, h_{HL}, h_{LH} and h_{LL}) ROC is measured with the same signals as it was measured with the quadrature extractors.

ROC measured for each output signal of ϕ_{Haar}:

- $h_{\mathrm{HH}}^{\mathrm{Haar}}$ output. In this output worse results were achieved for burst and narrow-band signals. Yet in these signal cases $P_{\mathrm{D}} = 1$ achieved with $P_{\mathrm{Fa}} = 0.03112$ and 0.03267 accordingly. Frequency hoping and wide-band signals extracted with significantly high accuracy. Full emission detection reached with 0.01 false alarm ratio in frequency hoping case and 0.017 in wide-band signal case.
- $h_{\mathrm{HL}}^{\mathrm{Haar}}$ output. This output performance is slightly worse in narrow-band and burst signal cases than results achived in $h_{\mathrm{HH}}^{\mathrm{Haar}}$ output, however ROC is slightly better for other two signals. Burst and narrow-band signals detection probability $P_{\mathrm{D}} = 1$ in both cases reached with $P_{\mathrm{Fa}} = 0.033$. Frequency hoping $P_{\mathrm{D}} = 1$ achieved, when P_{Fa} was just 0.006.
- $h_{\mathrm{LH}}^{\mathrm{Haar}}$ output. In all signal cases except narrow-band ROC is worse than in previous outputs ($h_{\mathrm{HH}}^{\mathrm{Haar}}$ and $h_{\mathrm{HL}}^{\mathrm{Haar}}$). Full narrow-band signal emission detection was achieved with 0.027 false alarm ratio. Burst signal $P_{\mathrm{D}} = 1$ reached with $P_{\mathrm{Fa}} = 0.05$. Same detection ratio achieved in frequency hoping case with $P_{\mathrm{Fa}} = 0.017$.
- $h_{\mathrm{LL}}^{\mathrm{Haar}}$ output. In this output best results were achieved for burst signal. All emissions of burst signal were detected with 0.029 false alarm ratio. All other signals $P_{\mathrm{D}} = 1$ achieved with significantly higher P_{Fa}. False alarm ratio for wide-band signal was 0.05 and 0.02 for frequency hoping.

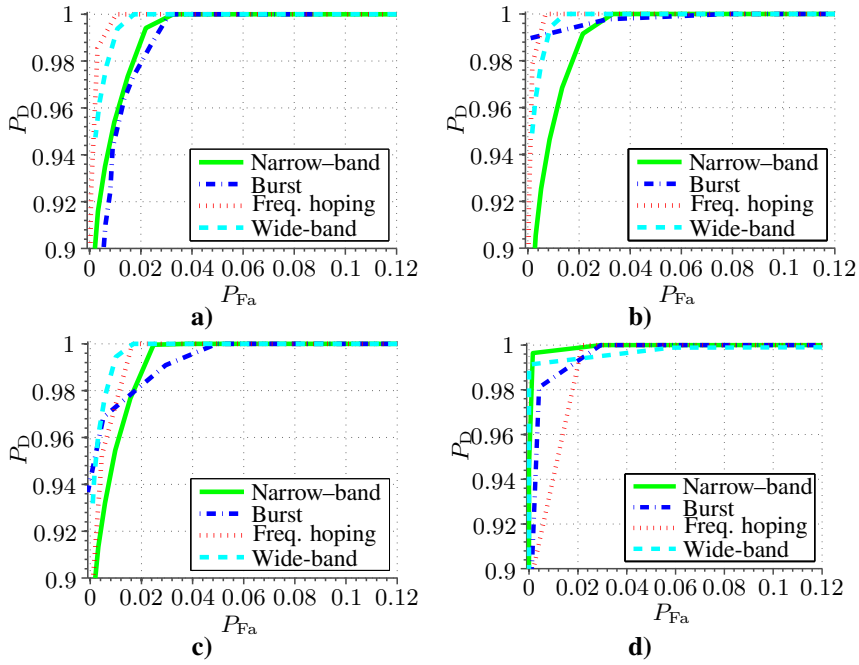


Fig. 3.10. ROC of Haar transform – ϕ_{Haar} outputs: a) $h_{\text{HH}}^{\text{Haar}}$; b) $h_{\text{HL}}^{\text{Haar}}$; c) $h_{\text{LH}}^{\text{Haar}}$; d) $h_{\text{LL}}^{\text{Haar}}$

ϕ_{Haar} each output is more accurate for some signal type: $h_{\text{HH}}^{\text{Haar}}$ – wide-band, $h_{\text{HL}}^{\text{Haar}}$ – frequency hoping, $h_{\text{LH}}^{\text{Haar}}$ – narrow-band, $h_{\text{LL}}^{\text{Haar}}$ – burst. Therefore by using all transform outputs, better extraction quality for various signals can be achieved.

ROC of ϕ_{Deb} transform:

- $h_{\text{HH}}^{\text{Deb}}$ output. In this Daubechies transform branch excellent results were achieved for wide-band signal. Full emission interception for this signal reached with P_{Fa} = 0.0044. Frequency hoping signals were extracted with higher false alarm ratio P_{Fa} = 0.012. Worse ROC is for burst signal, P_{D} = 1 reached with P_{Fa} = 0.06.
- $h_{\text{HL}}^{\text{Deb}}$ output. In this output best results were achieved for wide-band signal too. P_{D} = 1 for this signal was reached with even lower P_{Fa} = 0.002, than it was in $h_{\text{HH}}^{\text{Deb}}$ output. However burst signal full detection achieved with high false alarm ratio P_{Fa} = 0.1. Narrow-band signal 1 detection probability reached with 0.024 false alarm ratio.
- $h_{\text{LH}}^{\text{Deb}}$ output. Burst signal 1 detection ratio achieved with quite low false alarm probability P_{Fa} = 0.014. In this output better results than in others for narrow-band signal were attained. Narrow-band signal fully

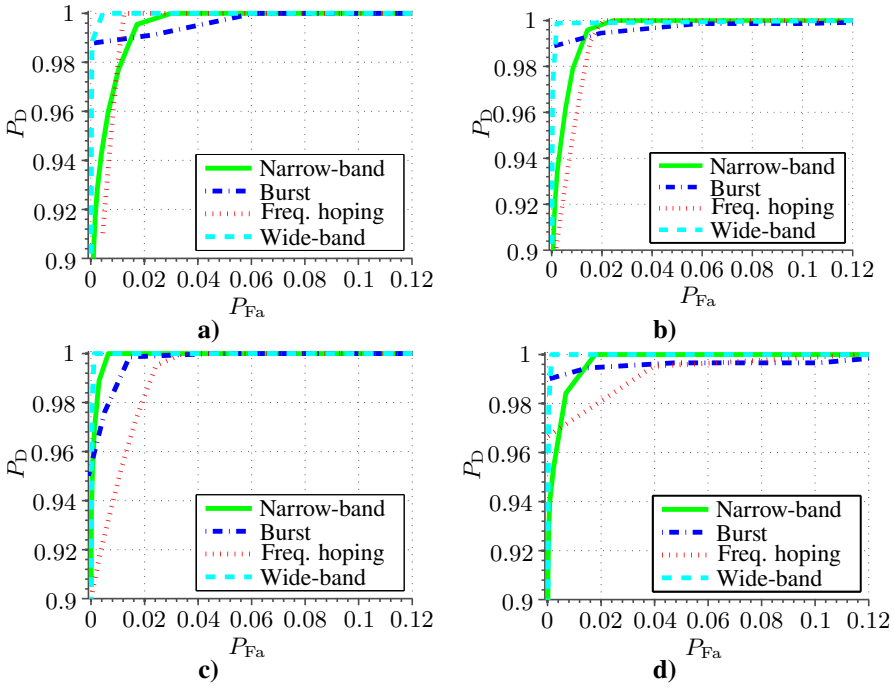


Fig. 3.11. ROC of Daubechies transform – ϕ_{Deb} outputs: a) $h_{\mathrm{HH}}^{\mathrm{Deb}}$; b) $h_{\mathrm{HL}}^{\mathrm{Deb}}$; c) $h_{\mathrm{LH}}^{\mathrm{Deb}}$; d) $h_{\mathrm{LL}}^{\mathrm{Deb}}$

detected with P_{Fa} = 0.0065. Frequency hoping signals extracted with high 0.034 false alarm ratio.

- $h_{\mathrm{LL}}^{\mathrm{Deb}}$ output. Burst and frequency hoping signals were detected with high false alarm probabilities. For frequency hoping case P_{D} = 1 reached with P_{Fa} = 0.1169 and for burst signal same accuracy attained with even greater P_{Fa} = 0.1369.

Differently than in ϕ_{Haar} in ϕ_{Deb} not all outputs are necessary. $h_{\mathrm{LL}}^{\mathrm{Deb}}$ output ROC of frequency hoping and burst signals are significantly worse than in others transform branches. And accuracy for other signal types isn't outstanding. Therefore in ϕ_{Deb} case better to use $h_{\mathrm{HH}}^{\mathrm{Deb}}$, $h_{\mathrm{HL}}^{\mathrm{Deb}}$ and $h_{\mathrm{LH}}^{\mathrm{Deb}}$ branches.

In all ϕ_{Sym} branches ROC was significantly worse than in ϕ_{Haar} andϕ_{Deb}:

- In $h_{\mathrm{HH}}^{\mathrm{Sym}}$ output P_{D} = 1 for frequency hoping and narrow-band signal attained with $\simeq$ 0.065 false alarm probability. P_{Fa} for same detection accuracy was even higher in wide-band and burst signals cases and it was $\simeq$ 0.11.
- $h_{\mathrm{HL}}^{\mathrm{Sym}}$ output. In this branch for wide and narrow-band signals were achieved acceptable results. For full emission detection P_{Fa} was 0.003

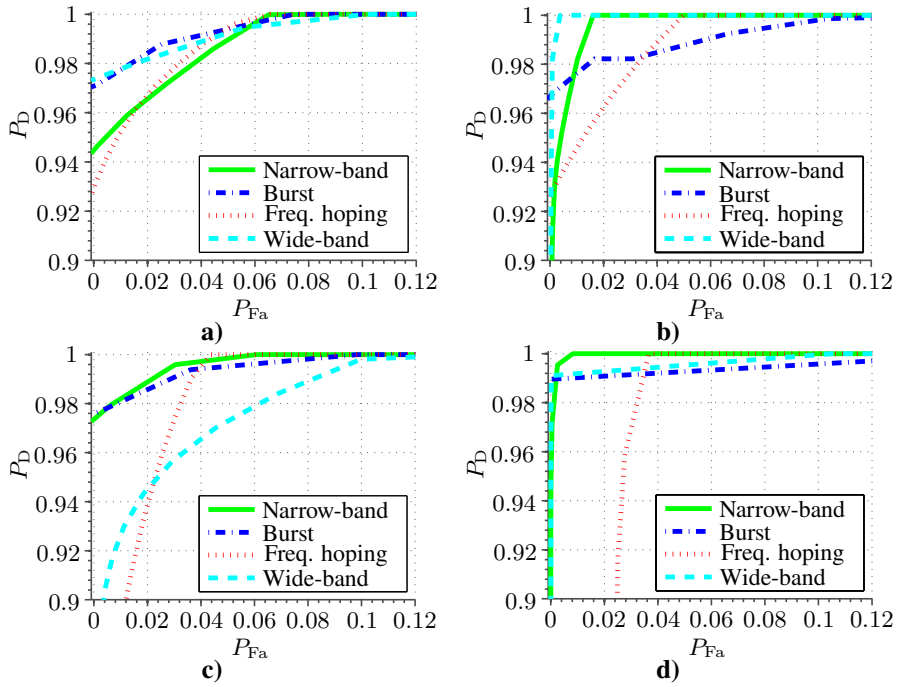


Fig. 3.12. ROC of Symlet transform – ϕ_{Sym} outputs: a) $h_{\mathrm{HH}}^{\mathrm{Sym}}$; b) $h_{\mathrm{HL}}^{\mathrm{Sym}}$; c) $h_{\mathrm{LH}}^{\mathrm{Sym}}$; d) $h_{\mathrm{LL}}^{\mathrm{Sym}}$

in wide-band case and 0.016 in narrow-band case. Frequency hoping signal P_{D} = 1 attained with P_{Fa} = 0.05.

- $h_{\mathrm{LH}}^{\mathrm{Sym}}$ output. Best ROC achieved for frequency hooping signals P_{D} = 1 with P_{Fa} = 0.043 and for narrow-band signals P_{D} = 0.996 with P_{Fa} = 0.03. Burst and wide-band signals extracted with high P_{Fa} as it was in previous branches.
- $h_{\mathrm{LL}}^{\mathrm{Sym}}$ output. Results for all types except narrow-band signal case were similar as in previous outputs. P_{D} = 1 for narrow-band signal attained with significantly low 0.008 false alarm probability.

As was mentioned in this section ROC analysis of ϕ_{Sym} performed poorly in comparing with ϕ_{Haar} and ϕ_{Deb}. ROC of ϕ_{Sym} all signal types in all outputs except few cases is significantly worse. Therefore this transform is less fit for RF spectrum feature extraction than previous two.

In this section ROC of 6 quadrature extractors and 3 wavelet transforms were measured. From these measurements it's clear that 3 quadrature extractors $\phi_{D_{\mathrm{M}}}$, ϕ_{σ} and $\phi_{\sigma_{\mathrm{N}}}$ have significantly poorer ROC characteristics than others extractors. Therefore these 3 extractors will not be used in further experi-

ments. With ϕ_{D} and $\phi_{D_{\mathrm{N}}}$ better results were achieved than it was attained with ϕ_{A}. However average value is necessary for variance calculations, therefore this extractor will be used in following experiments. From wavelet transforms worse ROC obtained with ϕ_{Sym}. It was less accurate for all signal types. So ϕ_{Sym} is excluded from further tests. ϕ_{Deb} transform $h_{\mathrm{LL}}^{\mathrm{Deb}}$ branch was less performant than others outputs specially in burst and frequency hoping signal cases, therefore this branch can not be used in further tests. So in next section experiments will be used ϕ_{A}, ϕ_{D}, $\phi_{D_{\mathrm{N}}}$, ϕ_{Haar} and ϕ_{Deb} transforms.

3.3. Experiments of the Spectrum Sensors based on Different Self Organizing Map Structures

In Section 3.2 most performant 5 features extractors sets were selected:

- ϕ_{AD} – Average and variance extraction. Variance showed better performance in ROC, yet for it calculation average estimate is needed. Therefore set of these estimates will be used.
- $\phi_{\mathrm{AD_N}}$ – Average and variance without FIFO. This variance modification helps save memory and shortens calculation delay. Therefore networks must be tested with this extractor set too.
- ϕ_{Haar} – Haar transform $h_{\mathrm{HH}}^{\mathrm{Haar}}$, $h_{\mathrm{HL}}^{\mathrm{Haar}}$, $h_{\mathrm{LH}}^{\mathrm{Haar}}$ and $h_{\mathrm{LL}}^{\mathrm{Haar}}$ extraction. This transform all branches showed good ROC performance.
- ϕ_{Deb} – Daubechies transform $h_{\mathrm{HH}}^{\mathrm{Deb}}$, $h_{\mathrm{HL}}^{\mathrm{Deb}}$, $h_{\mathrm{LH}}^{\mathrm{Deb}}$ and $h_{\mathrm{LH}}^{\mathrm{Deb}}$ extraction. In ROC LL branch showed less performance than others, however for better comparison networks experiments must be made with full transform too.
- ϕ_{Deb3} – Daubechies transform without $h_{\mathrm{LL}}^{\mathrm{Deb}}$ branch. Networks must be tested with transform which doesn't have less performant branch.

8 different network sizes were tested with every extractor in different RF bands. In all networks results tables and graphs are represented min, max and mean values of detection results. These values are calculated for each extractor.

Neural network. In narrow-band signals case neural network used least iterations with $\phi_{\mathrm{AD_N}}$ set $1.25 \cdot 10^4$. This result was achieved with smallest structure – 9 neurons. Longest training cycle $1.2475 \cdot 10^5$ iterations was achieved with ϕ_{Deb3} 25 neurons set. On average least training iterations were used with ϕ_{AD} $3.0813 \cdot 10^4$.

Best narrow-band signals emission detection performance was achieved with ϕ_{AD} (Fig. 3.13). However P_{Fa} ratio for this case was moderate in comparing with others extractors. Lowest false alarm ratio achieved with ϕ_{Deb} and network combination. Yet for ϕ_{Deb} case mean average P_{D} ratio was just 0.956. Therefore best results were achieved with ϕ_{AD} and network combination.

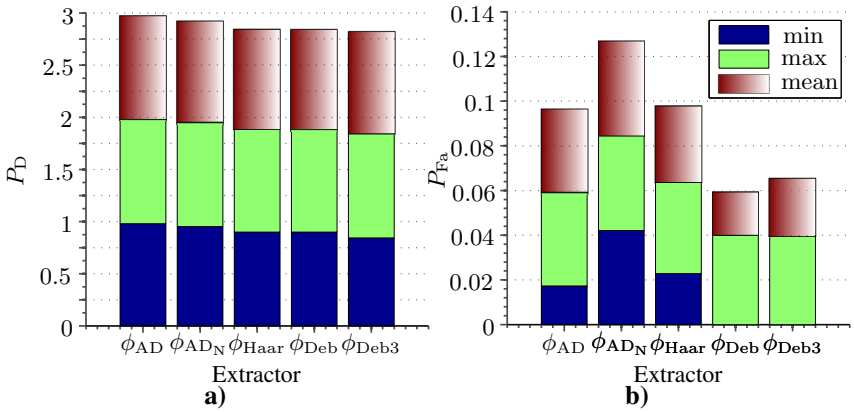


Fig. 3.13. Neural network: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

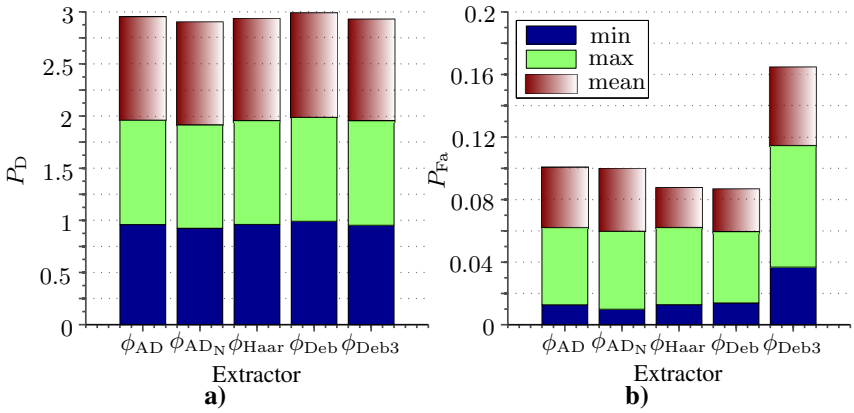


Fig. 3.14. Neural network: a) P_{D}; d) P_{Fa} ratios for burst signals

Least training iterations for burst signals were used in $\phi_{\mathrm{AD_N}}$ case $9.5 \cdot 10^3$. This result was reached with smallest network structure. Mean training duration was lowest in $\phi_{\mathrm{AD_N}}$ case $1.4351 \cdot 10^4$, too. Longest training period was with ϕ_{Haar} and 45 network structure combination $6.575 \cdot 10^4$.

In burst signal detection ϕ_{Deb} combination showed best results (Fig. 3.14). Mean P_{D} is greatest and P_{Fa} is lowest from all extractors and they are 0.997, 0.027 accordingly. However this combination averagely took 2 times more training iterations than $\phi_{\mathrm{AD_N}}$. In experiments least performant was ϕ_{Deb3}.

In wide-band signal case least training iterations were needed for ϕ_{Deb3} combination $9.86 \cdot 10^4$. This result achieved with smallest 9 neuron structure. However average training duration was lower in ϕ_{Deb} case $2.117 \cdot 10^5$. Longest average training period was in ϕ_{AD} case $2.876 \cdot 10^5$.

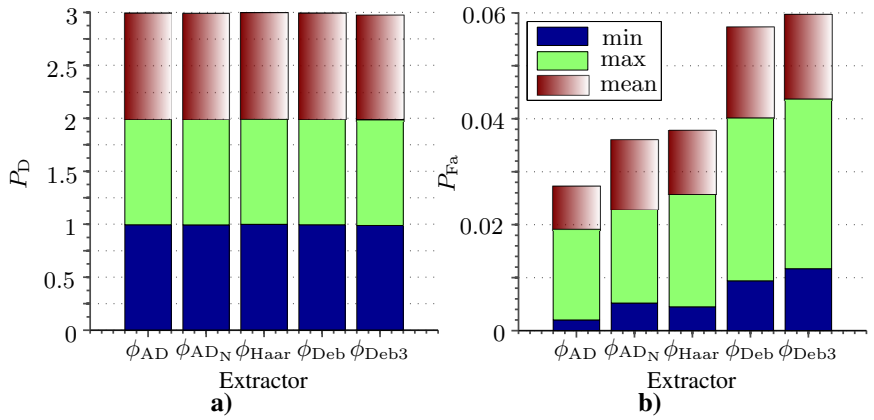


Fig. 3.15. Neural network: a) P_{D}; b) P_{Fa} ratios for wide-band signal

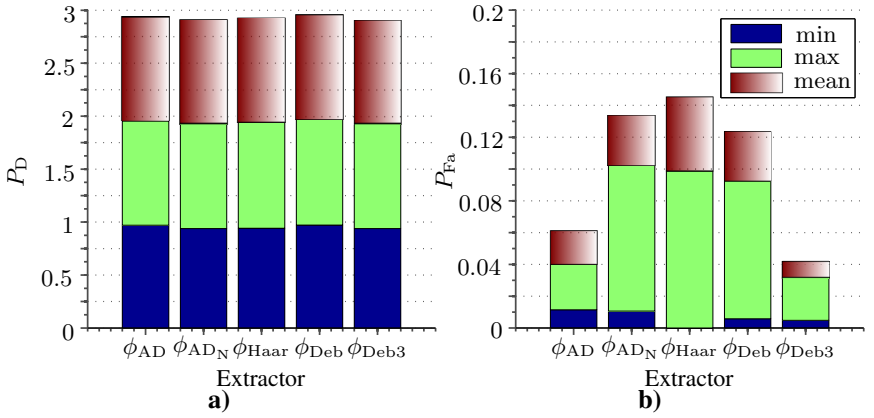


Fig. 3.16. Neural network: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Best detection performance (Fig. 3.15) was achieved with ϕ_{Haar}. Mean P_{D} for $\phi_{\mathrm{Haar}}$was even 1. Yet lowest P_{Fa} was achieved with ϕ_{AD}. Averagely false alarm ratio for ϕ_{AD} was just 0.008, when P_{D} reached 0.999. Therefore best common (P_{D} and P_{Fa}) detection performance was achieved with ϕ_{AD}. However ϕ_{AD} and network combination training duration was longest.

In frequency hopping signal case Least training iterations were needed for ϕ_{AD} and $\phi_{\mathrm{AD_N}}$ cases 32300 $3.23 \cdot 10^4$, when network structure had least neurons. Yet lowest mean duration was achieved with $\phi_{\mathrm{AD_N}}$ $1.098 \cdot 10^5$. Longest training periods was in ϕ_{Haar}combination. Average training duration for $\phi_{\mathrm{Haar}}$was 2 times longer than in $\phi_{\mathrm{AD_N}}$ case.

Table 3.1. Mean P_{D} and P_{Fa} for different neural network sizes

Network size	9	16	20	25
P_{D}	0.9843	0.9844	0.9853	0.9842
P_{Fa}	0.0307	0.026	0.0289	0.0302
Network size	30	36	40	45
P_{D}	0.9863	0.9795	0.9759	0.9865
P_{Fa}	0.0247	0.0273	0.0239	0.026

ϕ_{Deb} and network combination performed best at frequency hopping signal detection (Fig. 3.16). Mean P_{D} ratio for ϕ_{Deb} case 0.986. However for ϕ_{Deb} average P_{Fa} was one of largest. Worse detection ratio achieved with ϕ_{Deb3}, though mean P_{Fa} for this case was just 0.01. In frequency hopping detection best performance showed ϕ_{AD} set. Even so ϕ_{AD} wasn't most performant in duration, P_{D}, P_{Fa}, overall it had good results: mean training duration $1.253 \cdot 10^5$, P_{D} 0.98 and P_{Fa} 0.021.

In Table 3.1 are shown mean results of combined P_{D} and P_{Fa} values for network sizes, which were used in experiments. Highest P_{D} was achieved with 45 neurons 0.9865, and lowest P_{Fa} was in 30 neurons case 0.0247. Most performant size was 30 neurons because detection ratio is almost the same as with 45 neurons, and false alarm ration is significantly lower.

Overall (for all signals types) best results were shown by network with ϕ_{AD}. This combination in most cases has short training period, high mean P_{D} and low mean P_{Fa}. From all signal types network took significantly more training iterations to detect wide-band signal. For wide-band signal training period was 3–10 longer than for other signal types. Yet detection performance for wide-band signal type was the best.

General rule based self-training. This group of SOM is self-trained in accordance with the general rule, which defines quantity of self-training iterations for each neuron. It says that for full SOM adjustment must be used 500 self-training iterations for each neuron. Therefore for all networks (hquare, rhombus, hexagonal) will be used same amount of self-training iterations: 9–4500, 16–8000, 20–10000, 25–12500, 30–15000, 36–18000, 40–20000, 45–22500.

Square neighborhood function. Greatest narrow-band signal detection ratio (Fig. 3.17) was achieved with $\phi_{\mathrm{AD_N}}$ and square SOM combination. Mean P_{D} with $\phi_{\mathrm{AD_N}}$ set was 1. This result was reached with highest P_{Fa} too. Averagely false alarm ratio was 0.025. Lowest mean P_{Fa} was in ϕ_{Haar} case 0.007, with lower P_{D} 0.993. Therefore ϕ_{Haar} and square SOM had best performance (P_{D} and P_{Fa} combination). This combination result mean P_{Fa} is by 0.03418 lower than it was achieved with most performant neural combination.

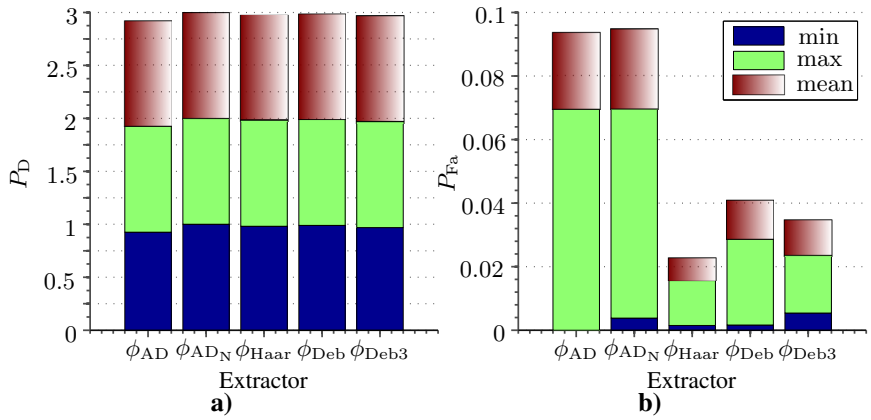


Fig. 3.17. Square SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

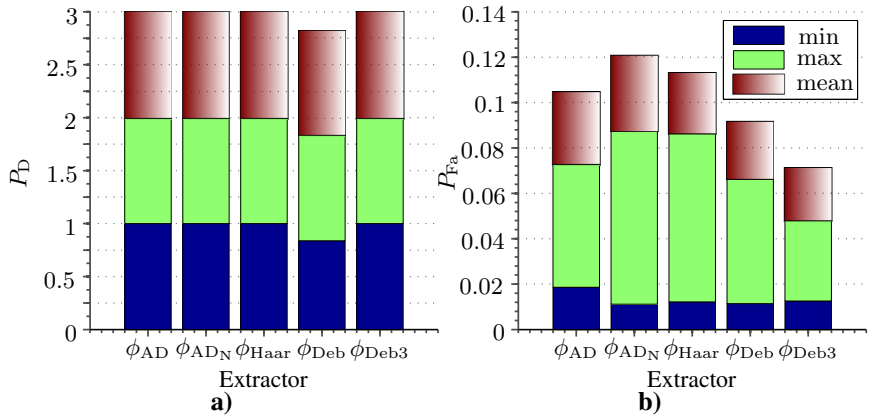


Fig. 3.18. Square SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

All burst signal emissions were detected by all combinations except ϕ_{Deb} (Fig. 3.18). In ϕ_{Deb} case mean P_{D} was 0.98, in other sets 1. Lowest min, max and mean P_{Fa} was in ϕ_{Deb3} and SOM combination, therefore this set is most performant. ϕ_{Deb3} and square SOM combination P_{D} and P_{Fa} was slightly better (by ~ 0.003) than best result achieved with neural network. However ROC of ϕ_{Deb3} was not outperformed by this SOM.

Highest detection ratio of wide-band signal was achieved by ϕ_{AD} set (Fig. 3.19), though P_{Fa} in this case was highest too. ϕ_{Haar} and SOM combination mean P_{D} was slightly lower (by 0.004), however it achieved significantly lower P_{Fa}. Therefore ϕ_{Haar} set had best performance for wide-band signal detection. In this signal type better P_{D} was achieved by neural network (by

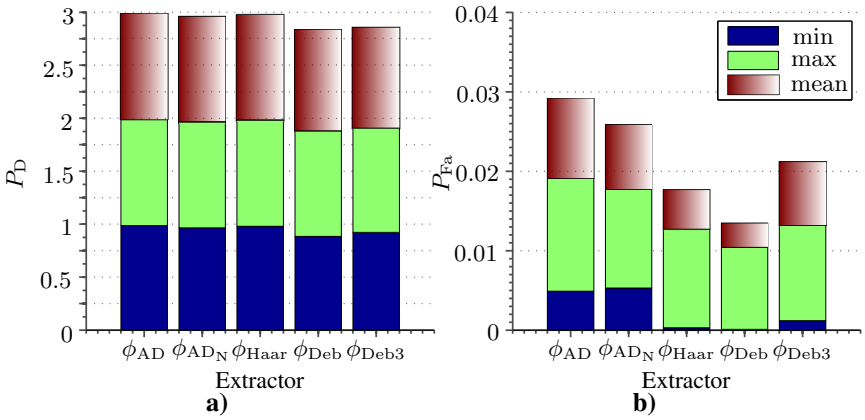


Fig. 3.19. Square SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

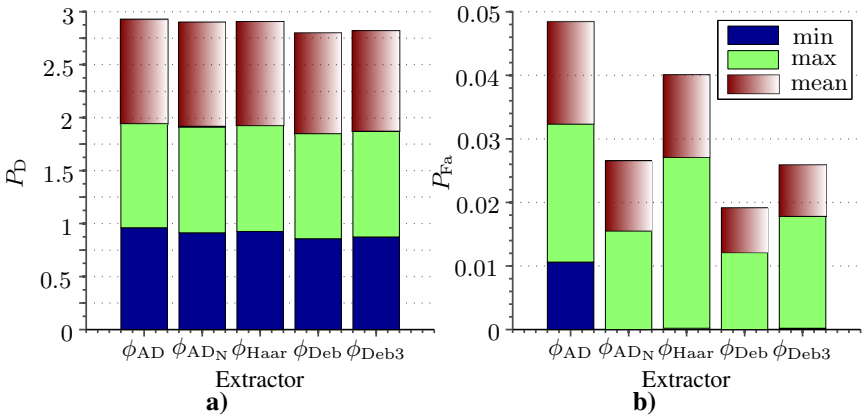


Fig. 3.20. Square SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

0.005). With this SOM mean P_{Fa} was lower by 0.003 than it was in supervised network.

Best performance in frequency hopping signal detection had $\phi_{\mathrm{AD_N}}$ combination (Fig. 3.20). Its mean P_{D} outperformed other SOM sets. This combination P_{Fa} was one of lowest too. This $\phi_{\mathrm{AD_N}}$ and square SOM combination false alarm ratio was by 0.01 lower than it was achieved with neural network best case. Square SOM outperformed ROC characteristic of $\phi_{\mathrm{AD_N}}$ too.

Best average detection ratio was achieved with 30 neurons SOMs 0.9949 (Table 3.2). For this size false alarm ratio is slightly higher (by 0.0001) than it was in 16 neurons size. Yet overall 30 neurons SOMs outperformed other network sizes. To self-train these networks took 15000 iterations.

Table 3.2. Mean P_{D} and P_{Fa} for different square SOM sizes

Network size	9	16	20	25
P_{D}	0.9553	0.9708	0.9863	0.9864
P_{Fa}	0.0171	0.0134	0.0173	0.0152
Network size	30	36	40	45
P_{D}	0.9949	0.9945	0.9933	0.9857
P_{Fa}	0.0135	0.0176	0.0144	0.0136

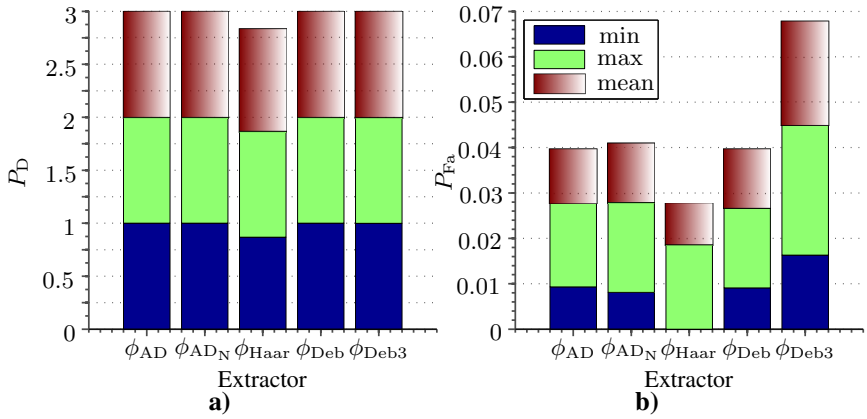


Fig. 3.21. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

In summary of this square SOM best results were shown by ϕ_{Haar} combination. In all signal types this set was most performant or was near the best results. Except wide-band signal this SOM type outperformed neural network results.

Rhombus neighborhood function. For narrow-band signals by all combinations except ϕ_{Haar} maximum detection (P_{D} = 1) ratio was achieved (Fig. 3.21). Though in ϕ_{Haar} case P_{Fa} was lowest 0.009. In ϕ_{AD} set false alarm ratio was slightly higher (by 0.003), therefore this combination performance in this case was the best. This ϕ_{AD} and rhombus combination showed better performance in narrow-band signal detection than neural network. Rhombus SOM outperformed ROC characteristic too.

All burst signal emissions were detected by all combinations (Fig. 3.22). Lowest mean P_{Fa} was achieved by $\phi_{\mathrm{AD_N}}$ 0.021. Rhombus SOM and $\phi_{\mathrm{AD_N}}$ set outperformed neural network best results too, yet ROC characteristic was not exceeded.

For wide-band signal extremely low mean P_{Fa} was achieved with ϕ_{Deb3} set, although detection ratio was poorest from all combinations (Fig. 3.23). Highest detection ratio was achieved with $\phi_{\mathrm{AD_N}}$ set 0.998, with low mean P_{Fa}

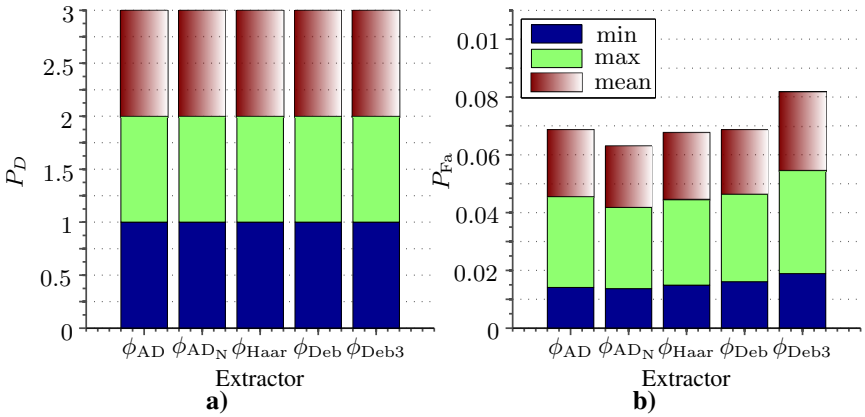


Fig. 3.22. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

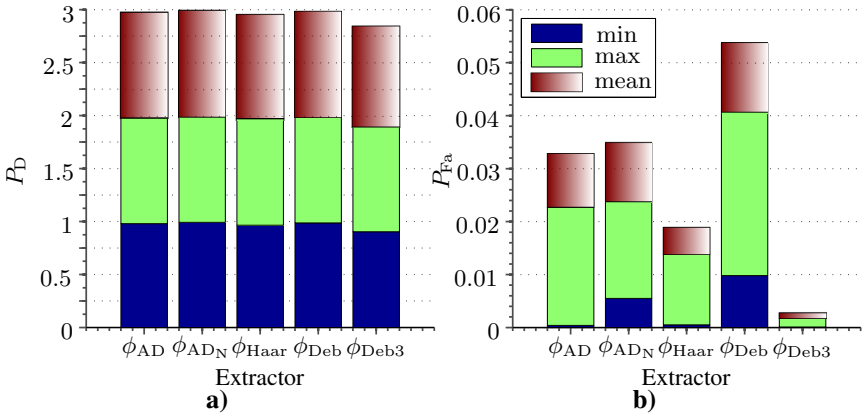


Fig. 3.23. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

0.01. $\phi_{\mathrm{AD_N}}$ and rhombus SOM combination was slightly less accurate than neural network.

Worse detection performance of frequency hopping signals was with ϕ_{AD} and ϕ_{Deb3} sets (Fig. 3.24).ϕ_{AD} mean detection ratios was just – 0.915 and ϕ_{Deb3} – 0.937. Highest detection performance reached by $\phi_{\mathrm{AD_N}}$ and ϕ_{Deb} combinations. Mean P_{D} for best cases was 1. Lower mean P_{Fa} was in $\phi_{\mathrm{AD_N}}$ set than in ϕ_{Deb}, therefore for frequency hopping signal with this network best results were shown by $\phi_{\mathrm{AD_N}}$. Rhombus and $\phi_{\mathrm{AD_N}}$ combination outperformed neural network results and ROC characteristic.

Best detection accuracy with one of lowest mean P_{Fa} was achieved by SOMs, which size was 40 neurons (Table 3.3). Achieved detection ratio by

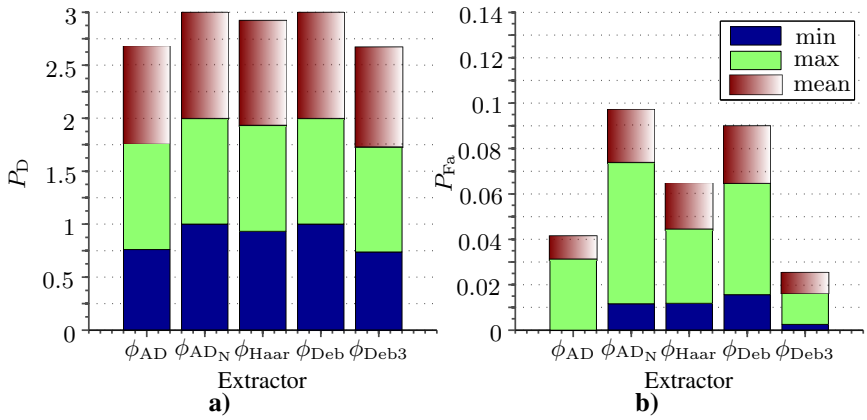


Fig. 3.24. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Table 3.3. Mean P_{D} and P_{Fa} for different rhombus SOM sizes

Network size	9	16	20	25
P_{D}	0.962	0.9888	0.9873	0.9946
P_{Fa}	0.0163	0.0165	0.0166	0.018
Network size	30	36	40	45
P_{D}	0.9943	0.9846	0.9967	0.9874
P_{Fa}	0.0154	0.0126	0.0135	0.0157

rhombus was higher than in neural network and square SOM cases. However in rhombus case this result was reached by larger structures, and for self-training this SOM took $2 \cdot 10^4$ iterations.

For all signal types best performance was shown by $\phi_{\mathrm{AD_N}}$ set. This combination just in narrow-band signal type was outperformed by ϕ_{AD} set. For wide-band signal better performance had neural network, although in other signal types rhombus SOM was more performant.

Hexagonal neighborhood function. From narrow-band signal detection (see Fig. 3.25) stood out ϕ_{Haar}set with very low detection ratio. Specially, when with other combinations P_{D} = 1. From successful combinations lowest P_{Fa} was in ϕ_{Deb} set 0.013. This combination outperformed neural network results.

With burst signals ϕ_{Haar} combination showed even worse results than it were with narrow-band signals (Fig. 3.26). For other combinations detection ratio was 1. Lowest mean P_{Fa} (except H) was in $\phi_{\mathrm{AD_N}}$ set 0.019. In burst signal case this hexagonal network outperformed neural network results.

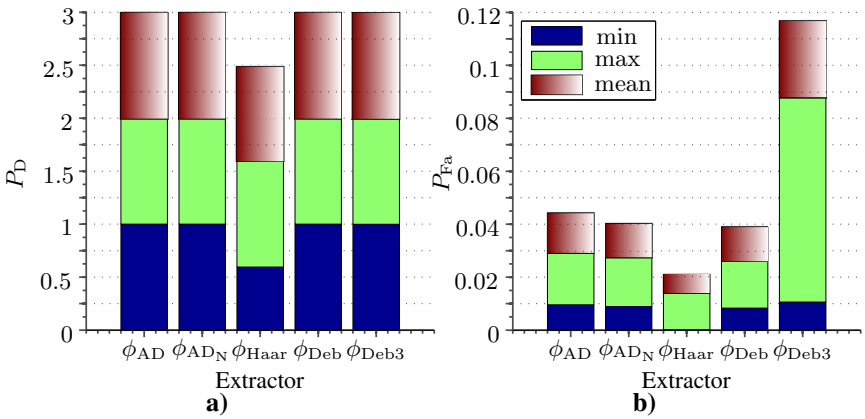


Fig. 3.25. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

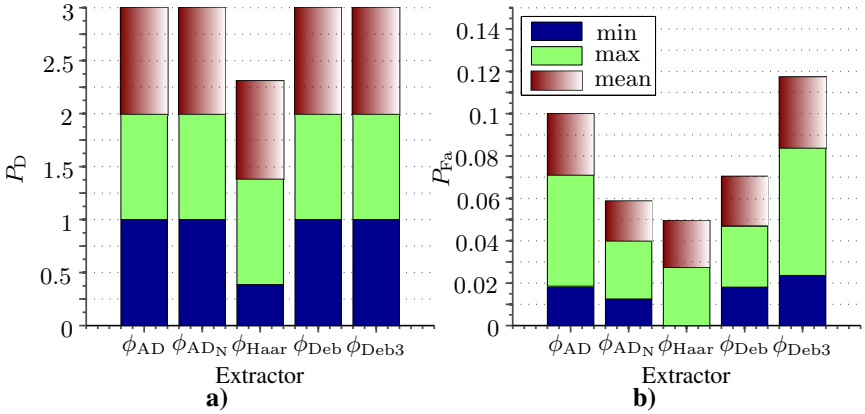


Fig. 3.26. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

In wide-band signal case ϕ_{Haar} set corrected its detection performance (Fig. 3.27). Although overall (mean P_{D} and P_{Fa}) performance was best in $\phi_{\mathrm{AD_N}}$ combination. Achieved detection ratio 0.998 with 0.009 false alarm ratio. Hexagonal SOM outperformed neural network and ROC results.

In frequency hopping signal case situation was opposite than in narrow-band and burst signal cases (Fig. 3.28). ϕ_{Haar} set stood out with best detection ratio 0.999, and in other combinations results were insufficient. Mean P_{D} in ϕ_{Haar} and hexagonal SOM combination was the same as it was in best neural case, yet P_{Fa} was by 0.004 lower with SOM.

As it was in neural and square networks, in hexagonal structure most efficient SOMs size was 30 neurons (Table 3.4). Yet with hexagonal network

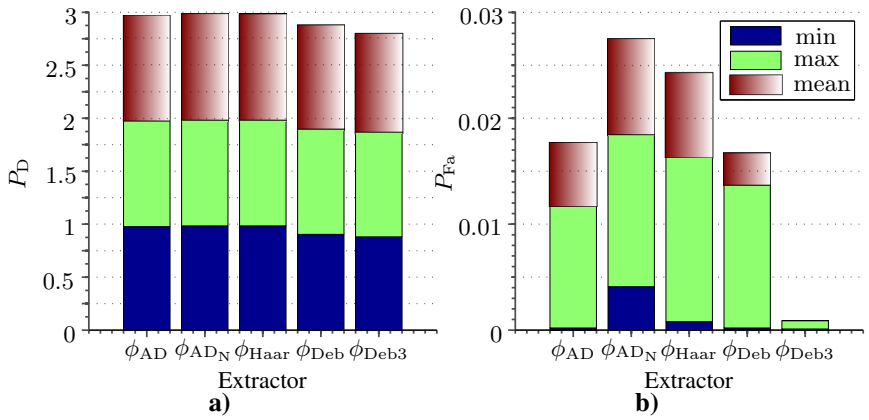


Fig. 3.27. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

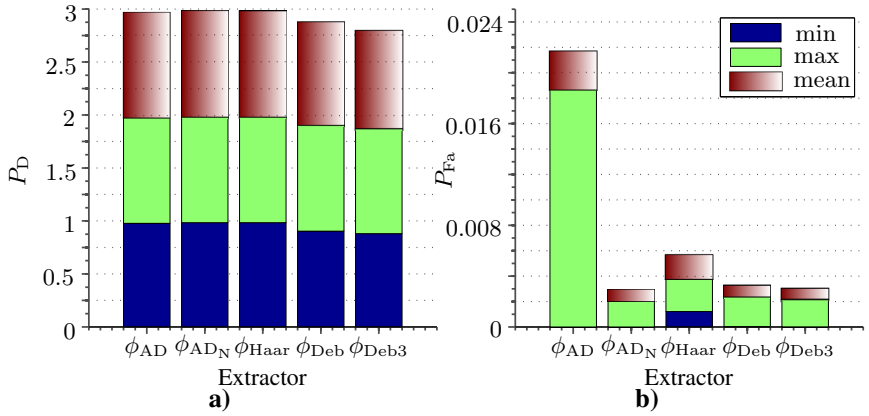


Fig. 3.28. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

achieved average P_{D} was lower than it was in square and rhombus SOMs. Hexagonal SOM P_{Fa} was similar to other 2 networks structures.

From these hexagonal SOM combinations most performant set can't be selected, because results were too various. In narrow-band and burst signal cases ϕ_{Haar} set was inefficient, however detection performance in frequency hopping signal record was good. Other extractors combinations were very inefficient in frequency hopping case, therefore these sets can't be selected also.

Self-training with endpoint detection. In this stage 3 SOMs (square, rhombus and hexagonal) will be tested with different than general rule

Table 3.4. Mean P_{D} and P_{Fa} for different hexagonal SOM sizes

Network size	9	16	20	25
P_{D}	0.9176	0.9686	0.9525	0.9777
P_{Fa}	0.0239	0.0151	0.0135	0.012
Network size	30	36	40	45
P_{D}	0.9937	0.9672	0.993	0.9859
P_{Fa}	0.0158	0.0132	0.0151	0.0144

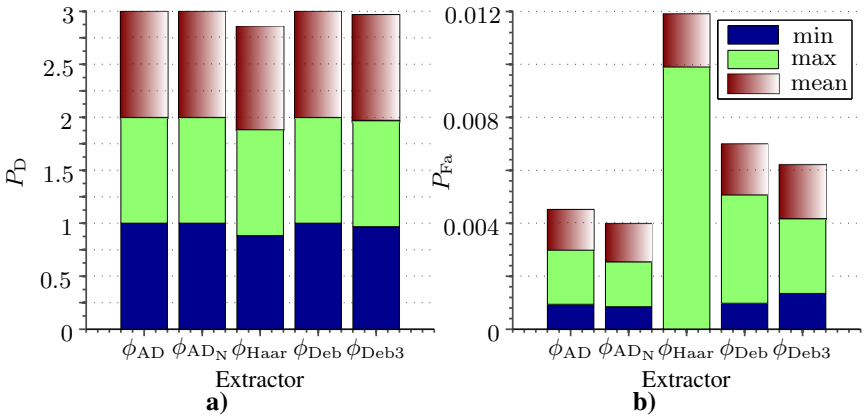


Fig. 3.29. Square SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

self-training iteration limiting procedure. Self-training process will be suspended when endpoint is reached. This endpoint detection was discussed in Section 2.3.1. Maximum quantity of possible self-training iterations in fallowing experiments will be limited by general rule. Therefore if self-training is not detected until max. iterations, which are defined by general rule, are reached, self-training will be suspended.

Square neighborhood function. For narrow-band signal least self-training iterations 512 were used by ϕ_{Deb} combination. Size of used SOM was 16 neurons. In comparison with general rule this self-training duration was $\sim$ 16 times shorter in endpoint detection case. Averagely least iterations were used by ϕ_{Deb3} sets – 2293.

Most accurate detection results were achieved by $\phi_{\mathrm{AD_N}}$ set. For this combination mean P_{D} was equal to 1 with 0.014 false alarm ratio. This result was better than in neural network or in general rule based self-training cases. To self-train $\phi_{\mathrm{AD_N}}$ combination averagely were used most iterations, although duration was more than 3 times lower than in general rule based self-training.

All combinations to self-train for burst signal detection averagely used 2000–2600 iterations. And these duration's are 6.5–5.5 shorter than in gen-

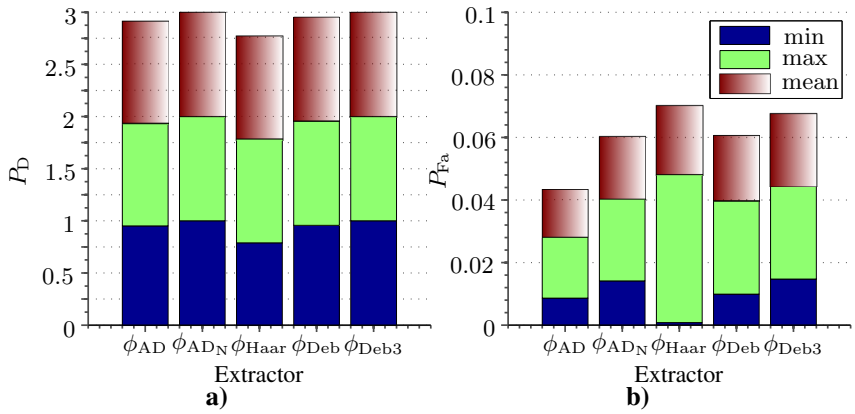


Fig. 3.30. Square SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

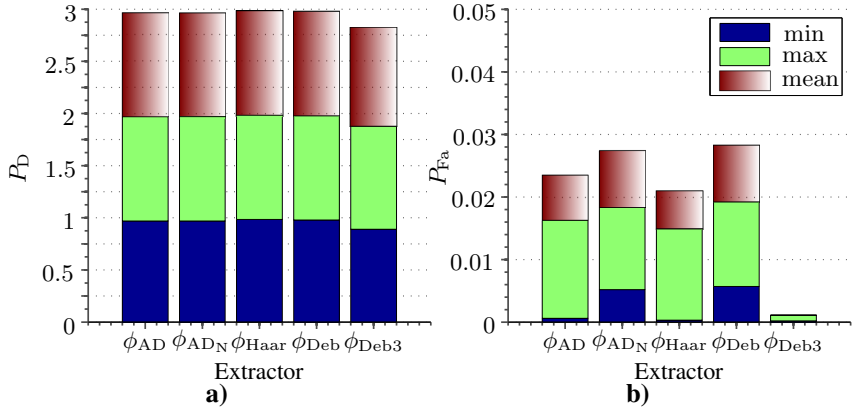


Fig. 3.31. Square SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

eral rule based self-training. Averagely least iterations – 2049 were used by ϕ_{Haar} set.

Full detection of burst signal was achieved by $\phi_{\mathrm{AD_N}}$ and ϕ_{Deb3} combinations (Fig. 3.30). Lowest P_{Fa} was in ϕ_{AD} set, although P_{D} was significantly lower than in the best results. Second lowest false alarm ratio 0.02 was in $\phi_{\mathrm{AD_N}}$. Therefore $\phi_{\mathrm{AD_N}}$ set outperformed ϕ_{Deb3} combination. This self-training approach in burst signal detection outperformed neural network and general rule based self-training.

To self-train square SOM with endpoint detection were used significantly more iterations than in burst or narrow-band signal cases. In some cases SOM reached maximum self-training iterations quantity and self-training process was suspended by general rule criteria. Averagely least iterations – 3918 were

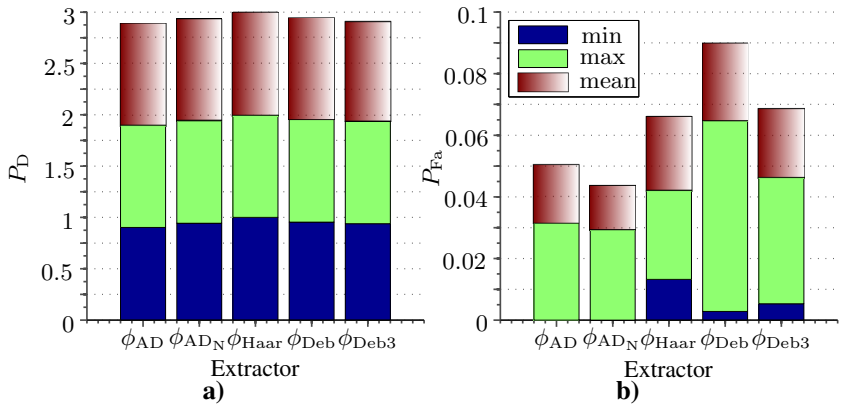


Fig. 3.32. Square SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Table 3.5. Mean P_{D} and P_{Fa} for different square SOM sizes

Network size	9	16	20	25
P_{D}	0.9581	0.9856	0.9888	0.993
P_{Fa}	0.0102	0.0138	0.0167	0.0239
Network size	30	36	40	45
P_{D}	0.9942	0.9955	0.9941	0.9931
P_{Fa}	0.0185	0.0156	0.014	0.0171

used by ϕ_{AD} set. Overall for this signal type SOM self-training duration was from 1.5 to 3 shorter than in general rule case.

Lowest mean P_{Fa} was in ϕ_{Deb3} combination. Practically it was 0 (Fig. 3.31), although mean P_{D} was lowest too. Best detection ratio with low P_{Fa} 0.006 was reached by ϕ_{Haar} and SOM set. ϕ_{Haar} and this SOM combination outperformed both neural and general rule based networks.

Self-training process duration for frequency hopping signals was significantly shorter in ϕ_{AD}, $\phi_{\mathrm{AD_N}}$ and ϕ_{Haar} combinations than in ϕ_{Deb} based sets. Averagely first three combinations used $\simeq$ 1800 iterations and it is $\sim$ 6.5 times less than general rule based self-training. ϕ_{Deb} based combinations were just 1.5–2.5 times faster.

Most detected emissions of frequency hopping signal 0.999, with respectably low P_{Fa} 0.019 were achieved with ϕ_{Haar} (Fig. 3.32). In this signal type neural and general rule based networks were outperformed by endpoint detection self-training algorithm too.

Lowest P_{Fa} 0.0102 was reached with 9 neurons SOM size (Table 3.5), yet average P_{D} with this size was just 0.9581. Highest detection ratio 0.9955 with low P_{Fa} 0.0156 reached with 36 neurons size.

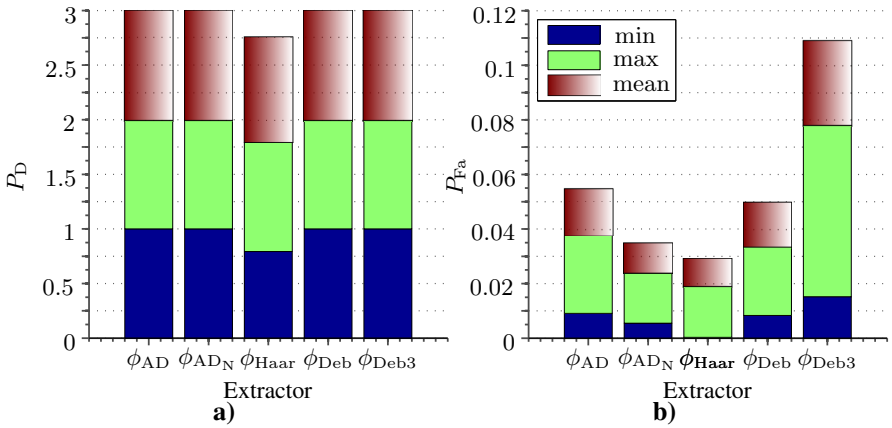


Fig. 3.33. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

For all signal types neural and general rule based networks were outperformed by this square SOM. Self-training algorithm based on endpoint detection helped to save up to 6.5 times self-train duration in comparing with general rule based self-training. All extractors and SOM combinations used more iterations to self-train for wide-band signal detection. Overall with this square SOM best performance was achieved with $\phi_{\mathrm{AD_N}}$ combination.

Rhombus neighborhood function. Least iterations – 1611 were used to self-train rhombus and ϕ_{Deb} RF emissions extraction and detection combination. This set self-training process duration was ~ 9 times lower than general rule based algorithm. Most iterations – $1.8 \cdot 10^4$ were used to self-train $\phi_{\mathrm{AD_N}}$ and 36 neurons SOM combination. In this case self-training was suspended not by endpoint detection, however self-training process reached self-training iteration boundary. Averagely least iterations – 1611 were used by ϕ_{Deb} set.

Best detection P_{D} 1 and lowest false alarm P_{Fa} 0.01 ratios was reached by $\phi_{\mathrm{AD_N}}$ set (Fig. 3.33). This SOM and extractor combination outperformed all previous detectors results, which were achieved for narrow-band signals. Yet for $\phi_{\mathrm{AD_N}}$ sets were used 3 times more self-training iterations than in ϕ_{Deb} case.

In burst signal case for self-training process overall were used just 1641 iterations, and it's 8.4 times less than in general rule based self-training. Absolute max. iterations 4556 for this signal were used in 45 rhombus SOM self-training.

Best accuracy in burst signal case was achieved with ϕ_{Deb} set (Fig. 3.34). In this case mean P_{D} was 1 and P_{Fa} 0.031. These results outperformed results, which was achieved with neural network, however in square SOM (endpoint case) performance was better.

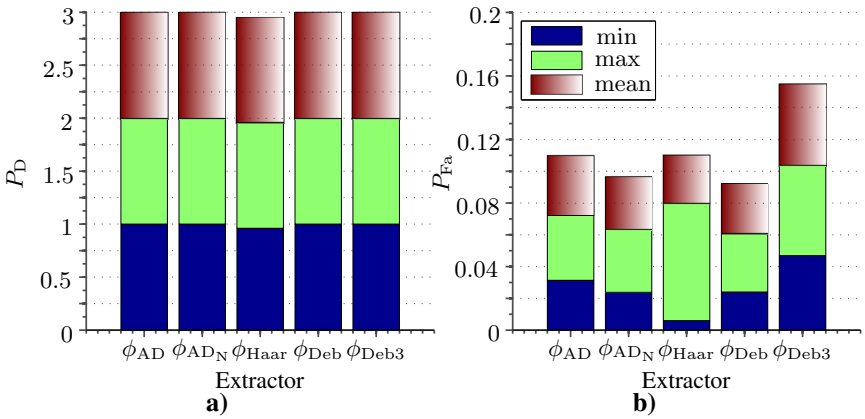


Fig. 3.34. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

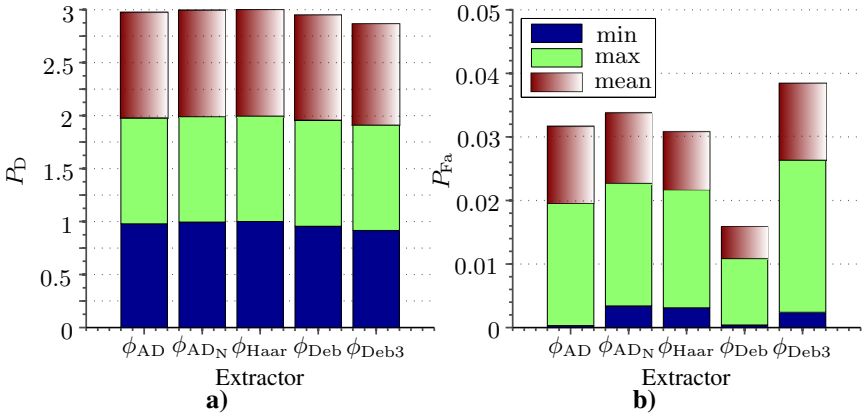


Fig. 3.35. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

As in square SOM (same self-training method), in rhombus network self-training process was longer in wide-band signal than in other signal types. Averagely least self-training iterations – 4956 were used in ϕ_{Deb} and SOM.

Lowest mean P_{Fa} for wide-band signal was achieved with ϕ_{Deb} set (Fig. 3.35). However higher detection ratio 1 with slightly higher P_{Fa} 0.09 was reached by ϕ_{Haar} combination. ϕ_{Haar} and rhombus combination performance in wide-band signal detection was better than in neural and square networks cases.

ϕ_{AD} and $\phi_{\mathrm{AD_N}}$ combinations used significantly less self-training iterations than others H, ϕ_{Deb} and ϕ_{Deb3} sets. Averagely best self-training performance 1428 (iteration economy) showed ϕ_{AD} and SOM combination.

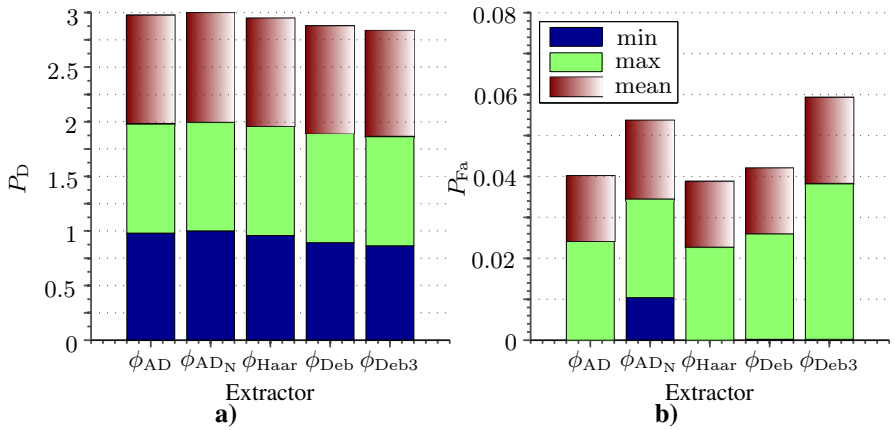


Fig. 3.36. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Table 3.6. Mean P_{D} and P_{Fa} for different rhombus SOM sizes

Network size	9	16	20	25
P_{D}	0.9701	0.9931	0.9933	0.9905
P_{Fa}	0.0159	0.0193	0.0177	0.0204
Network size	30	36	40	45
P_{D}	0.9983	0.998	0.9981	0.9938
P_{Fa}	0.0248	0.024	0.0199	0.0197

Highest detection performance was achieved by $\phi_{\mathrm{AD_N}}$ and SOM combination (Fig. 3.36). Mean P_{D} for this set was 1. False alarm ratio for $\phi_{\mathrm{AD_N}}$ combination was not lowest, though it was slightly higher than in ϕ_{Haar} case (by 0.003). $\phi_{\mathrm{AD_N}}$ and rhombus SOM combination outperformed not just neural network and previous SOMs results. It exceeded even ROC characteristic.

In rhombus SOM based on endpoint detection best accuracy was achieved by 30 neurons size network (Table 3.6). Performance of 30 rhombus SOMs was better than results achieved by previous neural and SOM networks.

In summary, best results for all signal types were reached with rhombus and $\phi_{\mathrm{AD_N}}$ combination. In most cases this combination outperformed results, which were achieved by previous networks (neural, SOM based on general rule self-training). Rhombus network, as square, network used more self-training iterations for wide-band signal self-training, however overall self-training duration was shorter than it was in general rule based networks.

Hexagonal neighborhood function. Shortest average self-training duration – 1341 were used by ϕ_{Haar} and SOM combination. This duration was 10

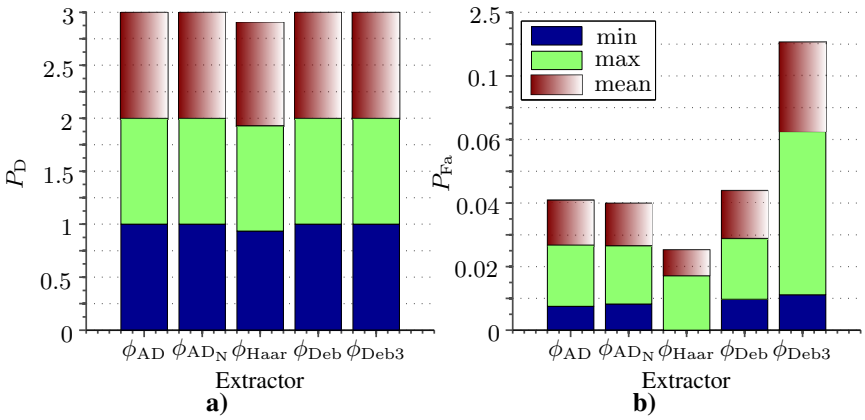


Fig. 3.37. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

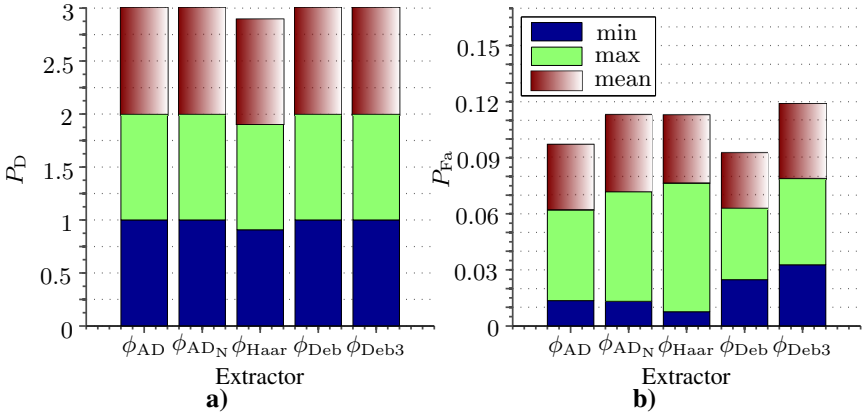


Fig. 3.38. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

times shorter than in general rule based self-training algorithms. Averagely longest self-training duration was used by $\phi_{\mathrm{AD_N}}$ and SOM set.

Better detection performance for narrow-band signals was achieved with $\phi_{\mathrm{AD_N}}$ set (Fig. 3.33). In ϕ_{Haar} average P_{D} was just 0.972, when in $\phi_{\mathrm{AD_N}}$ case detection ratio was 1. For $\phi_{\mathrm{AD_N}}$ average false alarm ratio was 0.013. P_{D} and P_{Fa} of $\phi_{\mathrm{AD_N}}$ combination was significantly better than neural network.

In burst signals detection with ϕ_{Deb} transform based extractor and SOM combinations were used significantly less self-training iterations – 1527 than in other 3 sets. In self-training process ϕ_{Deb} set averagely used 9 times less self-training iterations than general rule based based algorithm.

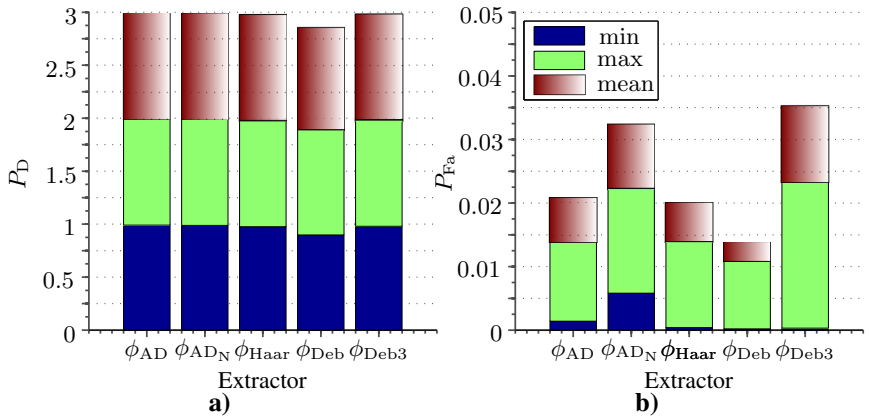


Fig. 3.39. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

Best detection performance for burst signals was achieved by ϕ_{Deb} and SOM set too (Fig. 3.34). Mean P_{D} and P_{Fa} was 1 and 0.029 accordantly. This result outperformed neural network, however results in general rule based SOMs self-training algorithms were better.

As in square and rhombus SOMs, in hexagonal network were used more self-training iterations for wide-band signals than for others signal types. Shortest average self-training process duration was in ϕ_{Deb} combination – 6360 iterations. Longest ϕ_{Deb3} combination – 9577 iterations.

For wide-band signal type best detection performance was in ϕ_{AD} and SOM set (Fig. 3.35). Mean P_{D} was 0.999 with 0.008 false alarm ratio. This ϕ_{AD} and hexagonal SOM combination outperformed neural and general rule based networks.

For frequency hopping signals least average self-training iterations – 1353 were used by ϕ_{Haar} set. This duration is 10 times shorter than in analogical SOM, which self-training was based on general rule. Longest average self-training duration – 4191 was with ϕ_{Deb} set.

Best detection performance for frequency hopping signal was reached by $\phi_{\mathrm{AD_N}}$ set (Fig. 3.36). With this combination full emission detection was achieved, when mean P_{Fa} was 0.019. Lowest mean P_{Fa} 0.013 was in ϕ_{Deb} set. However for ϕ_{Deb} case detection ratio was by 0.017 lower than it was in $\phi_{\mathrm{AD_N}}$ set. $\phi_{\mathrm{AD_N}}$ and hexagonal SOM performance was better than neural, general rule based networks and ROC characteristic.

Best network size for this hexagonal network , as in most SOMs types, was 30 neurons. For this SOM size average P_{D} and P_{Fa} was 0.9981 and 0.0198 accordingly.

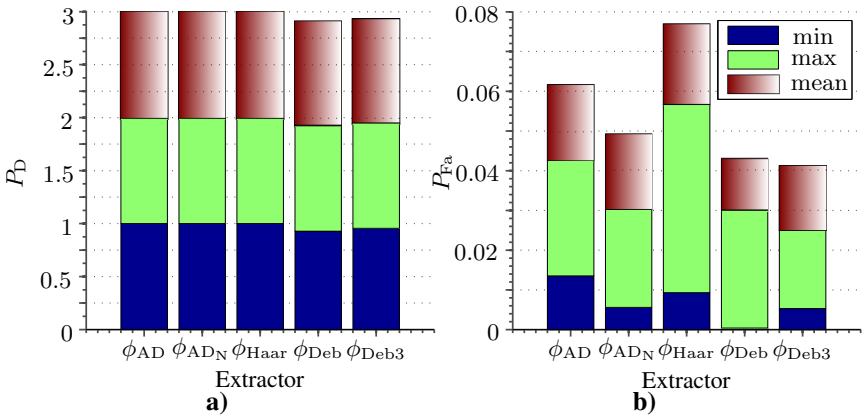


Fig. 3.40. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Table 3.7. Mean P_{D} and P_{Fa} for different hexagonal SOM sizes

Network size	9	16	20	25
P_{D}	0.9892	0.9867	0.9944	0.995
P_{Fa}	0.0181	0.0146	0.0185	0.0205
Network size	30	36	40	45
P_{D}	0.9981	0.9968	0.9972	0.99
P_{Fa}	0.0198	0.0202	0.0191	0.0217

Overall best performance was shown by $\phi_{\mathrm{AD_N}}$ and SOM combination. In most cases this hexagonal SOM outperformed neural and on general rule based networks. In burst signal case this network performance was slightly worse than in best case of general rule based self-training. Self-training duration of these networks group (hexagonal based on endpoint detection) was up to 10 times lower than general self-training.

Self-training with assistant. In this stage square, rhombus, and hexagonal SOMs will be tested with different self-training processes. Assistant, which was introduced in Section 2.3.2, will be involved in SOM self-training. Assistant is responsible for self-training process control. As in endpoint detection, in assistant case quantity of maximum self-training iterations is defined by general rule.

Square neighborhood function. With assistant least averagely self-training iterations – 1166 for narrow-band signal used by ϕ_{Haar} and square SOM set. Longest self-training duration – 2834 was with ϕ_{Deb}. This result is $\sim$ 12 times shorter than in general rule based self-training.

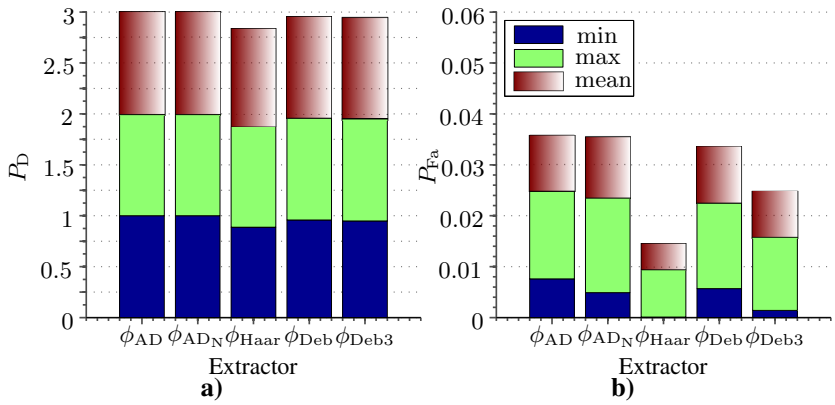


Fig. 3.41. Square SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

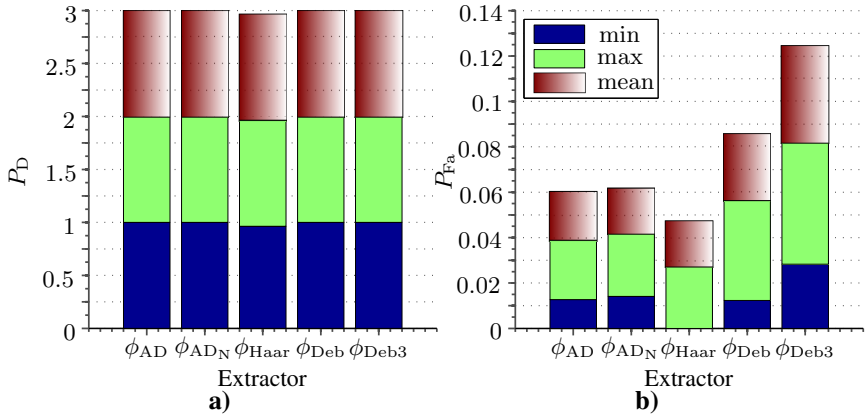


Fig. 3.42. Square SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

Best detection accuracy 1 with low mean P_{Fa} 0.011 for narrow-band signals was achieved by ϕ_{AD} and square SOM combination (Fig. 3.41). This result outperformed best results for this signal type of neural and previous SOMs (based on general rule or endpoint det.).

Significantly shorter self-training duration was in burst signal type case. Least self-training iterations – 697 were used by ϕ_{Deb} case. Averagely for all sets (ϕ_{AD}, $\phi_{\mathrm{AD_N}}$, ϕ_{Haar}, ϕ_{Deb} and ϕ_{Deb3}) self-training duration was just – 1010 iterations. This average duration was 13 times shorter than in general rule based self-training.

In burst signal type all extractor combinations except ϕ_{Haar} mean P_{D} reached 1 (Fig. 3.42). However lowest false alarm ratio was in $\phi_{\mathrm{AD_N}}$ case 0.02. $\phi_{\mathrm{AD_N}}$ performance was better than neural and general rule based networks.

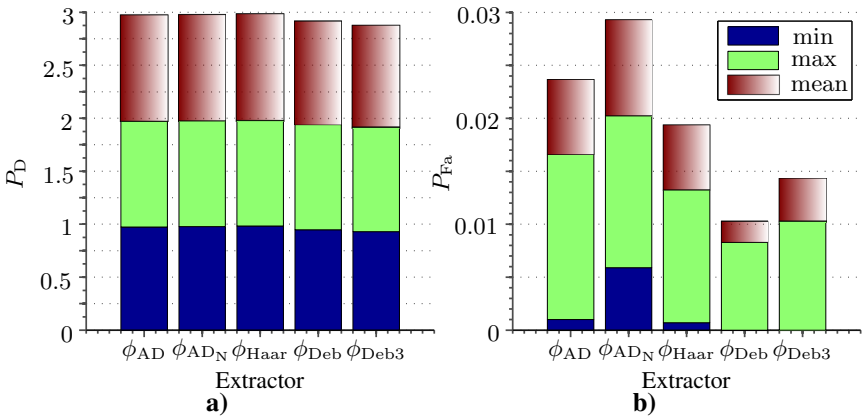


Fig. 3.43. Square SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

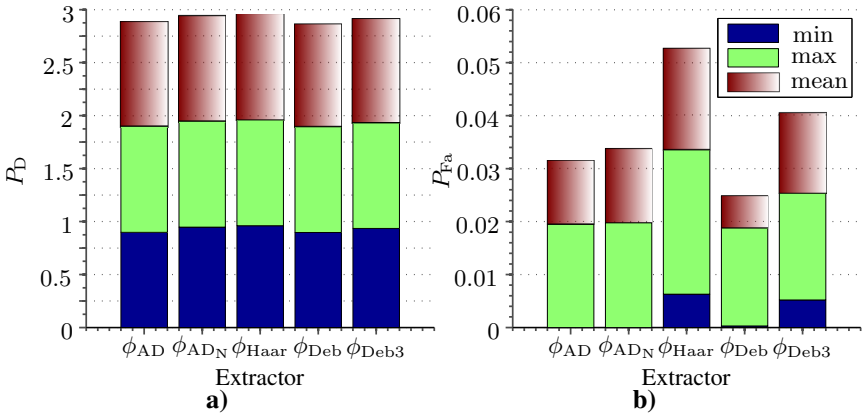


Fig. 3.44. Square SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

As in endpoint detection, in assisted square SOM case self-training process duration for wide-band signal was significantly higher than in other signal types. Least self-training iterations – 5534 were used by $\phi_{\mathrm{AD_N}}$ set. And this duration was ~ 2.5 times shorter than in general rule based self-training.

Best detection ratio with low P_{Fa} was achieved by ϕ_{Haar} and SOM combination (Fig. 3.43). For this case mean P_{D} was 0.996 and mean P_{Fa} 0.006. ϕ_{Haar} and assisted square SOM outperformed neural network and SOMs based on general rule results.

In frequency hopping assisted square SOM averagely for all extractors used – 1537 iterations. And this average duration is 9 times shorter than in general rule based SOMs.

Table 3.8. Mean P_{D} and P_{Fa} for different square SOM sizes

Network size	9	16	20	25
P_{D}	0.9855	0.984	0.9794	0.9955
P_{Fa}	0.0117	0.0124	0.0131	0.0146
Network size	30	36	40	45
P_{D}	0.9945	0.9955	0.9872	0.9879
P_{Fa}	0.0144	0.0139	0.0141	0.0152

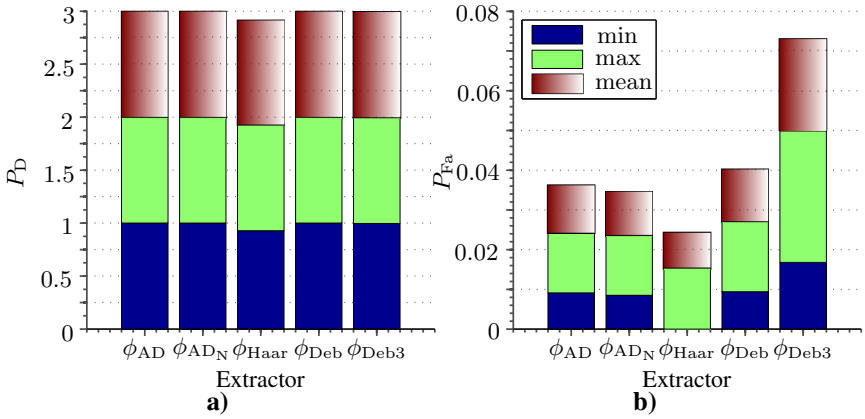


Fig. 3.45. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

Best detection ratio for frequency hoping was achieved by ϕ_{Haar} and SOM set (Fig. 3.44). By ϕ_{Haar} reached mean P_{D} was 0.995 with mean P_{Fa} 0.019. This set performance is better than neural network, however detection ratio is slightly lower than SOMs based on general rule self-training.

Overall (for all signal types) best results were achieved with 36 SOM size (Table 3.8). For these assisted square SOM size average P_{D} was 0.9955 and average P_{Fa} 0.0139.

In summary, best detection accuracy was in $\phi_{\mathrm{AD_N}}$ SOM case. Although its results wasn't the best in all signal types, however in most cases it performed well. Assisted SOM in all signal types outperformed neural network and in most types general rule based SOMs. Self-training duration of assisted algorithm was up to 13 times shorter than general rule based SOMs.

Rhombus neighborhood function. Assisted rhombus SOM with ϕ_{Haar} used least iterations – 844 for narrow-band signals. This self-training duration was even 16 times shorter than in general rule case based self-training. Longest average self-training duration was with $\phi_{\mathrm{AD_N}}$ extractor.

Best accuracy for narrow-band signal type was achieved by $\phi_{\mathrm{AD_N}}$ combination (Fig. 3.45). In $\phi_{\mathrm{AD_N}}$ case achieved mean P_{D} 1 with significantly

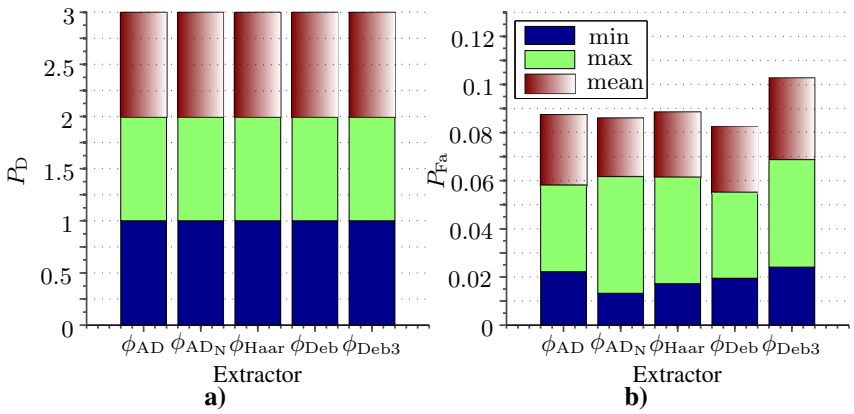


Fig. 3.46. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

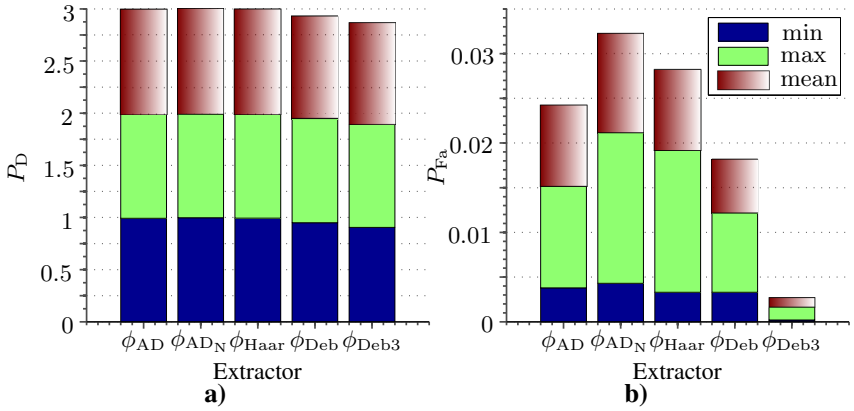


Fig. 3.47. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

low P_{Fa} 0.011. $\phi_{\mathrm{AD_N}}$ assisted rhombus SOM result repeated performance for narrow-band signal, which was achieved by assisted square SOM.

Overall (including all extractors) in burst signal case average self-training duration was 767 iterations. This average duration is even 18 times shorter than in general rule based self-training.

All emissions of burst signal were detected with all extractors (Fig. 3.46). Lowest P_{Fa} 0.024 was in ϕ_{Deb} case. ϕ_{Deb} result outperforms neural network, yet P_{Fa} was slightly higher than in general rule based SOMs.

Self-training duration for wide-band signals was significantly higher than it was in other signal types. However with assisted rhombus average duration was by $\sim$ 2000 iterations lower than in SOMs based on endpoint detection. Shortest self-training duration –3381 iterations was with $\phi_{\mathrm{AD_N}}$.

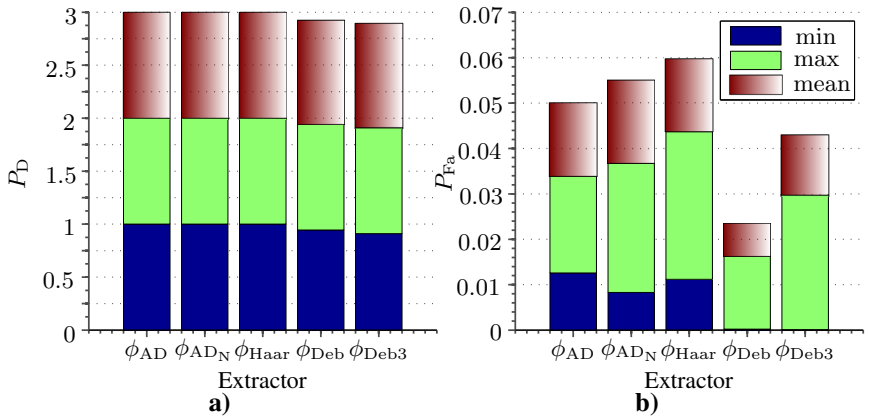


Fig. 3.48. Rhombus SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Table 3.9. Mean P_{D} and P_{Fa} for different rhombus SOM sizes

Network size	9	16	20	25
P_{D}	0.9888	0.9904	0.9913	0.9959
P_{Fa}	0.0139	0.0134	0.0147	0.0152
Network size	30	36	40	45
P_{D}	0.9974	0.9971	0.9992	0.9938
P_{Fa}	0.0156	0.0178	0.0182	0.0173

ϕ_{AD} and SOM set showed the best performance in wide-band signal type (Fig. 3.47). For ϕ_{AD} mean P_{D} was 0.999 and P_{Fa} 0.0009. This result was similar to neural network achieved results.

ϕ_{AD} and assisted rhombus SOM combination used least self-training iterations 586 for frequency hoping signal. This duration was significantly 23 times shorter than general rule based self-training. Overall for all extractors average self-training duration was just 1516 iterations.

For frequency hopping signals ϕ_{Haar}, ϕ_{AD} and $\phi_{\mathrm{AD_N}}$ sets was achieved same detection ratio 1 (Fig. 3.48). However false alarm ratio 0.016 was lower in ϕ_{AD} combination. Overall this ϕ_{AD} and assisted rhombus SOM combination reached best detection performance for frequency hopping signals. It outperforms neural network, SOMs based on general rule and assisted square SOM results.

Best detection performance from assisted rhombus SOMs was achieved by 40 neurons size (Table 3.9). Average P_{D} was 0.9992 and P_{Fa} 0.0182. This performance was one of the best achieved by SOMs.

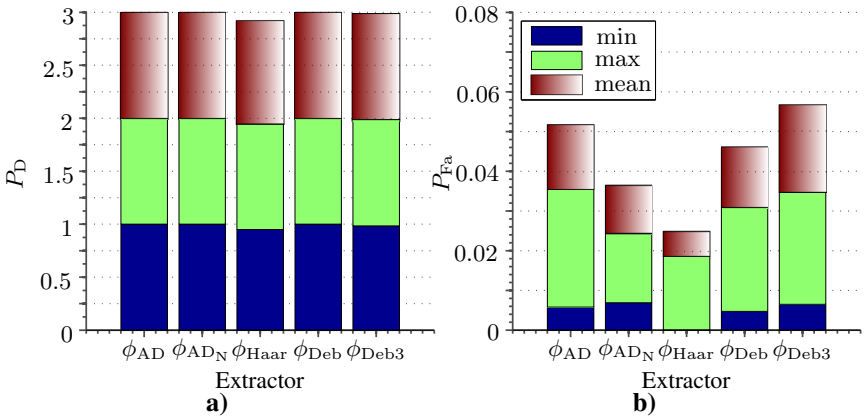


Fig. 3.49. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

Overall, for all signal types combination with ϕ_{AD} showed best performance. This combination outperformed all previous networks in frequency hopping signal detection. Assisted rhombus SOM self-training process duration was up to 23 times shorter than SOMs based on general rule.

Hexagonal neighborhood function. Least self-training iterations – 672 for narrow-band signal were used to self-train ϕ_{Deb3} and SOM set. Overall (for all extractors) average self-training process duration was – 1453.

Best accuracy for narrow-band signal was achieved with $\phi_{\mathrm{AD_N}}$ and SOM set (Fig. 3.49). $\phi_{\mathrm{AD_N}}$ achieved detection ratio was 1 and P_{Fa} 0.012. False alarm ratio achieved by this assisted hexagonal SOM was slightly higher (by 0.001) than it was with assisted rhombus.

As it was with previous two assisted SOMs (square, rhombus), self-training duration for burst signal was significantly shorter than for other signal types. Overall average self-training duration was just – 954 iterations. Most self-training iterations – 1053 were used for ϕ_{Deb} and SOM combination.

Best performance for burst signals was achieved by $\phi_{\mathrm{AD_N}}$ and SOM set (Fig. 3.50). Full interception was achieved with 0.033 average false alarm ratio. This result outperformed neural network achieved performance, however in general rule based self-training P_{Fa} was slightly lower.

With assisted hexagonal SOM self-training process duration for wide-band signal was by $\sim$ 2500 iterations shorter than in endpoint detection SOMs. For all extractors average self-training duration was 4538 iterations.

Best performance for wide-band signal detection was achieved by $\phi_{\mathrm{AD_N}}$ an SOM set (Fig. 3.51). $\phi_{\mathrm{AD_N}}$ mean detection ratio was 0.995 with 0.009 P_{Fa}. This $\phi_{\mathrm{AD_N}}$ performance was not enough to outperform neural network results.

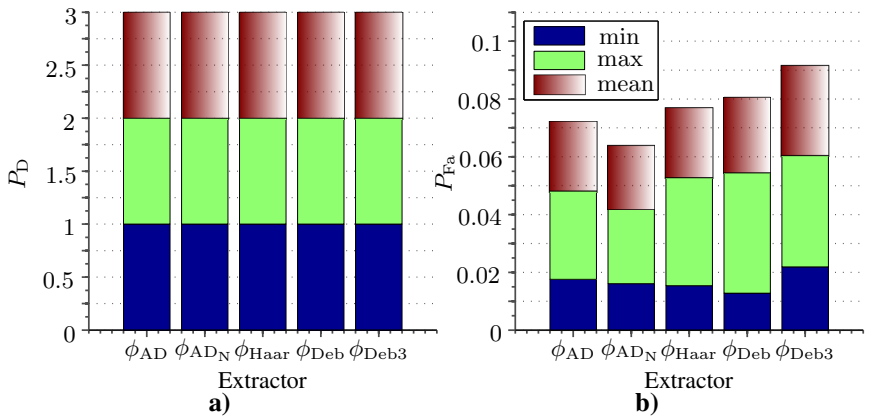


Fig. 3.50. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for burst signals

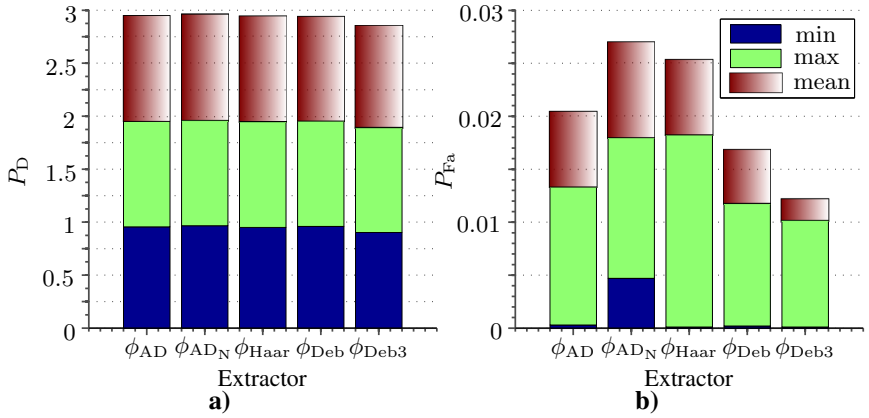


Fig. 3.51. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for wide-band signal

In frequency hopping signal type significantly lower self-training duration – 711 was in ϕ_{AD}, $\phi_{\mathrm{AD_N}}$ and ϕ_{Haar} cases. Self-training duration of these three sets was 16–22 times shorter than in general rule based self-training.

All frequency hopping signals emissions were detected by $\phi_{\mathrm{AD_N}}$ combination with 0.017 false alarm ratio (Fig. 3.52). Average P_{Fa} of $\phi_{\mathrm{AD_N}}$ and assisted hexagonal SOM was by 0.001 higher than it was in assisted rhombus network.

Best detection performance in assisted hexagonal SOM was achieved with 45 network sizes (Table 3.10). Detection ratio for this size was 0.9975 and P_{Fa} 0.0174.

In summary, best detection performance for all signal types was achieved by $\phi_{\mathrm{AD_N}}$ and assisted hexagonal SOM combination. In all cases, except

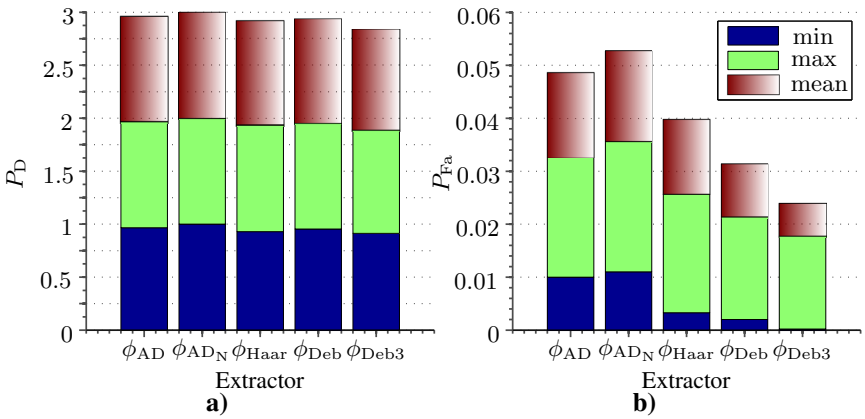


Fig. 3.52. Hexagonal SOM: a) P_{D}; b) P_{Fa} ratios for frequency hopping signals

Table 3.10. Mean P_{D} and P_{Fa} for different hexagonal SOM sizes

Network size	9	16	20	25
P_{D}	0.9851	0.9858	0.9884	0.9854
P_{Fa}	0.0129	0.0128	0.0158	0.0144
Network size	30	36	40	45
P_{D}	0.9955	0.9933	0.9918	0.9975
P_{Fa}	0.0152	0.0135	0.0142	0.0174

wide-band signal type, this SOM outperformed neural network results. Self-training duration of assisted SOM was up to 22 times shorter than it was general rule based self-training.

SOM with inner weights. About this SOM with inner weights was discussed in Sections 2.3.3. This SOM self-training process suspension was made by endpoint detection procedure, with which SOM preparedness level was determined.

SOM with inner weights for narrow-band signal averagely used just 62 iterations. This network self-training duration was significantly lower than neural networks and all previous SOMs. In comparison, to self-train this SOM was used 222 times less iterations than general rule based self-training and $\sim$ 24 times less than assisted SOMs.

Most emissions of narrow-band signal were detected by SOM and ϕ_{Deb} combination (Fig. 3.53). For this set mean P_{D} was 0.983 and P_{Fa} just 0.004. This detection ratio was not enough to outperform neural network achieved P_{D}, however SOM with inner weights achieved false alarm ratio was significantly lower.

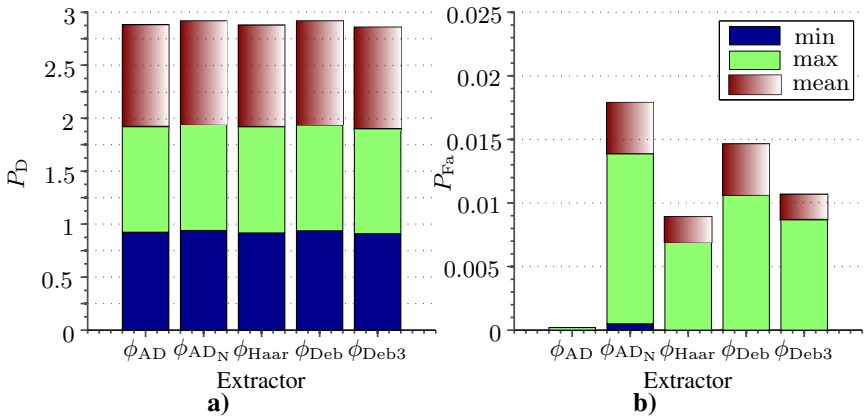


Fig. 3.53. SOM with inner weights: a) P_{D}; b) P_{Fa} ratios for narrow-band signals

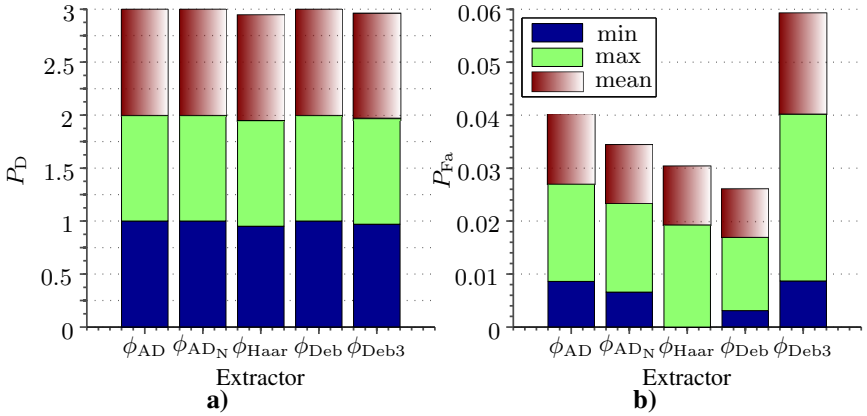


Fig. 3.54. SOM with inner weights: a) P_{D}; b) P_{Fa} ratios for burst signals

For burst signal SOM with inner weights self-training duration was significantly lower than previous SOMs too. Overall average self-training process duration was 56 iterations and it was 246 times shorter than general rule based self-training.

ϕ_{Deb} and SOM combination achieved best detection performance for burst signals (Fig. 3.54). All emissions were detected by ϕ_{Deb} set with low P_{Fa} 0.009. This ϕ_{Deb} and SOM with inner weights combination for burst signals outperformed all previous best networks results.

Self-training duration hasn't significantly increased for wide-band signal, as it was in endpoint detection and assisted SOMs. Overall average

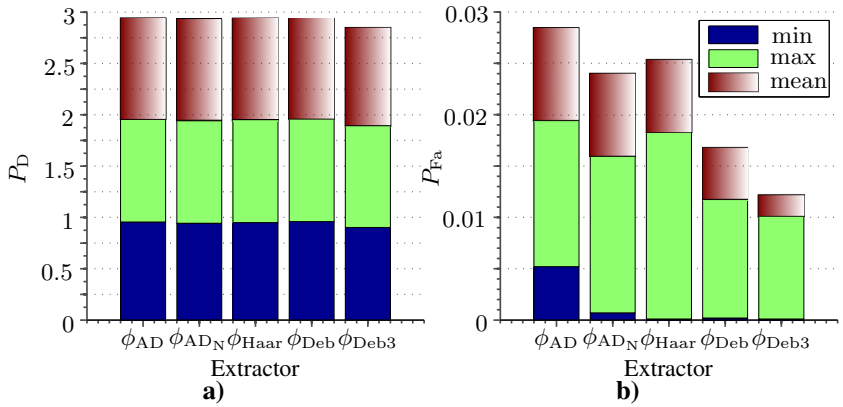


Fig. 3.55. SOM with inner weights: a) P_{D}; b) P_{Fa} ratios for wide-band signal

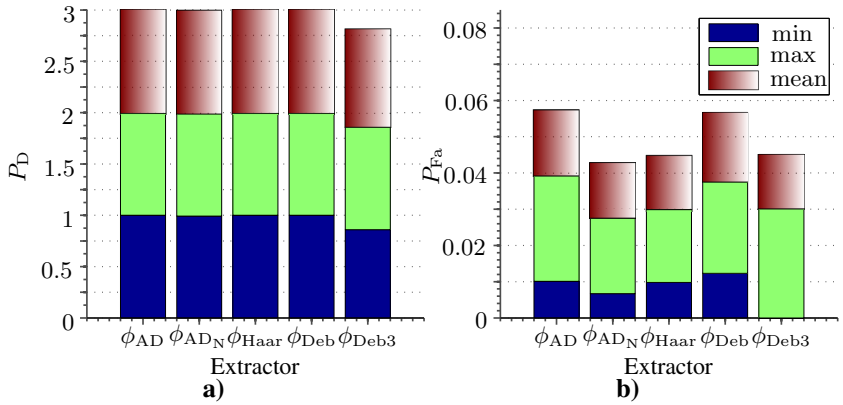


Fig. 3.56. SOM with inner weights: a) P_{D}; b) P_{Fa} ratios for frequency hopping signal

self-training duration (53 iterations) was even lower than it was in narrow-band and burst signals. SOM with inner weights self-training process was 260 times shorter than in general rule based self-training.

Best detection performance for wide-band signal was reached by ϕ_{Haar} and SOM combination (Fig. 3.55). Detection ratio for ϕ_{Haar} set was 0.993 and mean P_{Fa} 0.007. This result was slightly less performant than result achieved with neural network.

As in previous signal types, in frequency hopping SOM with inner weights used small quantity of self-training iterations. Averagely self-training duration was 246 times shorter than in general rule based SOM self-training.

Table 3.11. Mean P_{D} and P_{Fa} for different SOM with inner weights sizes

Network size	9	17	21	25
P_{D}	0.9739	0.9802	0.9883	0.9792
P_{Fa}	0.009	0.0092	0.009	0.0087
Network size	29	37	41	45
P_{D}	0.9844	0.9948	0.9823	0.9912
P_{Fa}	0.0078	0.0125	0.0081	0.0106

In frequency hopping signal best performance was shown by ϕ_{Haar} transform combination (Fig. 3.56). ϕ_{Haar}set achieved full emission interception with low 0.015 P_{Fa} This result slightly outperformed result, which was achieved by assisted rhombus. Therefore overall from all networks best result for frequency hopping signal was achieved by ϕ_{Haar} and SOM with inner weights combination.

Best performance from SOM with inner weights was reached by 36 network size (Table 3.11). Detection ratio for this size was 0.9948 and false alarm ratio 0.0125.

Overall for all signal types best performance was reached by ϕ_{Deb} combination. SOM with inner weights performance in burst and frequency hopping signal types was best from all networks, which were tested in these experiments. This SOM type self-training duration was remarkably shorter than in based on general rule, endpoint detection or assisted self-training algorithms.

Summarized results of offline experiments

Neural network:

- average training duration – 122835 iterations;
- most effective network size – 30 neurons;
- average P_{D} – 0.9863 and P_{Fa} – 0.0272;
- best accuracy was achieved by ϕ_{AD} set.

SOMs based on general rule self-training:

- average self-training duration – 13812 iterations;
- most effective network size – 30 neurons;
- average P_{D} – 0.9799 and P_{Fa} – 0.0154;
- best accuracy was achieved by $\phi_{\mathrm{AD_N}}$ and rhombus SOM combination.

SOMs based on break point detection:

- average self-training duration – 3787 iterations;
- most effective network size – 30 neurons;
- average P_{D} – 0.991 and P_{Fa} – 0.0185;
- best accuracy was achieved by $\phi_{\mathrm{AD_N}}$ and hexagonal SOM combination.

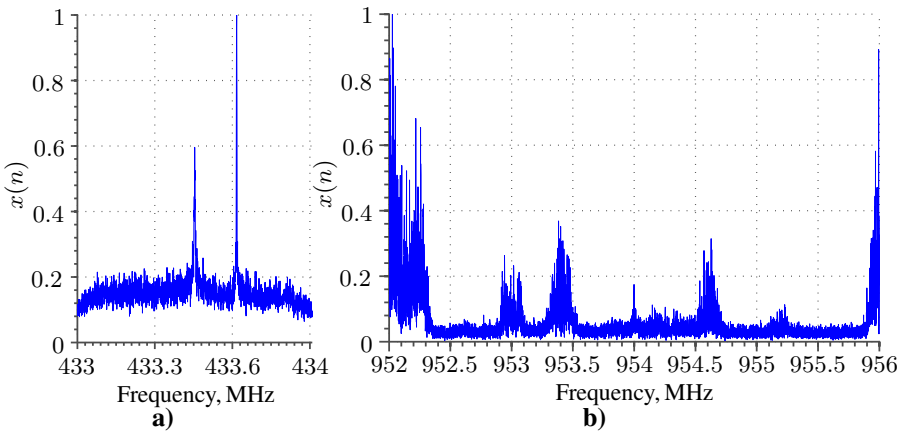


Fig. 3.57. RF spectrum environments examples: a) $F_{\mathrm{C}} = 433.5\,\mathrm{MHz}$, BW$= 1\,\mathrm{MHz}$; b) $F_{\mathrm{C}} = 954\,\mathrm{MHz}$, BW$= 4\,\mathrm{MHz}$

Assisted SOMs:

- average self-training duration – 2467 iterations;
- most effective network size – 36 neurons;
- average P_{D} – 0.9911 and P_{Fa} – 0.0146;
- best accuracy was achieved by $\phi_{\mathrm{AD_N}}$ and rhombus SOM combination.

SOMs with inner weights:

- average self-training duration – 57 iterations;
- most effective network size – 37 neurons;
- average P_{D} – 0.9843 and P_{Fa} – 0.0094;
- best accuracy was achieved by ϕ_{Deb} set.

For online experiments will be used these SOMs and extractors combinations: general rule based 30 rhombus SOM and $\phi_{\mathrm{AD_N}}$, 30 hexagonal SOM based on endpoint detection and $\phi_{\mathrm{AD_N}}$, assisted 36 rhombus SOM and $\phi_{\mathrm{AD_N}}$, 37 SOM with inner weights and ϕ_{Deb}.

3.4. Experiments of the Spectrum Sensors on a Real-time Data

Online experiments were made in two RF bands. First band is in the unlicensed range 433 MHz, another is in the licensed range 954 MHz. In unlicensed band most dominant signal type are burst signals (Fig. 3.57). In 953 MHz range mostly are signals with changing carrier frequency – frequency hopping. Both signal types are described more detailed in Section 3.1.

Experiments were made in 3 locations in order to check RF spectrum occupancy dependency on population density:

- city – Kaunas, Pasiles 52. Population density is high;
- countryside – Kaunas, Rugstyniu 26. Population density is average;
- village – Zeimiai, Blauzdziu 22. Population density is low.

Duration of each RF band experiments was 24 hours. Therefore overall experiments in 3 locations took 144 hours.

For experiments was used wide-band SDR platform see Fig. 3.6. Core of this platform are Spartan6 LX75 FPGA and AD9361 single chip receiver. RF samples digitized by receiver were passed to FPGA, where further processing was made. In FPGA RF IQ data was processed by FFT block (Fig. 3.58). 2 FFT sizes were used in these experiments:

- 4096 – for 433 MHz BW = 1 MHz;
- 16384 – for 954 MHz BW = 4 MHz.

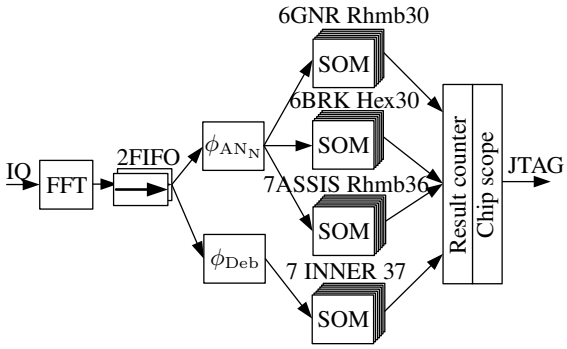


Fig. 3.58. Experimental radio signal processing flow graph

After transform RF spectrum samples were buffered into FIFO buffers. Buffers are needed for $\phi_{\mathrm{AD_N}}$ and ϕ_{Deb}. Processing latency of extractors is different than FFT, therefore FIFO is used for synchronization of different data rates. Buffers sizes are same as FFT transform length.

Calculated spectrum features are processed by SOMs:

- GNR Rhmb30 – On general rule based rhombus SOM, network size 30;
- BRK Hex30 – On endpoint detection based hexagonal SOM, network size 30;
- ASSIS Rhmb36 – On general rule based rhombus SOM, network size 36;
- Inner37 – SOM with inner weights, network size 37.

Table 3.12. FPGA resource utilization by different experimental structure implementation

Frequency band	Slc. reg.	LUTs	DSP slc.	B. RAM
433 MHz BW 1 MHz	8009	32893	115	83
954 MHz BW 4 MHz	9135	41713	124	112

Table 3.13. SOMs self-training durations in different experiments location

Intelligent detector	City	Countryside	Village
GNR Rhmb 30 $\phi_{\mathrm{AD_N}}$	24069305	24063450	24072006
BRK Hex 30 $\phi_{\mathrm{AD_N}}$	3045105	3560670	2127950
ASSIS Rhmb 36 $\phi_{\mathrm{AD_N}}$	861073	2682517	1903583
Inner 37 ϕ_{Deb}	351735	434496	392945

Several same SOMs type are dedicated for extractors. More networks were needed because latency of first two SOMs are ~ 5 times longer and of third and fourth SOMs ~ 6 times longer than extractors delay. SOMs self-training was made each hour in order to adapt networks in potentially changing environment. During self-training process networks don't produce detection output, therefore for this moment information about RF spectrum occupancy will be lost. Self-training duration as detection results are registered by result counter.

Result counter purpose is to combine detection results and pass them to Chip scope ip core. Counter counts detected emissions, free spaces and average self-training duration for each SOM group. These results are passed to PC for monitoring by chip scope through JTAG.

For signal which BW = 1 MHz was used 4096 points FFT, and for 4 MHz BW signal was used 16384 point transform. To implement experimental structure (Fig. 3.58) in Spartan6 LX75 FPGA for 433 MHz range were used 87% DSP slices, 48% 18 kb Block RAM and $\sim 50\%$ of logic (Table 3.12). For larger BW more resources were used: 94% DSP slices, 65% 18 kb Block RAM and $\sim 60\%$ of logic components. This resource utilization difference is influenced by FFT size. More multiplication elements, larger buffers and bigger quantity of logic are required for larger transform.

Average self-training duration's, which were measured by synchronization clock cycles, were measured for SOMs in all locations (Table 3.13). Most clock cycles for self-training were used by general rule based SOM. However in all locations for first network self-training duration it was most stable. This stability was achieved, because in all cases SOM was self-trained $1.8 \cdot 10^4$ self-training iterations. On endpoint detection based SOM self-training latency was ~ 8 times lower than in general rule based network. Longest average self-training duration was in countryside location. Assisted SOM self-training

Table 3.14. RF spectrum occupancy in 433 MHz range for 24 hours

Intelligent detector	City	Countryside	Village
GNR Rhmb 30 $\phi_{\mathrm{AD_N}}$	0.0957%	0.0062%	0.0863%
BRK Hex 30 $\phi_{\mathrm{AD_N}}$	0.0961%	0.0063%	0.0869%
ASSIS Rhmb 36 $\phi_{\mathrm{AD_N}}$	0.0962%	0.0063%	0.0870%
Inner 37 ϕ_{Deb}	0.0962%	0.0063%	0.0870%

Table 3.15. RF spectrum occupancy in 954 MHz range for 24 hours

Intelligent detector	City	Countryside	Village
GNR Rhmb 30 $\phi_{\mathrm{AD_N}}$	52.1001%	74.3509%	38.4697%
BRK Hex 30 $\phi_{\mathrm{AD_N}}$	52.1897%	74.4748%	38.4759%
ASSIS Rhmb 36 $\phi_{\mathrm{AD_N}}$	52.2001%	74.4798%	38.4801%
Inner 37 ϕ_{Deb}	52.2308%	74.5108%	38.4824%

process took 13 times less clock cycles than general rule based self-training. In this assisted SOM case longest duration was in countryside too. Lowest self-training duration was achieved by SOM with inner weights. This network self-training process was even 61 time faster than general rule based network self-training. As in endpoint detection and assisted SOM, network with inner weights used more self-training iteration in countryside than in other two locations.

In all locations 433 MHz range occupancy hasn't exceeded 0.1% (Table 3.14). For this band all networks detection ratio was similar. Difference between detection ratios was about 0.9%. Least occupied spectrum was in countryside and most emissions were detected in city. Least emissions were detected by SOM based on general rule.

In city and countryside 954 MHz range was exploited by primary users more than 50% (Table 3.14). Least primary user emissions were detected in township location. Difference between SOMs detection results are about 0.9%. As in 433 MHz range, in 954 MHz least primary user emissions were detected by general rule based SOM.

These generalizations can be made from experiments:

- Experimental structure implementation used up to 94% of Spartan6 LX75 FPGA resources.
- In all locations 433 MHz range was poorly exploited, therefore this range even in densely populated areas can be used freely.
- 954 MHz range in countryside was mostly exploited from all locations. RF spectrum occupancy by primary user was up to 74.5%. Average self-training process duration's for all SOMs in this location

were longest too. Therefore can be made assumption that longer self-training duration was influenced by densely occupied spectrum.
- All networks detection ratio was similar, although self-training duration was lowest in SOM with inner weights case. Therefore this SOM in online experiments can be selected as most performant.

3.5. Conclusions of Chapter 3

1. The elimination of the multiplication from the variance estimation algorithm decreases the sensitivity of the spectrum sensor and should not be used for practical applications. In addition, the implementation of the variance estimation without FIFO decreases average (for all signal types) false alarm rate from 5.63% to 1.94%.
2. The standard deviation, used as a feature vector instead signal spectrum average estimate, does not increase the sensitivity of the spectrum sensor.
3. The feature vector based on the Daubechies wavelet transform increases the sensitivity of the spectrum sensor and decreases average false alarm ratio more than 2 times in comparing with signal energy based feature extractors.
4. On an artificial neural network with binary step function based spectrum sensor has the best detection performance for wide-band signal type, however for other signal types better performance has on SOM based spectrum sensors.
5. All primary user emissions are detected with 1.4% false alarm rate, when self-training process modifications in spectrum sensors based on SOM are used.
6. The modification of the self-organizing map structure during self-training by adding inner neuron connections may decrease the number of self-training iterations from 2.6% to 44.6% before the endpoint is reached, keeping the same spectrum sensor sensitivity level.

General Conclusions

Two hypothesis were confirmed by investigation results presented in this dissertation. In addition, the modifications of the self-organizing map self-training phase, proposed in this dissertation, increased the efficiency of spectrum sensor implementation in FPGA based systems and reduced the total signal processing latency.

1. The application of the wavelet transform for the analysis of the signal spectrogram, in comparing with signal energy based features, may increase signal detection accuracy of the primary user transmission without any prior knowledge about the signal.
   1.1. The feature vector based on the Daubechies wavelet transform increases spectrum sensor sensitivity and decreases the average false alarm ratio from 1.94% to 1% for all signal types.
   1.2. The implementation of Haar wavelet transform, which can be used as a feature vector for the spectrum sensor is three times more efficient in comparing to Daubechies and Simlet wavelet transforms. However average false alarm rate reached with Haar wavelet transform is 1.8%.
2. Self-adaptation of the spectrum sensor to continuously changing radio environment can be achieved by the application of intelligent methods:
   2.1. A spectrum sensor based on an artificial neural network with binary step function and proposed network training activation

technique is able to detect 98.63% of primary user emissions with false alarm ratio below 2.73%.

2.2. A spectrum sensor based on a self-organizing map and classical self-training algorithm is able to detect all primary user emissions with false alarm ratio as low as 1% without the need of manual self-training data selection.

3. The modification of the self-organizing map topology during self-training phase increases the efficiency of the sensor implementation in FPGA based embedded systems.

3.1. The modification of the signal spectrum energy variance estimation formula reduces the feature extraction step delay from 37 to 21 cycles preserving the full primary user emission detection rate with average false alarm rate 1.94%.

3.2. An endpoint detection algorithm proposed for self-organizing map reduces the number of self-training iterations by 19–80% and preserves the SOM based spectrum sensor primary user detection and false alarm rate.

3.3. A training assistant proposed in this dissertation reduces the number of iterations required for self-training of the self-organizing map by 90%, comparing to the number of iterations, estimated accordingly to the number of neurons in the lattice multiplied by 500 as is recommended in the literature.

3.4. A modification of the self-organizing map structure during network training stage requires only 3–15% of recommended number iterations to reach the desired network weighs, uses 2.3 times less memory cells of the FPGA based embedded system and increases the self-training speed by 32.4 times.

3.5. The modification of the self-organizing map structure during self-training by adding inner neuron connections may decrease the number of self-training iterations from 2.6% to 44.6% before the endpoint is reached, keeping the same spectrum sensor sensitivity level.

References

The European Table of Frequency Allocations and Applications in the Frequency Range 8.3 kHz to 3000 GHz (Eca Table). 2015. Prieiga per internetą: <http://www.erodocdb.dk/Docs/doc98/official/pdf/ERCREP025.PDF>. [see 2 p.]

National frequency allocation table. 2015. Prieiga per internetą: <http://www.rrt.lt/lt/verslui/istekliai/radijo-dazniai/dazniu-valdymas.html>. [see 2 p.]

Aboaba, O. 2010. Noise analysis in mobile radio environments using pathlets and symlet 2 wavelets, in *TENCON 2010 – 2010 IEEE Region 10 Conference*, 1318–1323. ISSN pending. [see 19 p.]

Adoum, B.; Mossa, A.; Jeoti, V. 2010. Discrete wavelet packet transform based multiresolution spectrum sensing using cyclostationary feature detector, in *Intelligent and Advanced Systems (ICIAS), 2010 International Conference on*, 1–6. [see 14 p.]

Ali Tayaranian Hosseini, S.; Amindavar, H.; Ritcey, J. 2010. A new cyclostationary spectrum sensing approach in cognitive radio, in *Signal Processing Advances in Wireless Communications (SPAWC), 2010 IEEE Eleventh International Workshop on*, 1–4. ISSN 1948-3244. [see 20 p.]

Aliaga, R.; Gadea, R.; Colom, R.; Monzo, J.; Lerche, C.; Martinez, J.; Sebastia, A.; Mateo, F. 2008. Soc-based implementation of the backpropagation algorithm for mlp, in *Hybrid Intelligent Systems, 2008. HIS '08. Eighth International Conference on*, 744–749. [see 27 p.]

Appiah, K.; Hunter, A.; Dickinson, P.; Meng, H. 2012. Implementation and applications of tri-state self-organizing maps on fpga, *Circuits and Systems for Video Technology, IEEE Transactions on* 22(8): 1150–1160. ISSN 1051-8215. [see 29 p.]

Arianandа, D.; Lakshmanan, M.; Nikookar, H. 2009. A study on the application of wavelet packet transforms to cognitive radio spectrum estimation, in *Cognitive Radio Oriented Wireless Networks and Communications, 2009. CROWNCOM '09. 4th International Conference on*, 1–6. [see 14 p.]

Arshad, K.; Moessner, K. 2010. Mobility driven energy detection based spectrum sensing framework of a cognitive radio, in *Cognitive Wireless Systems (UKIWCWS), 2010 Second UK-India-IDRC International Workshop on*, 1–5. [see 10, 11, 12 p.]

Atanassov, K.; Krawczak, M.; Sotirov, S. 2008. Generalized net model for parallel optimization of feed-forward neural network with variable learning rate backpropagation algorithm, in *Intelligent Systems, 2008. IS '08. 4th International IEEE Conference*, vol. 2, 16–16–16–19. [see 60 p.]

Axell, E.; Leus, G.; Larsson, E.; Poor, H. 2012. Spectrum sensing for cognitive radio : State-of-the-art and recent advances, *Signal Processing Magazine, IEEE* 29(3): 101–116. ISSN 1053-5888. [see 7 p.]

Bagchi, S. 2014. Cognitive radio: Spectrum sensing under unknown signal and noise distributions, in *Computational Intelligence and Communication Networks (CICN), 2014 International Conference on*, 429–433. [see 1 p.]

Bin, S.; Longyang, H.; Chengshi, Z.; Zheng, Z.; Kyungsup, K. 2008. Energy detection based spectrum sensing for cognitive radios in noise of uncertain power, in *Communications and Information Technologies, 2008. ISCIT 2008. International Symposium on*, 628–633. [see 9, 10, 11, 12 p.]

Brugger, D.; Bogdan, M.; Rosenstiel, W. 2008. Automatic cluster detection in kohonen's som, *Neural Networks, IEEE Transactions on* 19(3): 442–459. [see 48 p.]

Bury, M.; Yashchyshyn, Y.; Modelski, J. 2008. Frequency domain measurements for an uwb imaging system, in *Microwaves, Radar and Wireless Communications, 2008. MIKON 2008. 17th International Conference on*, 1–4. [see 35 p.]

Cabric, D.; Tkachenko, A.; Brodersen, R. W. 2006. Experimental study of spectrum sensing based on energy detection and network cooperation, in *Proceedings of the First International Workshop on Technology and Policy for Accessing Spectrum*: TAPAS '06, New York, NY, USA: ACM. ISBN 1-59593-510-X. Prieiga per internetą: <http://doi.acm.org/10.1145/1234388.1234400>. [see 11 p.]

Cai, Q.; Chen, S.; Li, X.; Hu, N.; He, H.; Yao, Y.-D.; Mitola, J. 2010. An integrated incremental self-organizing map and hierarchical neural network approach for cognitive radio learning, in *Neural Networks (IJCNN), The 2010 International Joint Conference on*, 1–6. ISSN 1098-7576. [see 27 p.]

Caner, H.; Gecim, H.; Alkar, A. 2008. Efficient embedded neural-network-based license plate recognition system, *Vehicular Technology, IEEE Transactions on* 57(5): 2675–2683. ISSN 0018-9545. [see 29, 31 p.]

Carpenter, G.; Grossberg, S. 1992. A self-organizing neural network for supervised learning, recognition, and prediction, *Communications Magazine, IEEE* 30(9): 38–49. ISSN 0163-6804. [see 52 p.]

Chantaraskul, S.; Moessner, K. 2010. Experimental study of multi-resolution spectrum opportunity detection using wavelet analysis, in *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*, 1–6. [see 1, 15 p.]

Chen, H.; Tse, C.; Zhao, F. 2011. Optimal quantisation bit budget for a spectrum sensing scheme in bandwidth-constrained cognitive sensor networks, *Wireless Sensor Systems, IET* 1(3): 144–150. ISSN 2043-6386. [see 10, 11 p.]

Choi, K. W.; Jeon, W. S.; Jeong, D. G. 2009. Sequential detection of cyclostationary signal for cognitive radio systems, *Wireless Communications, IEEE Transactions on* 8(9): 4480–4485. ISSN 1536-1276. [see 20 p.]

Chu, G.; Niu, K.; Wu, W.; Yang, F.; Jiang, W. 2014. Performance evaluation of spectrum sensing based on wavelet entropy for cognitive radio networks, in *Wireless Personal Multimedia Communications (WPMC), 2014 International Symposium on*, 434–438. [see 14 p.]

Cohen, D.; Rebeiz, E.; Jain, V.; Eldar, Y.; Cabric, D. 2011. Cyclostationary feature detection from sub-nyquist samples, in *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2011 4th IEEE International Workshop on*, 333–336. [see 19 p.]

Gorunes cu, F.; El-Darzi, E.; Belciug, S.; Gorunescu, M. 2010. Patient grouping optimization using a hybrid self-organizing map and gaussian mixture model for length of stay-based clustering system, in *Intelligent Systems (IS), 2010 5th IEEE International Conference*, 173–178. [see 30 p.]

De Vito, L. 2012. A review of wideband spectrum sensing methods for cognitive radios, in *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*, 2257–2262. ISSN 1091-5281. [see 2, 8 p.]

Derakhshani, M.; Nasiri-kenari, M.; Le-Ngoc, T. 2010. Cooperative cyclostationary spectrum sensing in cognitive radios at low snr regimes, in *Communications (ICC), 2010 IEEE International Conference on*, 1–5. ISSN 1550-3607. [see 20 p.]

Dinesh Kumar, V.; Thomas, T. 2003. A modified fft-based algorithm for real-time computation of discrete wavelet transform, in *TENCON 2003. Conference on Convergent Technologies for the Asia-Pacific Region*, vol. 3, 1030–1034 Vol.3. [see 15 p.]

Dlugosz, R.; Kolasa, M.; Szulc, M. 2011. An fpga implementation of the asynchronous programmable neighborhood mechanism for wtm self-organizing map, in *Mixed Design of Integrated Circuits and Systems (MIXDES), 2011 Proceedings of the 18th International Conference*, 258–263. [see 29 p.]

Du, K.-L.; Mow, W. H. 2010. Affordable cyclostationarity-based spectrum sensing for cognitive radio with smart antennas, *Vehicular Technology, IEEE Transactions on* 59(4): 1877–1886. ISSN 0018-9545. [see 20 p.]

Egilmez, H.; Ortega, A. 2015. Wavelet-based compressed spectrum sensing for cognitive radio wireless networks, in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, 3157–3161. [see 17 p.]

El-Khamy, S.; Abdel-Malek, M.; Kamel, S. 2013. C9. a new wideband spectrum sensing technique based on compressive sensing, wavelet edge detection and parallel processing, in *Radio Science Conference (NRSC), 2013 30th National*, 202–210. [see 18, 19 p.]

Elramly, S.; Newagy, F.; Yousry, H.; Elezabi, A. 2011. Novel modified energy detection spectrum sensing technique for fm wireless microphone signals, in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, 59–63. [see 9, 10 p.]

Farhang-Boroujeny, B. 2008. Filter bank spectrum sensing for cognitive radios, *Signal Processing, IEEE Transactions on* 56(5): 1801–1811. ISSN 1053-587X. [see 12 p.]

Feng, G.; Bostian, C. 2008. A parallel computing based spectrum sensing approach for signal detection under conditions of low snr and rayleigh multipath fading, in *New Frontiers in Dynamic Spectrum Access Networks, 2008. DySPAN 2008. 3rd IEEE Symposium on*, 1–10. [see 20 p.]

Fessant, F.; Aknin, P.; Oukhellou, L.; Midenet, S. 2001. Comparison of supervised self-organizing maps using euclidian or mahalanobis distance in classification context, in *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*, vol. 2084, ed. by Mira, J.; Prieto, A.: Lecture Notes in Computer Science, Springer Berlin Heidelberg, 637–644. ISBN 978-3-540-42235-8. [see 52 p.]

Franzmeier, M.; Pohl, C.; Porrmann, M.; Ruckert, U. 2004. Hardware accelerated data analysis, in *Parallel Computing in Electrical Engineering, 2004. PARELEC 2004. International Conference on*, 309–314. [see 29 p.]

Fritzke, B. 1994. Growing cell structures—a self-organizing network for unsupervised and supervised learning, *Neural networks* 7(9): 1441–1460. [see 51 p.]

Fu, J.; Jun, P.; Zhengfa, Z. 2012. Improved weighted cooperative sensing algorithm based on distributed optimization in cognitive radio networks, in *Control Conference (CCC), 2012 31st Chinese*, 5487–5492. ISSN 1934-1768. [see 25 p.]

Gunes Kayacik, H.; Nur Zincir-Heywood, A.; Heywood, M. I. 2007. A hierarchical som-based intrusion detection system, *Eng. Appl. Artif. Intell.* 20(4): 439–451. ISSN 0952-1976. Prieiga per internetą: <http://dx.doi.org/10.1016/j.engappai.2006.09.005>. [see 28, 53 p.]

Herbert, J.; Yao, J. 2007. Gtsom: Game theoretic self-organizing maps, in *Trends in Neural Computation*, vol. 35, ed. by Chen, K.; Wang, L.: Studies in Computational Intelligence, Springer Berlin Heidelberg, 199–223. ISBN 978-3-540-36121-3. Prieiga per internetą: <http://dx.doi.org/10.1007/978-3-540-36122-0_8>. [see 48, 53, 58 p.]

Herrmann, L.; Ultsch, A. 2007. Label propagation for semi-supervised learning in self-organizing maps, in *International Workshop on Self-Organizing Maps, WSOM 2007*, vol. 6, 1–6. [see 28, 52 p.]

Huaguang, Z.; Zhanshan, W.; Derong, L. 2014. A comprehensive review of stability analysis of continuous-time recurrent neural networks, *Neural Networks and Learning Systems, IEEE Transactions on* 25(7): 1229–1262. ISSN 2162-237X. [see 23 p.]

Hulle, M. M. V. 2000. *Faithful Representations and Topographic Maps: From Distortion- to Information-Based Self-Organization*. New York, NY, USA: John Wiley & Sons, Inc. ISBN 0471345075. [see 53 p.]

Hwang, J.-K.; Li, Y.-P. 2009. Lms algorithm with an adaptive neural network cost function, *WTOC* 8(8): 935–940. ISSN 1109-2742. Prieiga per internetą: <http://dl.acm.org/citation.cfm?id=1718056.1718075>. [see 60 p.]

Ibrahim, F. 2008. Optimal linear neuron learning and kalman filter based backpropagation neural network for dgps/ins integration, in *Position, Location and Navigation Symposium, 2008 IEEE/ION*, 1175–1189. [see 26 p.]

Imani, S.; Dehkordi, A. B.; Kamarei, M. 2011. Adaptive sub-optimal energy detection based wideband spectrum sensing for cognitive radios, in *Electrical, Control and Computer Engineering (INECCE), 2011 International Conference on*, 22–26. [see 9, 11, 12 p.]

Čitavičius, A.; Jonavičius, A. 2006. Skaitmeninio moduliavimo sistemų su kvaziortogonaliosiomis sekomis modeliavimas, *Elektronika ir Elektrotechnika* 66(6): 11–16. ISSN 1392-1215. [see 13 p.]

Ivanikovas, S.; Dzemyda, G.; Medvedev, V. 2008. *Advances in Databases and Information Systems: 12th East European Conference, ADBIS 2008, Pori, Finland, September 5-9, 2008. Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, Chap. Large Datasets Visualization with Neural Network Using Clustered Training Data, 143–152. ISBN 978-3-540-85713-6. Prieiga per internetą: <http://dx.doi.org/10.1007/978-3-540-85713-6_11>. [see 27 p.]

Ivanovas, E.; Navakauskas, D. 2012. Towards speaker identification system based on dynamic neural network, *Elektronika ir Elektrotechnika* 18(10): 69–72. ISSN 1392–1215. [see 22 p.]

Janušauskas, A.; Jurkonis, R.; Lukoševičius, A.; Kurapkienė, S.; Paunksnis, A. 2005. The empirical mode decomposition and the discrete wavelet transform for detection of human cataract in ultrasound signals, *INFORMATICA* 16(4): 541–556. [see 13 p.]

Jaspreet, K.; Rajneet, K. 2014. Biomedical images denoising using symlet wavelet with wiener filter, *International Journal of Engineering Research and Applications, IJERA 2013* 3(3): 548–550. ISSN 2248-9622. [see 14, 15 p.]

Javed, F.; Mahmood, A. 2010. The use of time frequency analysis for spectrum sensing in cognitive radios, in *Signal Processing and Communication Systems (ICSPCS), 2010 4th International Conference on*, 1–7. [see 20 p.]

Jayavalan, S.; Mohamad, H.; Aripin, N.; Ismail, A.; Ramli, N.; Yaacob, A.; Ann Ng, M. 2014. Measurements and analysis of spectrum occupancy in the cellular and tv bands, *Lecture Notes on Software Engineering* 2(2): 133–138. ISSN 2301-3559. [see 2 p.]

Joshi, D.; Popescu, D.; Dobre, O. 2011a. Gradient-based threshold adaptation for energy detector in cognitive radio systems, *Communications Letters, IEEE* 15(1): 19–21. ISSN 1089-7798. [see 1 p.]

Joshi, D.; Popescu, D.; Dobre, O. 2011b. Gradient-based threshold adaptation for energy detector in cognitive radio systems, *Communications Letters, IEEE* 15(1): 19–21. ISSN 1089-7798. [see 11 p.]

Jun, W. 2010. Analysis and design of a k -winners-take-all model with a single state variable and the heaviside step activation function, *Neural Networks, IEEE Transactions on* 21(9): 1496–1506. ISSN 1045-9227. [see 24 p.]

Kannan, R.; Ravi, S. 2012. Digital signals classification in cognitive radio based on discrete wavelet transform, in *Control Engineering and Communication Technology (ICCECT), 2012 International Conference on*, 870–873. [see 13 p.]

Karthik, D.; Manikandan, P.; Hari, R.; Srinivas, A. 2011. A novel algorithm for spectrum sensing using adaptive wavelet edge detection scheme, in *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, vol. 3, 118–122. [see 17 p.]

Katkevičius, A.; Mališauskas, V.; Plonis, D.; Serackis, A. 2012. Calculations of characteristics of microwave devices using artificial neural networks, *Przegląd elektrotechniczny. Warsaw : Wydawnictwo SIGMA-NOT* 88(1a): 281–285. ISSN 0033-2097. [see 22 p.]

Khozeimeh, F.; Haykin, S. 2012. Brain-inspired dynamic spectrum management for cognitive radio ad hoc networks, *Wireless Communications, IEEE Transactions on* 11(10): 3509–3517. ISSN 1536-1276. [see 27 p.]

Kiang, M. Y. 2002. Extending the kohonen self-organizing map networks for clustering analysis, *Comput. Stat. Data Anal.* 38(2): 161–180. ISSN 0167-9473. Prieiga per internetą: <http://dx.doi.org/10.1016/S0167-9473(01)00040-8>. [see 58, 60 p.]

Kohonen, T. 1990. The self-organizing map, *Proceedings of the IEEE* 78(9): 1464–1480. ISSN 0018-9219. [see 60 p.]

Kohonen, T. 1991. Self-Organizing Maps: Optimization approaches, in *Artificial Neural Networks*, vol. II, ed. by Kohonen, T.; Mäkisara, K.; Simula, O.; Kangas, J., Amsterdam, Netherlands: North-Holland, 981–990. [see 48 p.]

Koikkalainen, P.; Oja, E. 1990. Self-organizing hierarchical feature maps, in *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*, IEEE, 279–284. [see 51 p.]

Kokkeler, A.; Smit, G.; Krol, T.; Kuper, J. 2007. Cyclostationary feature detection on a tiled-soc, in *Design, Automation Test in Europe Conference Exhibition, 2007. DATE '07*, 1–6. [see 20 p.]

Manola kos, I.; Logaras, E. 2007. High throughput systolic som ip core for fpgas, in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2, II–61–II–64. ISSN 1520-6149. [see 31 p.]

Kurasova, O.; Molytė, A. 2011. Integration of the self-organizing map and neural gas with multidimensional scaling, in *Information Technology and Control*, vol. 40, 12–20. ISSN 1392-124X. [see 27 p.]

Kyouwoong, K.; Akbar, I.; Bae, K.; Jung-Sun, U.; Spooner, C.; Reed, J. 2007. Cyclostationary approaches to signal detection and classification in cognitive radio, in *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, 212–215. [see 20 p.]

Kyungtae, K.; Yan, X.; Rangarajan, S. 2010. Energy detection based spectrum sensing for cognitive radio: An experimental study, in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, 1–5. ISSN 1930-529X. [see 1, 11 p.]

Lampinen, J.; Kostiainen, T. 1999. Overtraining and model selection with the self-organizing map, in *Neural Networks, 1999. IJCNN '99. International Joint Conference on*, vol. 3, 1911–1915 vol.3. ISSN 1098-7576. [see 51 p.]

Lampinen, J.; Oja, E. 1992. Clustering properties of hierarchical self-organizing maps, *Journal of Mathematical Imaging and Vision* 2(2-3): 261–272. [see 48 p.]

Laptik, R.; Navakauskas, D. 2005. Application of artificial neural networks for the recognition of car number plates, *Elektronika ir Elektrotechnika* 64(8): 27–30. ISSN 1392–1215. [see 22 p.]

Larsson, E.; Skoglund, M. 2008. Cognitive radio in a frequency-planned environment: some basic limits, *Wireless Communications, IEEE Transactions on* 7(12): 4800–4806. ISSN 1536-1276. [see 11 p.]

Levinskis, A. 2013. Convolutional neural network feature reduction using wavelet transform, *Elektronika ir Elektrotechnika* 19(3): 61–64. ISSN 1392–1215. [see 22 p.]

Li, B.; Zhou, Z.; Zou, W.; Zhao, F.; Li, Z.; Li, D. 2010. Interference mitigation between ultra-wideband sensor network and other legal systems, *EURASIP J. Wirel. Commun. Netw.* 2010: 6:1–6:15. ISSN 1687-1472. Prieiga per internetą: <`http://dx.doi.org/10.1155/2010/290306`>. [see 24 p.]

Liang, Y.; SiXing, Y.; Weijun, H.; ShuFang, L. 2011. Spectrum behavior learning in cognitive radio based on artificial neural network, in *Military Communications Conference, MILCOM 2011*, 25–30. ISSN 2155-7578. [see 23 p.]

Lin, Y.; He, C. 2008. Subsection-average cyclostationary feature detection in cognitive radio, in *Neural Networks and Signal Processing, 2008 International Conference on*, 604–608. [see 19 p.]

Lin, Y.; He, C.; Jiang, L.; He, D. 2009. A spectrum sensing method in cognitive radio based on the third order cyclic cumulant, in *Wireless Communications Signal Processing, 2009. WCSP 2009. International Conference on*, 1–5. [see 20 p.]

Mathew, M.; Premkumar, A.; Lau, C. 2010. Multiple access scheme for multi user cognitive radio based on wavelet transforms, in *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, 1–5. ISSN 1550-2252. [see 15, 18, 19 p.]

Matuzevičius, D.; Serackis, A.; Navakauskas, D. 2010. Application of k-means and mlp in the automation of matching of 2de gel images, in *Artificial Neural Networks – ICANN 2010*, vol. 6352, ed. by Diamantaras, K.; Duch, W.; Iliadis, L.: Lecture Notes in Computer Science, Springer Berlin Heidelberg, 541–550. ISBN 978-3-642-15818-6. Prieiga per internetą: <http://dx.doi.org/10.1007/978-3-642-15819-3_70>. [see 21 p.]

Mehdawi, M.; Riley, N.; Paulson, K.; Fanan, A.; Ammar, M. 2013. Spectrum occupancy survey in hull-uk for cognitive radio applications: Measurement & analysis, in *International Journal of Scientific Technology Research, IJSTR*, vol. 2, 231–236. ISSN 2277-8616. [see 2 p.]

Merkevičius, E.; Garšva, G. 2007. Prediction of changes of bankruptcy classes with neuro-discriminate model based on the self-organizing maps, in *INFORMATION TECHNOLOGY AND CONTROL*, vol. 36, 145–151. ISSN 1392-124X. [see 27 p.]

Merkevičius, E.; Garšva, G.; Simutis, R. 2004. Forecasting of credit classes with the self-organizing maps, in *INFORMATION TECHNOLOGY AND CONTROL*, vol. 43, 61–66. ISSN 1392-124X. [see 27 p.]

Moosavi, R.; Larsson, E. 2014. Fast blind recognition of channel codes, *Communications, IEEE Transactions on* 62(5): 1393–1405. ISSN 0090-6778. [see 8 p.]

Moreira, M.; Fiesler, E. 1995. *Neural Networks with Adaptive Learning Rate and Momentum Terms*. [see 53 p.]

Nair, P.; Vinod, A.; Krishna, A. 2010. An adaptive threshold based energy detector for spectrum sensing in cognitive radios at low snr, in *Communication Systems (ICCS), 2010 IEEE International Conference on*, 574–578. [see 1 p.]

Nascimento, Z.; Sadok, D.; Fernandes, S. 2013. A hybrid model for network traffic identification based on association rules and self-organizing maps (som), in *The Ninth International Conference on Networking and Services, ICNS 2013*, vol. 9, 213–219. ISSN 2308-4006. [see 27 p.]

Nastase, C.-V.; Martian, A.; Vladeanu, C.; Marghescu, I. 2014. An accurate average energy detection algorithm for spectrum sensing in cognitive radio systems, in *Electronics and Telecommunications (ISETC), 2014 11th International Symposium on*, 1–4. [see 1 p.]

Nawi, N.; Ransing, R.; Ransing, M. 2008. A new method to improve the gradient based search direction to enhance the computational efficiency of back propagation based neural network algorithms, in *Modeling Simulation, 2008. AICMS 08. Second Asia International Conference on*, 546–552. [see 60 p.]

Oba, Y.; Yamamoto, K.; Nagai, T.; Hikawa, H. 2011. Hardware design of a color quantization with self-organizing map, in *Intelligent Signal Processing and Communications Systems (ISPACS), 2011 International Symposium on*, 1–6. [see 29 p.]

Pateritsas, C.; Pertselakis, M.; Stafylopatis, A. 2004. A som-based classifier with enhanced structure learning, in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, vol. 5, 4832–4837 vol.5. ISSN 1062-922X. [see 27, 51 p.]

Paukštaitis, V.; Dosinas, A. 2010. Pulsed neural network cell in integrated circuit, *Elektronika ir Elektrotechnika* 98(2): 35–40. ISSN 1392–1215. [see 22 p.]

Penna, F.; Pastrone, C.; Spirito, M.; Garello, R. 2009. Energy detection spectrum sensing with discontinuous primary user signal, in *Communications, 2009. ICC '09. IEEE International Conference on*, 1–5. ISSN 1938-1883. [see 1, 11 p.]

Perera, L.; Herath, H. 2011. Review of spectrum sensing in cognitive radio, in *Industrial and Information Systems (ICIIS), 2011 6th IEEE International Conference on*, 7–12. [see 7, 8 p.]

Pikutis, M.; Vasarevičius, D.; Martavičius, R. 2014. Maximum power point tracking in solar power plants under partially shaded condition, *Elektronika ir Elektrotechnika* 20(4): 49–52. ISSN 1392–1215. [see 22 p.]

Plonis, D.; Mališauskas, V.; Serackis, A. 2005. Semi-automatic analysis of gyrotropic semiconductor waveguides using neural network, *Acta Physica Polonica A. Warszawa : Polish Academy of Sciences.* 119(4): 542–547. ISSN 0587-4246. [see 22 p.]

Popoola, J.; Van Olst, R. 2011. Application of neural network for sensing primary radio signals in a cognitive radio environment, in *AFRICON, 2011*, 1–6. ISSN 2153-0025. [see 21 p.]

Popoola, J. J.; van Olst, R. 2013. The performance evaluation of a spectrum sensing implementation using an automatic modulation classification detection method with a universal software radio peripheral, *Expert Syst. Appl.* 40(6): 2165–2173. ISSN 0957-4174. Prieiga per internetą: <http://dx.doi.org/10.1016/j.eswa.2012.10.047>. [see 26 p.]

Pupeikis, R. 2015. Fast fourier transform revisited, in *Proc. of the Lithuanian Mathematical Society*, 113–118. ISSN 0132-2818. [see 35 p.]

Ramirez, C.; Gomez, G.; Alarcon, A.; Beaz, L.; Enriquez, C. 2011. A biometric system based on neural networks and svm using morphological feature extraction from hand-shape images, in *INFORMATICA*, vol. 22, 225–240. [see 22, 23 p.]

Rasheed, H.; Rajatheva, N.; Haroon, F. 2010. Spectrum sensing with energy detection under shadow-fading condition, in *Wireless Pervasive Computing (ISWPC), 2010 5th IEEE International Symposium on*, 104–109. [see 1, 9, 11 p.]

Sabat, S.; Srinu, S.; Kumar, N.; Udgata, S. 2010. Fpga realization of spectrum sensing techniques for cognitive radio network, in *Cognitive Radio (IWCR), 2010 International Workshop on*, 1–5. [see 8 p.]

Sanchez, M.; Cuinas, I.; Alejos, A. 2007. Interference and impairments in radio communication systems due to industrial shot noise, in *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, 1849–1854. [see 10 p.]

Satiyan, M.; Hariharan, M.; Nagarajan, R. 2010. Comparison of performance using daubechies wavelet family for facial expression recognition, in *Signal Processing and Its Applications (CSPA), 2010 6th International Colloquium on*, 1–5. [see 17 p.]

Serackis, A.; Navakauskas, D. 2008. Reconstruction of overlapped protein spots using rbf networks, *Elektronika ir Elektrotechnika* 81(1): 83–88. ISSN 1392–1215. [see 22 p.]

Serackis, A.; Navakauskas, D.; Matuzevicius, D. 2010. 2de gel image preprocessing using self-organising maps, in *The International Society for Optical Engineering Wilga, Poland : SPIE 2010*, vol. 7745, 1–6. ISSN 0277-786X. [see 27 p.]

Seshukumar, K.; Saravanan, R.; Suraj, M. 2013. Spectrum sensing review in cognitive radio, in *Emerging Trends in VLSI, Embedded System, Nano Electronics and Telecommunication System (ICEVENT), 2013 International Conference on*, 1–4. [see 2, 7 p.]

Sharma, A.; Dey, S. 2013. Using self-organizing maps for sentiment analysis, *CoRR* abs/1309.3946: 1–4. Prieiga per internetą: <`http://arxiv.org/abs/1309.3946`>. [see 28 p.]

Sicheng, L.; Chunpeng, W.; Hai, L.; Boxun, L.; Yu, W.; Qinru, Q. 2015. Fpga acceleration of recurrent neural network based language model, in *Field-Programmable Custom Computing Machines (FCCM), 2015 IEEE 23rd Annual International Symposium on*, 111–118. [see 23 p.]

Sledevič, T.; Navakauskas, D. 2015. Towards optimal fpga implementation of lattice-ladder neuron and its training circuit, in *Information, Electronic and Electrical Engineering (AIEEE), 2015 IEEE 3rd Workshop on Advances in*, 1–4. [see 22 p.]

Solodov, M.; Svaiter, B. 2000. A comparison of rates of convergence of two inexact proximal point algorithms, in *Nonlinear Optimization and Related Topics*, vol. 36, ed. by Pillo, G.; Giannessi, F.: Applied Optimization, Springer US, 415–427. ISBN 978-1-4419-4823-6. [see 53 p.]

Srinu, S.; Sabat, S. 2010. Fpga implementation of spectrum sensing based on energy detection for cognitive radio, in *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on*, 126–131. [see 9, 11 p.]

Srinu, S.; Sabat, S.; Udgata, S. 2011. Wideband spectrum sensing based on energy detection for cognitive radio network, in *Information and Communication Technologies (WICT), 2011 World Congress on*, 651–656. [see 1, 12 p.]

Stankevičius, E.; Paulikas, .; Medeišis, A.; Anskaitis, A. 2015. Electromagnetic compatibility problems in lte mobile networks 800 mhz frequency band, *Electronics and electrical engineering* 26(6): 86–89. ISSN 1392-1215. [see 1 p.]

Stankevičius, G. 2001. Forming of the investment portfolio using the self-organizing maps (som), in *INFORMATICA*, vol. 12, 573–584. ISSN 0868-4952. [see 27, 29 p.]

Statkus, A.; Paulikas, . 2012. Analysis of home wi-fi internet access networks situation in vilnius city, *Electronics and electrical engineering* 118(2): 77–88. ISSN 1392-1215. [see 2 p.]

Stefanovič, P.; Kurasova, O. 2014. Creation of text document matrices and visualization by self-organizing map, in *Information Technology and Control*, vol. 43, 37–46. ISSN 1392-124X. [see 27 p.]

Stefanovic, P.; Kurasova, O. 2011. Visual analysis of self-organizing maps, in *Nonlinear Analysis: Modelling and Control*, vol. 16, 488–504. ISSN 1392-5113. [see 27 p.]

Stojanović, R.; Karadaglić, D.; Mirković, M.; Milošević, D. 2011. A fpga system for qrs complex detection based on integer wavelet transform, *Journal of the Institute of Measurement Science SAS* 11(4): 132–138. ISSN 1335 - 8871. [see 17 p.]

Sun, H.; Nallanathan, A.; Wang, C.-X.; Chen, Y. 2013. Wideband spectrum sensing for cognitive radio networks: a survey, *Wireless Communications, IEEE* 20(2): 74–81. ISSN 1536-1284. [see 2, 7, 8 p.]

Sun, M.; Shi, M.; Zhao, C.; Li, B. 2015. Spectrum sensing for radar communications with unknown noise variance and time-variant channel, in *Communication Workshop (ICCW), 2015 IEEE International Conference on*, 2513–2518. [see 9 p.]

Sunday, I.; Goodyer, E.; Gow, J.; Gongora, M. 2015. Spectrum occupancy survey in leicester, uk, for cognitive radio application, *International Journal of Scientific & Engineering Research* 6(8): 385–392. ISSN 2229-5518. [see 2 p.]

Szadkowski, Z. 2012. Fpga implementation of the 32-point dft for a wavelet trigger in cosmic rays experiments, in *Real Time Conference (RT), 2012 18th IEEE-NPSS*, 1–8. [see 15 p.]

Taher, T. M.; Bacchus, R. B.; Zdunek, K. J.; Roberson, D. A. 2011. Long-term spectral occupancy findings in Chicago, in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on*, IEEE, 100–107. [see 2 p.]

Tamaševičiūtė, E.; Mykolaitis, G.; Tamaševičius, A. 2012. "realistic" electronic neuron, in *Int. Symp. on Nonlinear Theory and its Applications, Nolta2012*, 860–863. [see 22 p.]

Tamukoh, H.; Aso, T.; Horio, K.; Yamakawa, T. 2004. Self-organizing map hardware accelerator system and its application to realtime image enlargement, in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 4, 2683–2687 vol.4. ISSN 1098-7576. [see 29 p.]

Tian, Z.; Giannakis, G. 2006. A wavelet approach to wideband spectrum sensing for cognitive radios, in *Cognitive Radio Oriented Wireless Networks and Communications, 2006. 1st International Conference on*, 1–5. [see 14, 15 p.]

Tumuluru, V.; Wang, P.; Niyato, D. 2010. A neural network based spectrum prediction scheme for cognitive radio, in *Communications (ICC), 2010 IEEE International Conference on*, 1–5. ISSN 1550-3607. [see 21 p.]

Umebayashi, K.; Lehtomaki, J.; Hatakeyama, S.; Suzuki, Y. 2011. Cyclostationary spectrum sensing under four-level hypothesis for spectrum sharing, in *Personal Indoor and Mobile Radio Communications (PIMRC), 2011 IEEE 22nd International Symposium on*, 2338–2343. ISSN pending. [see 20 p.]

Valantinas, J.; Kančelkis, D.; Valantinas, R.; Viščiūtė, G. 2013. Improving space localization properties of the discrete wavelet transform, *INFORMATICA* 24(4): 657–675. [see 13, 17 p.]

Vaserevičius, D.; Martavičius, R.; Pikutis, M. 2012. Application of artificial neural networks for maximum power point tracking of photovoltaic panels, *Elektronika ir Elektrotechnika* 18(10): 65–68. ISSN 1392–1215. [see 22 p.]

Vegas-Azcarate, S.; Gautama, T.; Van Hulle, M. 2005. *Topology preservation in topographic maps*. Technical Reports on Statistics and Decision Sciences, Rey Juan Carlos University. [see 48, 52, 58 p.]

Villmann, T.; Cichocki, A.; Principe, J. 2011. Information theory related learning, in *European Symposium on Artificial Neural Networks, ESANN 2011*, vol. 19, 1–10. ISBN 2-87419-044-6. [see 27 p.]

Wang, S.; Zhang, B.; Liu, H.; Xie, L. 2010. Parallelized cyclostationary feature detection on a software defined radio processor, in *Signals Systems and Electronics (ISSSE), 2010 International Symposium on*, vol. 1, 1–4. [see 20 p.]

Wang, Z.; Li, S.; Qixian, C.; Su, S.; Liu, M. 2009. Multi-spectrum image fusion algorithm based on weighted and improved wavelet transform, in *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, vol. 4, 63–66. [see 15 p.]

Wang, Z.; Salous, S. 2011. Spectrum occupancy statistics and time series models for cognitive radio, *Journal of Signal Processing Systems* 62(2): 145–155. ISSN 1939-8018. Prieiga per internetą: <http://dx.doi.org/10.1007/s11265-009-0352-5>. [see 1 p.]

Wen-Bin, C.; Chih-Kai, Y.; Yuan-Hao, H. 2011. Energy-saving cooperative spectrum sensing processor for cognitive radio system, *Circuits and Systems I: Regular Papers, IEEE Transactions on* 58(4): 711–723. ISSN 1549-8328. [see 12 p.]

Xiang-lin, Z.; Yuan-an, L.; Wei-Wen, W.; Dong-Ming, Y. 2008. Channel sensing algorithm based on neural networks for cognitive wireless mesh networks, in *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on*, 1–4. [see 23 p.]

Yadav, A.; Roy, R.; Kumar, A.; Kumar, C.; Dhakad, S. 2015. De-noising of ultrasound image using discrete wavelet transform by symlet wavelet and filters, in *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, 1204–1208. [see 19 p.]

Yamamoto, K.; Oba, Y.; Rikuhashi, Z.; Hikawa, H. 2011. Automatic generation of hardware self-organizing map for fpga implementation, in *Intelligent Signal Processing and Communications Systems (ISPACS), 2011 International Symposium on*, 1–6. [see 31 p.]

Yang, H.; Xie, X.; Wang, R. 2012. Som-ga-svm detection based spectrum sensing in cognitive radio, in *Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on*, 1–7. ISSN 2161-9646. [see 27 p.]

Yang, M.; Dai, H.; Ye, N. 2014. A dynamic threshold computing method based on som neural network in frequency occupancy analysis, in *Wireless Communications, Networking and Mobile Computing (WiCOM 2014), 10th International Conference on*, 153–158. [see 27 p.]

Yano, H.; Takyu, O.; Fujii, T.; Ohtsuki, T. 2011. Low complexity cyclostationary feature detection method to compenstate cyclostationarity-degradation by guard interval insertion, in *Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM), 2011 Sixth International ICST Conference on*, 51–55. [see 19 p.]

Yi, Z.; Xianzhong, X.; Lili, Y. 2009. Cooperative spectrum sensing based on blind source separation for cognitive radio, in *Future Information Networks, 2009. ICFIN 2009. First International Conference on*, 398–402. [see 10, 11 p.]

Yixian, L.; Chunyan, Z.; Hongjiang, W.; Gang, W. 2010. Energy detection threshold optimization for cooperative spectrum sensing, in *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, vol. 4, 566–570. [see 11 p.]

Yonghong, Z.; Ying-Chang, L.; Rui, Z. 2008. Blindly combined energy detection for spectrum sensing in cognitive radio, *Signal Processing Letters, IEEE* 15: 649–652. ISSN 1070-9908. [see 11 p.]

Youn, Y.; Jeon, H.; Jung, H.; Lee, H. 2007. Discrete wavelet packet transform based energy detector for cognitive radios, in *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, 2641–2645. ISSN 1550-2252. [see 13 p.]

Young, A.; Bostian, C. 2013. Simple and low-cost platforms for cognitive radio experiments [application notes], *Microwave Magazine, IEEE* 14(1): 146–157. ISSN 1527-3342. [see 1 p.]

Yu-Jie, T.; Qin-yu, Z.; Wei, L. 2010. Artificial neural network based spectrum sensing method for cognitive radio, in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*, 1–4. [see 21, 23 p.]

Yucek, T.; Arslan, H. 2009. A survey of spectrum sensing algorithms for cognitive radio applications, *Communications Surveys Tutorials, IEEE* 11(1): 116–130. ISSN 1553-877X. [see 2, 7, 8 p.]

Zhang, N.; Wang, H.; Creput, J.-C.; Moreau, J.; Ruichek, Y. 2013. Cellular gpu model for structured mesh generation and its application to the stereo-matching disparity map, in *Multimedia (ISM), 2013 IEEE International Symposium on*, 53–60. [see 29 p.]

Zhang, P.; Qiu, R.; Guo, N. 2011. Demonstration of spectrum sensing with blindly learned features, *Communications Letters, IEEE* 15(5): 548–550. ISSN 1089-7798. [see 11 p.]

Zhao, H.; Wang, F.; Liu, J. 2014a. A robust audio watermarking algorithm based on svd-dwt, *Elektronika ir Elektrotechnika* 20(1): 75–80. ISSN 1392-1215. [see 13 p.]

Zhao, J.; Wang, M.; Yuan, J. 2011. Based on neural network spectrum prediction of cognitive radio, in *Electronics, Communications and Control (ICECC), 2011 International Conference on*, 762–765. [see 23 p.]

Zhao, Y.; Wu, Y.; Wang, J.; Zhong, X.; Mei, L. 2014b. Wavelet transform for spectrum sensing in cognitive radio networks, in *Audio, Language and Image Processing (ICALIP), 2014 International Conference on*, 565–569. [see 14 p.]

Zhuan, Y.; Memik, G.; Grosspietsch, J. 2008. Energy detection using estimated noise variance for spectrum sensing in cognitive radio networks, in *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*, 711–716. ISSN 1525-3511. [see 1, 10 p.]

List of Publications by the Author on the Topic of the Dissertation

Papers in the Reviewed Scientific Journals

Stasionis, L.; Serackis, A. 2015. A New Method for Adaptive Selection of Self-Organizing Map Self-training Endpoint, *Baltic Journal of Modern Computing*, 1–14. ISBN 978-1-4799-8568-5. (ISI Web of Science).

Stasionis, L.; Serackis, A. 2014. Burst signal detector based on signal energy and standard deviation, *Elektronika ir Elektrotechnika* 20(2), 48–51. ISSN 1392–1215. (ISI Web of Science). IF=0.56.

Serackis, A.; Tamulevicius, G.; Sledevic, T.; Stasionis, L.; Navakauskas, D. 2014. Self-organizing feature map preprocessed vocabulary renewal algorithm for the isolated word recognition system, *Elektronika ir Elektrotechnika* 20(6), 114–117. ISSN 1392–1215. (ISI Web of Science). IF=0.56.

Stasionis, L.; Serackis, A. 2011. Selection of an optimal adaptive filter for speech signal noise cancellation using C6455 DSP, *Elektronika ir Elektrotechnika* 115(9), 101–104. ISSN 1392–1215. (ISI Web of Science). IF=0.56.

Other Papers

Stasionis, L.; Serackis, A. 2015. A new self-organizing map topology for real-time spectrum sensing and fast convergence, in *Proceedings of EUROCON - International Conference on Computer as a Tool (EUROCON)*, 1–4. ISBN 978-1-4799-8568-5. (Thomson ISI Proceedings).

Serackis, A.; Stasionis, L. 2014. Spectrum sensor based on a self-organizing feature map, in *Proceedings of 22nd International Conference, NDES 2014, Albena, Bulgaria, July 4–6, 2014. Nonlinear Dynamics of Electronic Systems. Communications in Computer and Information Science, Springer International Publishing*, vol. 438, 51–58. ISBN 978-3-319-08671-2. (Thomson ISI Proceedings).

Sledevic, T.; Stasionis, L. 2013. FPGA-based implementation of Lithuanian isolated word recognition algorithm, *Science – Future Of Lithuania*, 101–104. ISSN 2029-2341. (ICONDA).

Stasionis, L.; Sledevic, T. 2013. Energy detector implementaton in FPGA for estimation of word boundaries, *Science – Future Of Lithuania*, 105–108. ISSN 2029-2341. (ICONDA).

Stasionis, L.; Serackis, A. 2013. A new approach for spectrum sensing in wideband, in *Proceedings of EUROCON, 2013*, 125–132. ISBN 978-1-4673-2230-0. (Thomson ISI Proceedings).

Stasionis, L.; Serackis, A. 2012. Experimental study of spectrum sensing algorithm with low cost sdr, in *Proceedings of 22nd International Conference "Electromagnetic disturbances"*, vol. 22, 117–120. ISSN 1822-3249. (Thomson ISI Proceedings).

Santrauka lietuvių kalba

Įvadas

Problemos formulavimas

Šioje disertacijoje yra sprendžiama žemo signalas triukšmas santykio $\sim 0\,\mathrm{dB}$ signalo aptikimo plačioje dažnių juostoje problema, kai naudojamos tik kelios imtys požymiams apskaičiuoti. (Penna *et al.* 2009; Rasheed *et al.* 2010; Srinu *et al.* 2011; Zhuan *et al.* 2008).

Pagrindinis dėmesys yra skiriamas automatiniam signalo aptikimui, kai nėra jokios pradinės informacijos nei apie signalo turinį, nei apie signalo moduliacijos tipą ar signalo nešlio dažnį (Joshi *et al.* 2011a; Nair *et al.* 2010). Tokia problema dažnai sprendžiama kognityvinio radijo sistemose. Tam, kad būtų galima išsiųsti antrinio vartotojo signalą radijo kanalu, kuris jau užimtas pirminio vartotojo, spektro jutiklis turėtų aptikti pirminio vartotojo signalą (Stankevičius *et al.* 2015). Todėl spektro jutimo sistema turi užtikrinti visišką ar artimą visiškam pirminio vartotojo aptikimą(Nastase *et al.* 2014). Šiuo metu yra nagrinėjami skirtingi spektro analizės metodai, kuriais siekiama padidinti pirminio vartotojo signalų (emisijų) aptikimo realiame laike efektyvumą (Chantaraskul, Moessner 2010; Kyungtae *et al.* 2010; Young, Bostian 2013). Ypač problematiška aptikti žemo energijos lygio, lyginant su kanalo triukšmo lygiu, pirminio vartojo signalus, kai triukšmo skirstinys nėra visiškai Gauso (Bagchi 2014; Wang, Salous 2011). Todėl jutiklis turi nuolat arba periodiškai prisitaikyti prie besikeičiančios radijo spektro aplinkos.

Darbo aktualumas

Šiuo metu yra labai maža radijo spektro dalis (iki 3 GHz), kuri nepaskirta paslaugų tiekėjams (Lietuvos Respublikos ryšių reguliavimo tarnyba 2015, Elektroninių ryšių komitetas prie Europos pašto ir telekomunikacijų administracijų konferencijos 2015). Tačiau net tankiai apgyvendintose teritorijose spektro užimtumas labai aukštuose ir ultra aukštuose dažniuose yra tik 10–15%. Tai reiškia, kad daugiau kaip 85% spektro yra nepanaudota (Taher *et al.* 2011). Tačiau kai kurie nelicencijuoti radijo dažnių ruožai, pavyzdžiui 2,4 GHz, yra apkrauti vartotojų (Statkus, Paulikas 2012).

Šiuo laikotarpiu daug tyrimų orientuojasi į efektyvesnį nenaudojamo radijo spektro išnaudojimą, net jei jis jau yra priskirtas pirminiam vartotojui (De Vito 2012; Jayavalan *et al.* 2014; Mehdawi *et al.* 2013; Seshukumar *et al.* 2013; Sun *et al.* 2013; Sunday *et al.* 2015; Yucek, Arslan 2009). Spektro jutiklis yra kognityviniais radijais grįstų komunikacijų sistemos fizinio lygmens dalis, kuri atsakinga už pirminio vartotojo aptikimą triukšmingoje aplinkoje. Antrinio vartotojo signalo komunikacijos galimos tik tada, kai nėra pirminio vartotojo signalo.

Spektro jutiklio, kuris aptinka radijo spektre skyles, jautrumas tiesiogiai priklauso nuo algoritmo, kuris naudojamas kognityviniame radijuje. Taipogi spektro jutiklis turi vertinti nuolat besikeičiančią radijo spektro aplinką. Todėl spektro jutiklis turi prisiderinti prie pakitusios aplinkos.

Tyrimų objektas

Disertacijos tyrimų objektas yra saviorganizuojantis spektro jutiklis, kuris gali aptikti visas pirminių vartotojų signalų emisijas 25 MHz juostoje, su kuo mažesniu klaidos santykiu. Pirminių vartotojų signalų emisijos gali būti aptiktos bet kur iki 3 GHz radijo dažnių ruože:

- analizuojamame signalo spektre triukšmo komponentės gali būti kartu su pirminių naudotojų signalais arba be jų;
- jeigu kartu su triukšmo komponentėmis yra pirminių naudotojų signalai, nėra žinoma apie pirminių naudotojų signalų laikines ar dažnines savybes.

Darbo tikslas

Disertacijos tikslas – ištirti ir įgyvendinti saviorganizuojančius spektro analizės metodus, tinkamus atpažinti pirminio naudotojo signalą, paveiktą triukšmu, kai nėra žinoma pirminė informacija apie šį signalą.

Darbo uždaviniai

Po atliktos dabar vykdomų tyrimų apžvalgos, iškeltos dvi hipotezės:

1. Taikant diskrečiąją vilnelių transformaciją signalo spektrogramai galima padidinti signalo atpažinimo triukšme metodų tikslumą, lyginant su signalo energija grįstų požymių išskyrimo metodais.

2. Spektro jutiklio prisitaikymas prie nuolat besikeičiančios radijo aplinkos gali būti pasiektas naudojant saviorganizuojančius intelektualiuosius metodus, keičiant jų mokymosi algoritmus įgyvendinimui įterptinėse sistemose.

Disertacijos hipotezėms patikrinti suformuluoti trys uždaviniai:

1. Ištirti vilnelių transformacijos taikymo spektro jutikliams efektyvumą.
2. Ištirti dirbtinių daugiasluoksnių perceptroninių tinklų taikymo spektro jutikliams tinkamumą.
3. Sukurti spektro jutimo metodus, grįstus Kohonen tinklais, ir ištirti tinklų patobulinimus, siekiant sumažinti mokymosi trukmę, išsaugant priminio vartotojo signalo emisijų aptikimo tikslumą.

Tyrimų metodika

Disertacijoje yra išskirti du signalo analizės etapai: signalo požymių išskyrimas ir sprendimo apie signalo buvimą priėmimas. Signalo požymių išskyrimas atliktas taikant skaitmeninį signalų apdorojimą laiko ir dažnio srityse.

Sprendimui apie pirminio vartotojo buvimą priimti buvo tirti dirbtiniai neuronų tinklai (mokymas su mokytoju) ir saviorganizuojantys tinklai (apsimokymas be mokytojo).

Eksperimentiniai tyrimai atlikti panaudojant MATLAB™ ir Python programinius įrankius. Atliekant eksperimentus realiuoju laiku algoritmai įgyvendinti lauku programuojamoje loginėje matricoje – LPLM.

Darbo mokslinis naujumas

Rengiant disertaciją buvo gauti šie elektros ir elektronikos inžinerijos mokslui nauji rezultatai:

1. Spektro požymių išskyrimo algoritmų patobulinimai, efektyviam įgyvendinimui LPLM, išlaikant pirminio vartotojo signalo emisijų aptikimo radijo eteryje tikslumą.
2. Pasiūlytas spektro požymių iškyrimo būdas, grįstas vilnelių transformacija, kuris sumažina klaidos santykį 2–9 kartus, lyginant su signalo energijos vidurkiu, kvadratiniu ir standartiniu nuokrypiais.
3. Pasiūlytas sprendimo priėmimo būdas, grįstas Kohonen tinklais, kuris gali prisitaikyti prie skirtingų spektrinių savybių signalų, išsaugant 1% klaidos santykį, kuris nustatytas analizuojant ROC kreives.
4. Originalus Kohonen tinklo patobulinimas, kuriems reikia 32 kartus mažiau mokymosi iteracijų nei rekomenduojama literatūroje.
5. Pasiūlytas originalus Kohonen tinklo mokymosi galinio taško sekimo algoritmas, kuris sumažina mokymosi iteracijų kiekį 2,6–44,6%, lyginant su klasterio kokybės nustatymo būdu.

Darbo rezultatų praktinė reikšmė

Disertacijoje pasiūlyti nauji metodai pritaikyti įgyvendinimui praktikoje naudojamose įterptinėse sistemose. Metodai įgyvendinti MATLAB™ aplinkoje, o taip pat dauguma jų įgyvendinti LPLM sistemoje. Atlikti praktiniai eksperimentai leido apskaičiuoti radijo spektro užimtumą (dviejuose ruožuose) skirtingose vietovėse: didmiestyje, priemiestyje ir mažame miestelyje. Eksperimentinio tyrimo rezultatai parodė, kad 954 MHz dažnių ruože išnaudojama 38–74% ryšio kanalo pajėgumų, o 433 MHz dažnių ruože išnaudojama vos $8,6 \cdot 10^{-5}$–$6,275 \cdot 10^{-3}$% kanalo pajėgumų.

Ginamieji teiginiai

1. Spektro jutikliams taikant Daubechies vilnelių transformaciją, kai nėra pirminės informacijos, galima sumažinti aptikimo klaidos santykį iki mažiau nei 1%, išsaugant pilną pirminio vartotojo aptikimą, kai signalas triukšmas santykis didesnis nei 0,8 dB.

2. Spektro jutiklio prisitaikymas prie besikeičiančios radijo aplinkos gali būti pasiektas naudojant Kohonen tinklus, užtikrinant aptikimo klaidos santykį mažesnį nei 1% ir išsaugant pilną pirminio vartotojo aptikimą, kai signalas triukšmas santykis didesnis nei 0,8 dB.

3. Kohonen tinklų topologijos patobulinimai mokymosi fazėje padidina įgyvendinimo efektyvumą LPLM grįstose įterptinėse sistemose, ir sumažina mokymosi trukmę iki 32 kartų, lyginant su literatūroje rekomenduojama trukme.

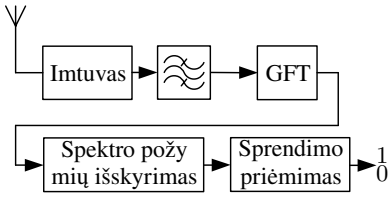
Disertacijos struktūra

Disertaciją sudaro: įvadas, trys skyriai, bendrosios išvados. Papildomai disertacijoje yra pateikti naudotų žymėjimų ir santrumpų sąrašai bei dalykinė rodyklė. Darbo apimtis yra 165 puslapiai, kuriuose yra pateikta: 45 formulės, 123 paveikslai ir 30 lentelių. Disertacijoje remtasi 157 kitų autorių literatūros šaltiniais.

1. Literatūros šaltinių apie spektro jutimo metodus apžvalga

Siekiant padidinti belaidžių ryšių sistemų aprėpties atstumą ir perduodamų duomenų spartą, įvairiuose tyrimuose ieškoma naujų radijo signalų perdavimo būdų. Viena iš tyrimų krypčių yra radijo spektro optimizavimas, naudojant kognityvinį radiją. Ši tyrimų kryptis remiasi radijo spektro stebėjimų rezultatais tankiai apgyvendintose miestų teritorijose, kuriose buvo nustatytas žemas radijo spektro užimtumas. Vidutinis radijo spektro panaudojimas 0,05–3 GHz ruože nesiekia 15%. Kognityvinio radijo paskirtis yra ieškoti pirminio vartotojo neužimtų radijo dažnių ruožų ir juos panaudoti arba paskirti antrinio vartotojo komunikacijoms.

Kognityvinis radijas susideda iš dviejų dalių: imtuvo ir signalų apdorojimo grandies. Nuo imtuvo parametrų priklauso kognityvinio radijo darbinis dažnių ruožas ir darbinė juosta. Nuo signalų apdorojimo grandies priklauso darbinės juostos spektro raiška, išskiriamų spektro požymių kokybė ir sprendimo apie spektro užimtumą tikslumas. Spektro požymių išskyrimas ir sprendimo apie spektro užimtumą priėmimas apibendrintai vadinimas spektro jutikliu.



1S pav. Kognityvinio radijo struktūra

Spektro jutikliai yra skirstomi į dvi grupes: jutikliai be išankstinių žinių apie pirminį vartotoją ir jutikliai su išankstinėmis žiniomis apie pirminį vartotoją. Pagrindinis skirtumas tarp šių jutiklių yra tas, kad antroji jutiklių grupė yra apmokyta/paruošta aptikti tam tikro tipo pirminio vartotojo signalus. Todėl spektro jutikliai su išankstinėmis žiniomis turi: arba turėti dideles pirminio vartotojo signalų parametrų duomenų bazes, arba dirbti siauruose dažnių ruožuose. Jutiklių be išankstinių žinių apie pirminį vartotoją darbui plačiuose dažnių ruožuose signalų parametrų duomenų bazės nereikalingos. Šį žinių praradimą galima kompensuoti naudojant Neuroninius arba Kohonen tinklus (angl. *Self Organizing Maps* – SOM). Siekiant sukurti spektro jutiklius, kurie galėtų dirbti plačiame dažnių ruože, disertacijoje tirti jutikliai be išankstinių žinių apie pirminį vartotoją, kuriuose sprendimo apie spektro užimtumą priėmimas grįstas intelektualiaisiais metodais.

Jutikliuose be išankstinių žinių plačiai naudojami spektro energija ir vilnelių transformacija grįsti spektro požymių išskyrimo algoritmai. Šie požymiai yra naudojami sprendimui apie spektro užimtumą priimti. Kaip spektro energijos požymiai gali būti naudojami: galios spektro vidurkis, kvadratinis nuokrypis arba standartinis nuokrypis. Galios spektro vidurkio skaičiavimas yra efektyvus, išskiriant signalus iš adityvinio Gauso triukšmo, tačiau jei triukšmas yra impulsinės arba nepastovios prigimties, geriau naudoti kvadratinio arba standartinio nuokrypio skaičiavimus. Literatūroje plačiai tyrinėjamos dar 3 vilnelės: Haaro, Daubechies ir Symlet, kurios taikomos radijo spektro požymiams išskirti. Naudojant šias vilneles galima ne tik efektyviai nuslopinti minėtus triukšmo tipus, bet ir gauti pirminio vartotojo signalo parametrus.

Priimti sprendimą apie kanalo užimtumą iš apskaičiuotų spektro požymių galima naudojantis intelektualiaisiais metodais, tokiais kaip neuroniniai ir Kohonen tinklai. Šie tinklai gali būti apmokyti (neuroniniai) arba apsimokyti (Kohonen) įvairioms radijo spektro situacijoms, taip sumažindami arba išvis panaikindami išankstinės informacijos apie radijo spektrą poreikį.

Literatūroje gausu įvairių teorinių tyrimų, kuriose vykdomi algoritmų ir metodų verifikavimai, modeliuojant radijo eterio kanalą. Tačiau yra didelis kognityvinio radijo

eksperimentinių tyrimų poreikis, kuriuose spektro jutikliai būtų išbandyti įvairiose radijo spektro situacijose, ir kuriuose aptikimo tikslumas būtų matuojamas įvairaus tipo signalams.

2. Spektro jutimo metodų teoriniai tyrimai

Svarbi kognityvinio radijo dalis yra radijo spektro jutiklis, kuris susideda iš požymių išskyrimo ir sprendimo priėmimo modulių. Todėl yra poreikis sukurti efektyviai veikiantį jutiklį, t.y. efektyvią spektro požymių išskyrimo ir sprendimų priėmimo porą. Kadangi ši pora turi veikti realiu laiku esant dideliam apdorojamų duomenų kiekiui, ji turi būti įgyvendinta lauku programuojamoje loginėje matricoje – LPLM. Jutiklio komponentų įgyvendinimo procese, be tikslumo ir greitaveikos, didelį dėmesį reikia skirti efektyviam LPLM resursų išnaudojimui.

1S lentelė. LPLM naudojami resursai įgyvendinant požymių išskyrimo modulius

Požymių išskyrimas	Registrai	Paieškos lentelės	Dauginimo elementai	Operatyvioji atmintis	Vėlinimas ciklai
Galios vidurkis	99	78	1	0	20
Kvadratinis nuokrypis	261	293	5	1	37
Kvadratinis nuokrypis be buferio	227	240	5	0	21
Standartinis nuokrypis	504	553	5	1	63
2 lygių Haar	291	554	0	0	32
2 lygių 4-8 vilnelės	1428	891	18	0	162
2 lygių 8-6 vilnelės	1428	943	20	0	192

Požymiams išskirti naudojami spektro energija ir vilnelių transformacija grįsti algoritmai. Šie požymių išskyrimo moduliai įgyvendinti LPLM. Jų naudojami LPLM resursai ir apdorojimo vėlinimai palyginti lentelėje. Modulių vėlinimai fiksuoti, kai kanalo plotį sudaro 16 spektro imčių.

Spektro energija grįstų požymių išskyrimo moduliams įgyvendinti naudotos paprastos loginės ir matematinės operacijos, tokios kaip akumuliavimas, postūmis, vėlinimas, sudėtis, atimtis ir daugyba. Tik standartinio nuokrypio skaičiavimui reikia papildomos kvadratinės šaknies traukimo operacijos. Mažiausias vėlinimas ir mažiausiai resursų naudota galios vidurkio skaičiavimo įgyvendinimui. Kvadratinio nuokrypio vėlinimas beveik dvigubai didesnis nei vidurkio skaičiavimo. Šis vėlinimas atsiranda, nes kvadratiniam nuokrypiui skaičiuoti reikalinga kanalo vidurkio vertė, todėl nuokrypis gali būti apskaičiuotas tik po kanalo vidurkio radimo. Vėlinimą galima sumažinti kvadratinio nuokrypio skaičiavimui naudojant kaimyninio kanalo vidurkio vertę (Kvad. nuokr. be B.). Tokiu atveju sumažinamas ne tik vėlinimas, bet eliminuojama

atmintis, naudota kanalo imčių saugojimui. Standartinio nuokrypio operacijai įgyvendinti naudota daugiausiai LPLM resursų iš visų spektro energija grįstų modulių. Tam turėjo įtakos šaknies traukimo operacijos, nes šiai operacijai reikia tiek LPLM resursų (loginių celių ir vėlinimo), kiek kvadratiniam nuokrypiui skaičiuoti.

Dviejų lygių Daubechies ir Symlet vilnelių transformacijos įgyvendintos kaip filtrų bankai, kai filtrų koeficientams naudojami vilnelių koeficientai. Šioms dviejų lygių transformacijoms įgyvendinti reikia daugiau nei 10 kartų resursų, negu reikia spektro galios vidurkio skaičiavimui, o vėlinimas yra daugiau nei 8 kartus ilgesnis. Dviejų lygių Haaro transformacijai įgyvendinti reikia $\sim$3 kartus mažiau LPLM resursų nei Daubechies ir Symlet, o vėlinimas yra tik 1,6 karto ilgesnis nei vidurkio operacijos. Pagrindinis Daubechies ir Symlet transformacijų trūkumas – didelis naudojamų DSP celių kiekis. Šių DSP celių kiekis LPLM yra mažas. Tačiau naudojant dviejų lygių vilneles gaunami net 4 radijo spektro požymiai.

Iš apskaičiuotų spektro požymių turi būti priimtas sprendimas apie kanale ar juostoje esantį-neesantį pirminio vartotojo signalą. Iš šių požymių, be išankstinių žinių, sprendimą apie pirminio vartotojo signalą gali priimti intelektualieji metodai. Tačiau tiek neuroninių, tiek Kohonen tinklų mokymo/apsimokymo procesai yra ilgi. Priklausomai nuo tinklo dydžio, šio proceso trukmė siekia 10–1000 tūkstančių mokymosi iteracijų. Kohonen tinklų atveju, pagal pagrindinę mokymosi taisyklę, kiekvienas neuronas turi sunaudoti mokymosi procese bent 500 iteracijų. Jei tinklo dydis yra 9×9, tai tinklas turi būti mokomas 40500 iteracijų. Norint optimizuoti Kohonen tinklų mokymosi trukmę, šiame skyriuje pasiūlyti du mokymosi proceso optimizavimo algoritmai ir tinklo struktūra su vidiniais svoriais.

Abu mokymosi optimizavimo algoritmai sutrumpina Kohonen tinklo mokymosi procesą, tik pirmasis nutraukia šį procesą kai yra tenkinamos tam tikros sąlygos, o antrasis atlieka pakeitimus šiame procese. Pirmo mokymosi algoritmo pavadinamas – mokymosi galinio taško sekimas, o antrojo – mokymasis su asistentu. Abu algoritmai yra taisyklių sistemos, kurios įsiterpia į mokymosi procesą ir priklausomai nuo rezultatų priima sprendimą.

Mokymosi galinio taško sekimo algoritmas, mokymosi proceso metu, seka laimėjusio neurono Euklido vidutinį atstumą tarp neurono svorių ir įėjimo vektoriaus. Ir jeigu tam tikrame slenkančio vidurkio lange Euklido atstumas yra mažesnis už pasvertą pradinį įvertį, mokymosi procesas yra stabdomas. Šis pradinis įvertis randamas mokymosi pradžioje, kai apskaičiuojamas pradinis tinklo pokytis Δ_{init} tam tikrame N_{wind} lange:

$$\theta_{\mathrm{end}} = \kappa\Delta_{\mathrm{init}} = \kappa\sqrt{\frac{1}{N_{\mathrm{wind}}}\sum_{n=0}^{N_{\mathrm{wind}}-1}\left(d_{n+1}^{\perp}(\boldsymbol{x}) - d_{n}^{\perp}(\boldsymbol{x})\right)^2}. \tag{1S}$$

Pokytis nustatomas tarp gretimų iteracijų laimėjusių neuronų Euklido atstumų $d_{n+1}^{\perp}(\boldsymbol{x})$ ir $d_{n}^{\perp}(\boldsymbol{x})$. N_{wind} lange yra surandamas šių pradinių pokyčių vidurkis. Šis vidurkis pasveriamas per κ, norint nustatyti mokymosi proceso nutraukymo slenkstį θ_{end}. κ parenkamas 0,2–0,25 rėžiuose t.y. jei vidutinis laimėjusio neurono Euklido atstumas yra daugiau nei 75–80% mažesnis nei Δ_{init} – mokymosi procesas sustabdomas.

Mokymosi su asistentu algoritmas šio proceso metu atlieka pakeitimus keisdamas mokymosi intensyvumą η ir esant labai prastam mokymosi rezultatui perkrauna neuronų svorius. Mokymosi intensyvumą η galima keisti keičiant tris parametrus: pradinį intensyvumą η_0, eksponentinės funkcijos nuolydžio koeficientą τ ir mokymosi intensyvumo funkcijos argumento reikšmę n:

$$\eta(n) = \eta_0 e^{-\frac{n}{\tau}}. \tag{2S}$$

Asistentas mokymosi patikrinimą atlieka periodiškai kas 400–600 mokymosi iteracijų. Sprendimą apie mokymosi progresą asistentas gali priimti iš tų pačių parametrų kaip ir galinio taško sekimo algoritmas: iš Δ_{init} ir vidutinio Euklido atstumo. Pagal šiuos parametrus Asistento algoritmas nustato 5 būsenas: Nepatenkinamas, Regresyvus, Progresyvus, Patenkinamas ir Stabilus mokymasis. Jei asistentas nustato, kad tinklo mokymosi procesas atitinka pirmąsias dvejas būsenas, atliekamas pilnas arba dalinis tinklo mokymosi proceso perkrovimas. Inicijuojami nauji svoriai, padidinama mokymosi intensyvumo vertė. Jei nustatoma, kad tinklo mokymosi rezultatai yra progresyvūs arba patenkinami, tai atliekamas mokymosi intensyvumo mažinimas. Esant stabiliai mokymosi būsenai, derinimas nėra atliekamas.

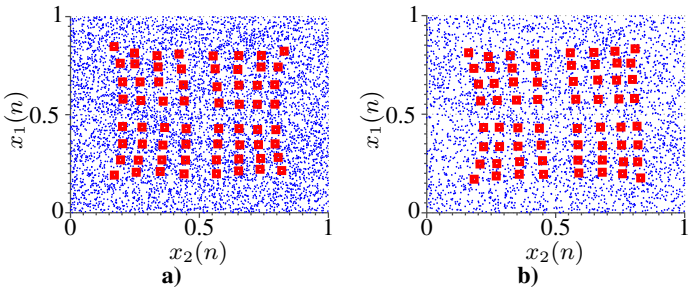
Abiejų pasiūlytų algoritmų apsimokymo trukmės palygintos su pagrindine mokymo taisykle ir klasterio kokybės nustatymu grįstų mokymo algoritmų trukmėmis, kai į įvairaus dydžio Kohonen tinklų įėjimus pateikiamas atsitiktinio triukšmo vektorius. Iš lentelės matyti, kad abu algoritmai, lyginant su pagrindine mokymo taisykle grįstu algoritmu, sumažina mokymosi trukmę iki 5–8 kartų. Ypač šie algoritmai efektyvūs su didesnėmis nei 2×2 tinklais.

2S lentelė. Mokymosi trukmės naudojant skirtingus mokymosi proceso nutraukimo algoritmus

Mokymosi algoritmas	2×2	4×4	6×6	8×8
Pagrindinė mokymosi taisyklė	2000	8000	16000	32000
Klasterio kokybės nustatymas	1921	3663	5517	6415
Δ_{init}	1627	3405	4267	6194
Asistentas	1761	1745	3283	3783

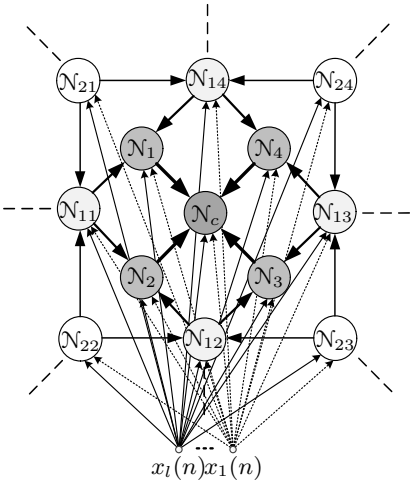
Apmokytų Kohonen tinklų, panaudojant abu pasiūlytus apsimokymo trukmės optimizavimo algoritmus, pavyzdžiai pateikti paveiksle. Nors ir mokymuisi su asistentu reikėjo beveik 2 kartus mažiau iteracijų nei mokymosi galinio taško sekimo algoritmui, tačiau tinklo struktūros kokybiškai panašios.

Kohonen tinklo struktūroje su vidiniais svoriais įvedami svoriai, kurie jungia gretimus neuronus. Taip įvedamas kaimynystės sąryšis tarp neuronų. Papildomi tarpneuroniniai svoriai nukreipti į centrą. Su išoriniais sluoksniais centrinis neuronas sujungtas per 4 neuronus, kurie su centriniu neuronu turi po vieną sąryšį. Išorinių sluoksnių neuronai turi po du sąryšius su arčiau centro esančio sluoksnio kaimyniniais neuronais. Todėl šios struktūros tinklo išėjimų skaičiavimas vykdomas iš kraštinių sluoksnių į centrinį neuroną. Išskyrus galinio tinklo sluoksnio neuronus, į visų neuronų išėjimų



2S pav. 8×8 Kohonen tinklai kai mokymosi procesas sustabdytas naudojant a) Δ_{init}; b) Asistentą

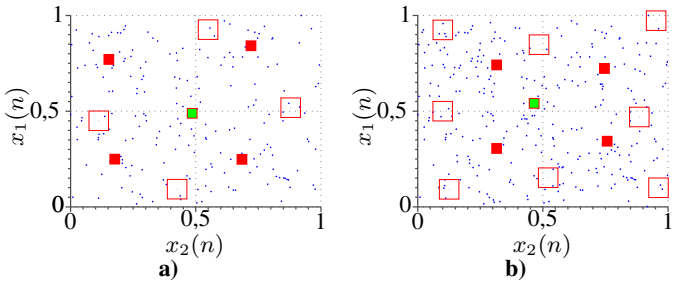
skaičiavimus įvedami kaimyninių neuronų pasverti išėjimai. Kadangi šioje struktūroje jau įvesti ryšiai tarp kaimynų, mokymosi proceso metu kaimynystės funkcijos netaikomos. Atnaujinami tik laimėjusio neurono įėjimo ir į jį nukreipti tarp-neuroniniai svoriai.



3S pav. Kohonen tinklo su vidiniais svoriais struktūra

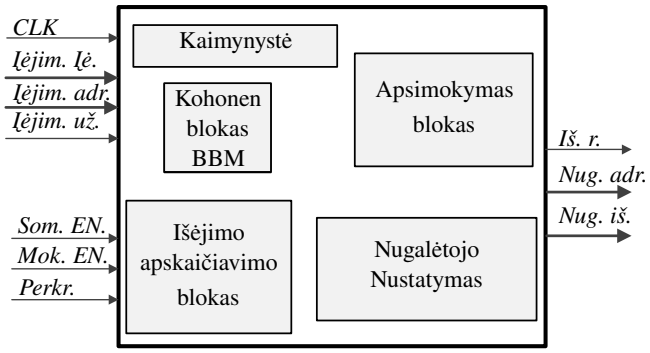
Kohonen tinklas su vidiniais ryšiais išbandytas taip pat kaip ir mokymosi optimizavimo algoritmai. Išbandytos 3 struktūros, kuriose naudotas skirtingas išorinių sluoksnių skaičius. Nustatyta, kad vidutiniškai kiekvienam neuronui apmokyti sunaudota 39 mokymosi iteracijos ir tai yra apie 13 kartų mažiau nei taikant pagrindinę mokymosį taisyklę. Kai tinklo dydis $9\mathcal{N}$ mokymosi trukmė 196 iteracijos, $13\mathcal{N}$ – 304 ir $17\mathcal{N}$ – 1026.

Paveiksle 4S pavaizduoti apsimokiusių struktūrų pavyzdžiai, kai tinkle buvo 1 ir 2 išoriniai sluoksniai. Šiose struktūrose centre pažymėtas centrinis neuronas, ryškiai - jungiamieji neuronai, kurie centrinį neuroną jungia su išoriniu sluoksniu.



4S pav. Kohonen su vidiniais svoriais apmokyti tinklai kai tinkle yra: a) 1 išorinis sluoksnis; b) 2 išoriniai sluoksniai

Atlikus Kohonen tinklų mokymosi ir struktūros optimizavimo tyrimus, įgyvendinta modulinė tinklo struktūra, kurią panaudojant įgyvendinami aptarti mokymosi algoritmai ir tinko struktūros patobulinimai. Šios struktūros valdymas atliekamas naudojantis baigtinių būvių mašina, kuri pagal įėjimo signalus nustato mokymosi ir išėjimo skaičiavimo procesų būsenas.



5S pav. Kohonen tinklo struktūros įgyvendinimas panaudojant baigtinių buvių mašiną

Panaudojus modulinę tinklo struktūrą įgyvendinti 16 neuronų Kohonen tinklai, kurie buvo aptarti tyrimuose. Pagrindiniai skirtumai tarp šių įgyvendinimų yra naudojamos atminties kiekis ir vėlinimas. Abu šie parametrai yra įtakojami naudojamų iteracijų kiekio. Apsimokymo vėlinimas tiesiogiai priklauso nuo iteracijų kiekio, o didesnis kiekis atminties reikalingas mokymosi intensyvumo vertėms saugoti. Kadangi mokymosi intensyvumas kinta eksponentiniu dėsniu, šio parametro alternatyvūs apskaičiavimo būdai arba pareikalautų kitų svarbių LPLM resursų, arba padidintų skaičiavimo vėlinimą.

Įgyvendinant spektro jutiklį Kohonen tinklo modulis turi būti apjungtas su požymių išskyrimo moduliu. Šiam apjungimui reikalinga papildoma atmintis, kuri būtų naudojama skirtingų duomenų spartų sinchronizavimui. Nes spektro požymių išskyrimo modulio vėlinimas yra mažesnis už sprendimo priėmimo procesą.

3S lentelė. LPLM naudojami resursai įgyvendinant Kohonen tinklų struktūras

SOM struktūros	Registrai	Paieškos lentelės	Dauginimo elementai	Operatyvioji atmintis	Vėlinimas ciklai
8000 mokymo iteracijų	344	624	2	14	5704000
Δ_{init}	405	654	2	9	2523105
Su asistentu	431	693	2	7	1298280
Su vidiniais svoriais	383	674	2	6	175951

3. Spektro jutimo metodų eksperimentiniai tyrimai

Eksperimentiniai tyrimai atlikti trimis etapais. Pirmieji du etapai vykdyti nerealiu laiku, kai buvo apdorojami radijo įrašai. Trečiajame etape atrinkti spektro jutikliai išbandyti realiame laike, kai matuotas radijo eterio aktyvumas įvairiose vietovėse ir dažnių ruožuose. Pirmąjam ir antrąjam etapams įrašyti keturi įrašai, kuriose matuoti keturių tipų signalai: plačiajuostis, siaurajuostis, su besikeičiančiu nešlio dažniu ir trumpų emisijų. Šie signalai pagal savo pobūdį yra charakteringi 0,050–3 GHz dažnių ruožui. Pirmajame etape tikrinta, kaip spektro požymių išskyrimo algoritmai išskiria įvairaus tipo signalus, o antrajame etape – kaip intelektualieji metodai tiksliai priima sprendimus apie radijo eterio užimtumą.

Pirmajame etape visiems spektro požymių išskyrimo algoritmams išmatuotos keturios darbinės imtuvų charakteristikos (kiekvienam signalo tipui). Šios charakteristikos matuotos šešiems spektro energija ir trims vilnelių transformacija grįstiems spektro požymių išskyrimo algoritmams. Bendras geriausias požymių išskyrimo tikslumas pasiektas naudojant Daubechies vilnelių transformaciją. Nes su šia transformacija išskirtiems požymiams, taikant slenksčio funkciją, pilnas pirminio vartotojų signalo emisijų išskirimas pasiektas, kai klaidos aptikimo santykis yra: 1,2% signalui su besikeičiančio nešlio dažniu, 1,4% trumpų emisijų, 0,2% plačiajuosčiam ir 0,65% siaurajuosčiam signalams. Kai taikytas kvadratinio nuokrypio skaičiavimu grįstas požymių išskirimas, mažesnis nei Daubechies klaidos aptikimo santykis nustatytas plačiajuosčiam 0,017% ir siaurajuosčiam 0,56% signalų tipams. O trumpų emisijų tipo signalai geriausiai išskirti taikant kvadratinio nuokrypio be papildomos atminties modulį. Su šiuo moduliu visos trumpų emisijų pirminio signalo emisijos išskirtos esant 1,1% klaidos santykiui. Dėl prastų darbinių imtuvų charakteristikų iš tolimesnių tyrimų pašalinti 3 spektro energija ir 1 vilnelių transformacija grįsti spektro požymių išskirimo algoritmai. O tolimesniuose tyrimuose naudoti spektro energijos vidurkio, kvadratinių nuokrypio, kvadratinio nuokrypio be atminties, Haaro ir Daubechies transformacijomis grįsti požymių išskyrimo algoritmai.

Antrajame eksperimentų etape tirtas pirminio vartotojo nustatymo tikslumas, taikant įvairius Kohonen ir neuroninius tinklus, kurių dydis keistas nuo 9 iki 45 neuronų. Eksperimentuose taikytas neuroninis tinklas, kuris apmokytas taikant atgalinio sklidimo algoritmą. O Kohonen tinklams taikyti trys apsimokymo algoritmai (pagrindinė mokymosi taisyklė, galinio mokymosi taško sekimas ir mokymasis su asistentu) su trejomis skirtingomis kaimynystės funkcijomis (kvadratas, rombas ir šešiakampis). Kohonen tinklų su vidiniais svoriais struktūrai taikytas laimėtojas pasiima viską mokymosi algoritmas, kai mokymosi procesas stabdytas naudojant galinio taško

sekimo algoritmą. Kiekvieno tinklo pirminio vartotojo nustatymo tikslumas matuotas keturiems signalų tipams, kai spektro požymiams išskirti taikyti spektro energija ir vilnelių transformacija grįsti algoritmai. Apibendrinus antrojo etapo eksperimentinius tyrimus, išskiriami šie tinklų charakteringi rezultatai:

- Neuroninis tinklas. Šiam tinklui apmokyti vidutiniškai naudotos $1,228 \cdot 10^5$ iteracijos. Pasiektas vidutinis pirminio vartotojo nustatymo tikslumas 98,63% su 2,72% vidutiniu klaidos santykiu. Tiksliausiai nustatytas pirminis vartotojas, kai neuroninis tinklas naudotas kartu su spektro galios vidurkio ir kvadratinio nuokrypio požymių išskyrimo moduliu.
- Pagrindine mokymosi taisykle grįstas Kohonen tinklas. Vidutinė šio tinklui apsimokymo proceso trukmė – $1,381 \cdot 10^4$ iteracijų. Vidutiniškai pirminio vartotojo emisijos nustatytos su 97,99% tikslumu, kai nustatyta 1,54% vidutinis klaidos santykis. Tiksliausiai pirminis vartotojas nustatytas naudojant Kohonen tinklą su rombo kaimynystės funkcija, kai spektro požymiai išskirti naudojant spektro galios vidurkio ir kvadratinio nuokrypio be atminties modulius.
- Galinio mokymosi taško sekimu grįstas Kohonen tinklas. Vidutinė mokymosi proceso trukmė – 3787 iteracijos. Pirminio vartotojo emisijos nustatytos su vidutiniu 99,1% tikslumu ir 1,85% vidutiniu klaidos santykiu. Geriausias tikslumas pasiektas naudojant Kohonen tinklą apsimokiusį pagal šešiakampę kaimynystės funkciją, kartu su vidurkio ir kvadratinio nuokrypio be atminties moduliu.
- Kohonen tinklas su mokymosi asistentu. Vidutiniškai mokymosi procese sunaudotos 2467 iteracijos. Pirminis vartotojas nustatytas vidutiniu 99,11% tikslumu ir 1,46% klaidos santykiu. Tiksliausi emisijų nustatymo rezultatai pasiekti naudojant Kohonen tinklą su rombo kaimynystės funkciją ir požymius, kurie išskirti naudojant spektro galio vidurkio bei kvadratinio nuokrypio be atminties modulius.
- Kohonen tinklas su vidiniais svoriais. Vidutinė mokymosi trukmė tik – 57 iteracijos. Pasiektas vidutinis pirminio vartotojo nustatymo tikslumas 98,43%, kai nustatytas 0,94% vidutinis klaidos santykis. Tiksliausiai pirminio vartotojo emisijos nustatytos naudojant Daubechies transformacija išskirtus spektro požymius.

Iš visų antrojo etapo tinklų rezultatų galima išskirti Kohonen tinklą su mokymosi asistentu ir Kohonen tinklą su vidiniais svoriais. Nes su mokymosi asistentu pasiektas didžiausias vidutinis pirminio vartotojo nustatymo tikslumas, o Kohonen tinklo su vidiniais svoriais mokymosi proceso trukmė mažiausia.

Iš pirmųjų etapų realaus laiko eksperimentams atrinkti tokie tinklai: pagrindine mokymosi taisykle grįstas Kohonen tinklas su rombo kaimynystės funkcija, galinio mokymosi taško sekimu grįstas Kohonen tinklas su šešiakampio kaimynystės funkcija, Kohonen tinklas su mokymosi asistentu - rombo kaimynystės funkcija ir Kohonen tinklas su vidiniais svoriais. Minėtiems tinklams požymiai išskiriami naudojant: spektro galios vidurkio, kvadratinio nuokrypio be atminties modulius ir Daubechies transformaciją. Realaus laiko eksperimentai atlikti trejose vietovėse, kuriuose stebėti du radijo dažnių ruožai. Vietovės parinktos pagal skirtingą gyventojų tankumą: miestas, priemiestis, miestelis. O radijo dažnių ruožai parinkti pagal galimą skirtingą pirminių

vartotojų signalų aktyvumą: nelicencijuotas ruožas 433 MHz centrinis dažnis 1 MHz juosta ir licencijuotas ruožas 954 MHz centrinis dažnis 4 MHz juosta. Kiekvienas radijo ruožas stebėtas po parą. Vienoje vietovėje eksperimentai vykdyti 2 paras, o bendra trečiojo etapo eksperimentų trukmė – 144 valandos. Panaudojant spektro jutiklius, sudarytus iš atrinktų Kohonen tinklų ir spektro požymių išskyrimo modulių, realaus laiko eksperimentuose nustatyta:

- Visose vietovėse 433 MHz dažnių ruožo užimtumas yra žemas. Vartotojų aktyvumas nesiekia 0,1%. Šis ruožas itin laisvas priemiestyje, kur vartotojų aktyvumas – $6,257 \cdot 10^{-3}\%$.
- 954 MHz dažnių ruožo užimtumas svyruoja nuo 38,47% (miestelyje) iki 74,45% (priemiestyje). Miesto teritorijoje šio ruožo užimtumas siekia 52,81%. Šis dažnių ruožas, lyginant su nelicencijuotu ruožu, yra efektyviai išnaudojamas, ypač priemiesčio teritorijoje.

Iš atliktų radijo eterio eksperimentinių stebėjimų pastebėta, kad visose tirtose vietovėse nelicencijuoto ruožo vartotojų skaičius gali būti plečiamas, o 954 MHz ruože vartotojų skaičius gali būti padidintas miestelyje ir mieste.

Bendrosios išvados

Disertacijoje patvirtintos abi iškeltos hipotezės. Taip pat disertacijoje pasiūlyti Kohonen tinklų mokymosi pakeitimai, kurie sumažina LPLM naudojamų resursų kiekį ir signalo apdorojimo vėlinimą.

1. Spektro požymių išskyrimas naudojant Daubechies vilnelių transformaciją padidina spektro jutiklio jautrumą ir sumažina vidutinį aptikimo klaidos santykį visiems signalų tipams nuo 1,94% iki 1%.
2. Taikant daugisluoksniu percetroniniu tinklu grįstą spektro jutiklį, galima aptikti iki 98,63% pirminio naudotojo signalų, kai klaidingai aptiktų signalų yra ne daugiau 2,72%.
3. Taikant Kohonen neuronų tinklu grįstą spektro jutiklį galima aptikti visus pirminio naudotojo signalus, kai klaidingai aptiktų signalų yra ne daugiau 1%.
4. Taikant naujai pasiūlytą mokymo sustabdymo algoritmą, grįstą tinklo laimėjusio neurono vidutinio atstumo stebėjimu, galima 19–80% sumažinti mokymo iteracijų skaičių, o papildžius algoritmą pasiūlytu tinklo mokymosi asistentu, tinklui apsimokyti pakanka 10% iteracijų, lyginant su rekomenduojamu literatūroje iteracijų skaičiumi.
5. Taikant naujai pasiūlytą Kohonen neuronų tinklo struktūrą su vidiniais ryšiais, pakanka 3–15% tinklo mokymosi iteracijų, lyginant su rekomenduojamu literatūroje iteracijų skaičiumi, o įgyvendinant LPLM sistemoje naudoja 2,3 karto mažiau atminties ir mokymosi greitis padidėja 32 kartus.

Annexes

Annex A. Results of spectrum sensors based on different self organizing maps structures offline experiments

Annex B. The copies of author scientific publications in dissertation theme

Annex C. The co-authors agreement to present publications material in the dissertation

Liudas STAŠIONIS

DEVELOPMENT OF SELF-ORGANIZING METHODS FOR
RADIO SPECTRUM SENSING

Doctoral Dissertation

Technological Sciences,
Electrical and Electronic Engineering (01T)

SAVIORGANIZUOJANČIŲ METODŲ RADIJO DAŽNIŲ
JUOSTOS UŽIMTUMUI STEBĖTI KŪRIMAS

Daktaro disertacija

Technologijos mokslai,
elektros ir elektronikos inžinerija (01T)

2016 03 25. 15,0 sp. l. Tiražas 20 egz.
Vilniaus Gedimino technikos universiteto leidykla „Technika“,
Saulėtekio al. 11, LT-10223 Vilnius, *http://leidykla.vgtu.lt*
Spausdino UAB „BMK leidykla“,
J. Jasinskio g. 16, LT-01112 Vilnius, *http://www.bmkleidykla.lt*.