Andrejs Bondarenko was born on 28 January 1982 in Riga, Latvia. In 2004, he obtained a Bachelor’s degree in Computer Science from the University of Latvia, and in 2006 a Master’s degree in Computer Science from Transport and Telecommunication Institute. Since 2010, he has been a student of the academic doctoral study programme “Information Technology” at Riga Technical University. He has worked at the computing centre of Oncology Centre of Latvia, where he developed internal information systems. Since 2006, he has worked at “CTCo” Ltd where he participated in many international IT projects mainly in reinsurance and catastrophe modelling. Last three years he has been a Lead Data Scientist responsible for text, image and structured data analysis. In Riga Technical University he has participated in the following projects: “Simulation and computational intelligence methods for logistics and e-business optimization” (2010–2012); “Fast and non-contact optical estimation of micro-organisms activity” (2018–2020); and “Skin cancer early diagnostics accuracy improvement by using neural networks” (2018–2020). His scientific interests are artificial neural networks and general artificial intelligence.
DEVELOPMENT OF KNOWLEDGE EXTRACTION METHODOLOGY FROM TRAINED ARTIFICIAL NEURAL NETWORKS

Summary of the Doctoral Thesis

Scientific supervisors:

Professor Dr. habil. sc. comp.
ARKADIJS BORISOVS

Professor Dr. sc. ing.
LUDMILA ALEKSEJEVA

RTU Press
Riga 2020

Published in accordance with the decision of the Institute of Information Technology Council of 25 November 2019, Minutes No. 12100-2/8.
DOCTORAL THESIS PROPOSED TO RIGA TECHNICAL UNIVERSITY FOR THE PROMOTION TO THE SCIENTIFIC DEGREE OF DOCTOR OF SCIENCES

To be granted the scientific degree of Doctor of Sciences (Ph. D.), the present Doctoral Thesis has been submitted for the defence at the open meeting of RTU Promotion Council on 18 May 2020 at the Faculty of Computer Science and Information Technology of Riga Technical University, 1 Setas Street, Room 202.

OFFICIAL REVIEWERS

Professor Dr. habil. sc. ing. Jānis Grundspeņķis
Riga Technical University

Professor Dr. sc. ing. Pēteris Grabusts
Rezekne Academy of Technologies, Latvia

Professor Dr. habil. sc. ing. Yevgeniy Bodyanskiy
Kharkiv National University of Radio Electronics, Ukraine

DECLARATION OF ACADEMIC INTEGRITY

I hereby declare that the Doctoral Thesis submitted for the review to Riga Technical University for the promotion to the scientific degree of Doctor of Sciences (Ph. D.) is my own. I confirm that this Doctoral Thesis had not been submitted to any other university for the promotion to a scientific degree.

Andrejs Bondarenko ................................ (signature)
Date: ............................

The Doctoral Thesis has been written in English. It consists of an introduction, 5 chapters, conclusions, 37 figures, 22 tables, 5 appendices; the total number of pages is 158, including appendices. The Bibliography contains 150 titles.
# TABLE OF CONTENTS

**GENERAL DESCRIPTION OF THE WORK** ................................................................. 5  
**Introduction** ........................................................................................................... 5  
**Topicality** ............................................................................................................. 5  
**Research Aim and Tasks** ...................................................................................... 5  
**Research Object and Subject** ................................................................................ 6  
**Research Hypotheses** ........................................................................................... 6  
**Research Methods** ............................................................................................... 6  
**Scientific Novelty of the Thesis** ........................................................................... 6  
**Practical Significance of Thesis** .......................................................................... 7  
**Approbation** .......................................................................................................... 7  
**Structure and Content of the Thesis** ..................................................................... 10  

**SUMMARY OF THESIS CHAPTERS** .......................................................................... 11  

1. **NEURAL NETWORKS AND KNOWLEDGE EXTRACTION** ........................................... 11  
   1.1. **Neural Networks and Knowledge Extraction Overview** .................................. 11  
   1.2. **Types of Knowledge Extraction Methods** ....................................................... 12  

2. **NEURAL NETWORK PRUNING** ............................................................................ 13  
   2.1. **Sensitivity-Based Pruning Algorithm** ............................................................... 13  
   2.2. **Validation of the Developed Pruning Algorithm** ............................................ 14  

3. **DECISION TREE EXTRACTION FROM MULTILAYER PERCEPTRON** ................. 16  
   3.1. **Knowledge Extraction Overview** .................................................................. 16  
   3.2. **Knowledge Extraction Algorithm** ................................................................. 17  
   3.3. **Developed Algorithm Validation** ..................................................................... 18  

4. **OPTIMIZATION-BASED METHODS FOR RULE EXTRACTION** ............................. 21  
   4.1. **Elliptical Rules Extraction From RBF Neural Networks** ............................... 21  
      **Optimization Problem** ....................................................................................... 21  
      **Experiments and Results** ............................................................................... 22  
   4.2. **Rules Extraction Using a Piece-Wise Approximation Algorithm** .................... 23  
      **Overview of the Approach** .............................................................................. 23  
      **If–Then Rules Extraction Algorithm** ............................................................... 23  
      **Experiments and Results** ............................................................................... 24  

5. **KNOWLEDGE EXTRACTION METHODOLOGY** .................................................... 26  
   5.1. **Methodology Development** ......................................................................... 26  
   5.2. **Precise vs. Comprehensive Rules** .................................................................. 28  

**RESULTS AND CONCLUSIONS** ............................................................................. 31  
**BIBLIOGRAPHY** .................................................................................................... 33
GENERAL DESCRIPTION OF THE WORK

Introduction

Artificial neural networks (ANN) are widely used in machine learning. They are powerful non-linear models that can be trained in a supervised, semi-supervised, and unsupervised manner. There is no single best machine learning classifier that can be used in all scenarios, but ANNs are frequently outperforming other classifiers. On the downside, it is hard to explain how classification decision is made within ANN. Artificial neural networks are essentially black-boxes. Lack of understanding of how such classifiers work severely limits their applicability. The Thesis is devoted to the development of approaches allowing to extract knowledge in the form of rules from trained ANN classifier.

Topicality

Comprehensibility of the classification model is a crucial requirement in mission-critical domain areas like nuclear power, medicine, finance, and others. Additionally there could be law requirements, like the European Union GDPR 2018 law [118] stating that all life-changing algorithmic decisions should be explainable. Explainability allows ensuring there are no classification biases and discrimination and can generate new knowledge. There exist publications in the knowledge extraction domain, but no ready to use algorithms are available. In addition, as it was discovered reproducibility is a huge problem, thus the development of tooling for explaining ANN classifiers can significantly improve their usability.

Research Aim and Tasks

The research aim is to develop algorithms for pruning and knowledge extraction (KE) from trained ANN and unify them into knowledge extraction methodology. Such methodology should allow representing trained feedforward neural networks as an If–Then rules set, as a binary classification tree or set of equation rules. Research tasks to be solved to accomplish the stated research aim are the following.

1. To review and analyse existing knowledge representation and extraction approaches described in scientific literature addressing the same problem.
2. To study artificial neural networks pruning methods, their pros and cons, develop an improved method and evaluate it.
3. To develop, implement and evaluate approaches, which allow performing knowledge extraction from trained multilayer perceptron.
4. To develop and assess optimization-based methods for If–Then and elliptical rules extraction from trained piece-wise linear classifier and RBFNN.
5. To develop a generalized methodology for knowledge extraction from ANN.
Research Object and Subject

The research object is an explanation of trained artificial neural network classification decision, research subject – machine learning and specifically knowledge extraction approaches.

Research Hypotheses

During research and development of ANN pruning techniques and rules extraction methods, the following hypotheses were defined.

1. Improved sensitivity-based pruning algorithm successfully escapes local minimums and controls classification error rise.
2. Discreet input space subdivision acquired from MLP neurons outputs can be used to build a classification decision tree with controllable classification precision.
3. If–Then rules acquired via posing and solving the convex optimization problem allow approximate input space regions bounded by hyperplanes.
4. Elliptical rules extracted from feedforward radial basis function neural network by solving a non-convex optimization problem allow to approximate original RBFNN.

Research Methods

The study is based on mathematical and statistical analysis, machine learning, optimization theory, and experimental research methodologies. Literature review and analysis are used as well to gather information about the existing approaches in the subject domain area.

Scientific Novelty of the Thesis

The scientific novelty of the study is based on reviewing the existing and developing methods for knowledge extraction. Which, in turn, holds four specifically developed methods, which can be applied whenever model understanding and knowledge in explicit form is required. The scientific novelty and achievements are listed below.

1. A sensitivity-based artificial neural network pruning algorithm developed with several modifications allowing it to escape local minimums. A comparison of weights versus nodes pruning performed with recommendations on approach selection depending on requirements.
2. An implemented method for binary classification decision tree extraction from trained feedforward ANN multi-class classifier. An experimental testing of the proposed solution was performed.
3. An approach developed for extraction of If–Then rules from piece-wise linear approximation of a non-linear classifier. This approach allows rules extraction from a set of hyperplanes defined in input space. Although the approach has shown itself as
prone to curse of dimensionality, it can be utilized for datasets with a small amount of input data dimensions.

4. An approach developed for extraction of elliptical rules from two-dimensional or three-dimensional RBF neural network. Although this approach has shown itself prone to the curse of dimensionality developed optimization problem posed as the right approach for elliptical rules extraction in case of two or three dimensions. Larger dimensions counts can be supported via algorithmic enhancement.

5. Based on the conducted research and experiments developed the methodology for utilization of rules extraction approaches, listed above, with recommendations on cases when one approach should be selected over the other.

**Practical Significance of Thesis**

The practical significance lies in programmatic realizations, experimental validation and assessment of the discussed methods. Full list of practical achievements is as follows.

1. Performed review and comparison of knowledge representation schemas, and recommendations are given for scheme selection.
2. In the scope of developed methodology, recommendations allowing extraction of accurate or comprehensible rules are given.
3. Recommendations for using nodes or weights pruning are given, allowing to rise generalization of ANN.
4. Applicability of reviewed classifiers (MLP, RBFNN, Piece-Wise linear classifier) has been improved, as now it becomes possible to validate them, understand model classification decision, and discover new knowledge.
5. Programmatic realizations have been created in Matlab, extensions of Lua-based Torch7 deep learning (DL) framework and Python-based PyTorch DL framework. PyTorch version is applicable to medium-sized datasets.

**Approbation**

Research results were presented at thirteen international scientific conferences.

1. RTU 60th International Scientific Conference. Latvia, Riga, 10–11 October 2019.

Research results that served as the basis for the Thesis were published in the following scientific papers.


**The results of the Doctoral Thesis research have been used in the following projects.**


**Structure and Content of the Thesis**

The Doctoral Thesis contains an introduction, five main chapters, results analysis, and conclusions.  

The introduction validates the topicality of the conducted investigations, formulates the object, the aim, and research tasks. It describes scientific novelty, as well as briefly characterizes basic directions of the research performed.  

Chapter 1 describes the problem area, which is artificial neural networks. A short introduction is given along with brief descriptions of the main well-known ANN types.  

Chapter 2 describes in detail the initial step required for knowledge extraction from a trained artificial neural network – network pruning. This chapter covers several existing algorithms along with their evaluations and comparison to the developed pruning algorithm.  

Chapter 3 presents the developed approach for the extraction of binary classification decision tree from a trained multilayer perceptron. This chapter describes the implemented algorithm, provides its pseudocode, and holds algorithm evaluation.  

Chapter 4 presents the developed optimization-based methods for oblique (If–Then) and equation rules extraction from a piece-wise linear classifier and RBFNN.  

Chapter 5 describes the developed methodology for choosing one of the described methods over others.  

Results and Conclusions chapter recaps the aim, tasks, and hypotheses, makes conclusions, covers scientific and practical novelty of the Thesis, and discusses future research directions.
SUMMARY OF THESIS CHAPTERS

1. NEURAL NETWORKS AND KNOWLEDGE EXTRACTION

Based on the posed aim and tasks, the first chapter performs an overview of the machine-learning field and artificial neural networks (ANN) as a research object. This chapter covers ANN types, their training and usage. Knowledge extraction (KE) algorithms and knowledge representation forms are reviewed.

1.1. Neural Networks and Knowledge Extraction Overview

Look into [90] for the in-depth introduction to biological and artificial neural networks. Authors of [40] posed a plausible explanation describing how neural networks can operate and approximate simple linear functions. Later, researchers inspired by biological neural networks have proposed different artificial neural network architectures.

Data itself represent little value. Information on the other side is data bound into a specific context, which gives some meaning to the data and allows us to see relations. Knowledge is information, which is organized in a way that allows seeing specific patterns. Wisdom is even more abstract and allows an understanding of general principles. In the scope of general AI systems, knowledge had to be represented in such a way that specific reasoning could be applied over it. Knowledge is the level covered by current research, it uses the definition for “rule extraction” term proposed in [73]. Given definition is broader than others like one given in [96] and underlines the fact that extracted rules can take different forms, not only lexical:

“Given an opaque predictive model and the data on which it was trained, produce a description of the predictive model’s hypothesis that is understandable yet closely approximates the predictive model’s behaviour.”

Depending on the context, rule extraction algorithms can be tuned to produce either more comprehensible, and hence compact and understandable, or more accurate rules.

According to the knowledge extraction taxonomy proposed in [76], there exist three types of knowledge extraction algorithms. The fourth knowledge extraction algorithm family (compositional) was proposed in [139]. These types are decompositional, pedagogical, eclectic, and compositional.

Depending on the use-case, the decision regarding the trade-off between readability and high classification rate should be made. This decision will allow selecting the most appropriate rules type. Types of rules (see detailed description in [22], [73]) are: propositional If–Then / If–Then–Else rules; M of N rules; Oblique rules / Equational rules; and Fuzzy rules.

If–Then rules and Decision trees are most welcome as they are easily embeddable, have good comprehensibility and acceptable expressiveness and compactness. Equational and oblique rules are most expressive while least interpretable by a human expert. The last two
groups – Fuzzy rules and M of N rules – are of less interest as they either bound to fuzzy neural networks, which are less common and are not easily embeddable, or have low expressive power. The current Thesis is focused on If–Then rules and binary classification decision tree extraction from MLP, as well as on extraction of elliptical rules from RBFNN.

As it was noted in [49], to get compact FFNN, there exist several approaches. The current Thesis is concentrating on the pruning approach. In case one does not have a trained neural network, the same pruning approach can be used for training an overly complex network with subsequent pruning to remove unnecessary neurons. Lowering the number of neurons can result in a smaller number of rules. The pruning algorithm developed in the scope of the Thesis is covered in Chapter 2.

1.2. Types of Knowledge Extraction Methods

There are four main types of KE algorithms, their strengths and weaknesses are summarized in Table 1.1. The current Thesis concentrates on decompositional and compositional KE algorithms. Decompositional algorithms, as it was shown in [39], are performing better than pedagogical approaches. Classification accuracy, portability and ability to influence the complexity and precision of extracted rules were selected as the most important properties.

The pedagogical approach is applied to RBFNN to extract Elliptical rules due to their high expressive power. For a decompositional approach If–Then rules and classification decision tree were chosen as knowledge representation forms to be explored. They are most commonly widespread, can be easily embedded into any existing information system and have an “embedded” inference engine, while being easily understandable.

<table>
<thead>
<tr>
<th>Property</th>
<th>Decompositional</th>
<th>Eclectic</th>
<th>Pedagogical</th>
<th>Compositional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification accuracy</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>n/a</td>
</tr>
<tr>
<td>Portability (not specific to classifier)</td>
<td>−</td>
<td>+</td>
<td>−</td>
<td>−</td>
</tr>
<tr>
<td>Tunability</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>Algorithm consistency (several runs − same result)</td>
<td>−</td>
<td>+</td>
<td>−</td>
<td>n/a</td>
</tr>
<tr>
<td>Speed</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>Knowledge representation variety</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>Scalability (Big data)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>n/a</td>
</tr>
<tr>
<td>Algorithm complexity (computational)</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

The chapter provides a general overview of the ML field and ANN generalization theory. Knowledge representation schemas and typical knowledge extraction workflow are covered. Task number one of the stated research tasks is accomplished.
2. NEURAL NETWORK PRUNING

It is known that neural networks with a smaller number of neurons are easier to extract knowledge from, and extracted knowledge is of smaller complexity. The pruning step is always welcome as it can positively influence ANN generalization abilities. In addition, pruning controls the comprehensibility and accuracy of extracted rules by controlling the number of neurons to be processed. Papers [3], [6], [25] give an overview of pruning algorithms. To select the pruning approach, based on a literature review a summary Table 2.1 was created using the scale from zero to five (higher is better).

<table>
<thead>
<tr>
<th>Criteria / Pruning type</th>
<th>Sensitivity based</th>
<th>Sensitivity analysis II OBD/OBS</th>
<th>Magnitude based</th>
<th>Weight decay</th>
<th>Mutual information-based</th>
<th>Significance based</th>
<th>Interactive pruning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplicity</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Execution time</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Memory footprint</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>No special training procedure</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Classification precision / generalization</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>n/a</td>
<td>n/a</td>
<td>3</td>
<td>n/a</td>
</tr>
<tr>
<td>Pruned neurons / weights count</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>n/a</td>
<td>n/a</td>
<td>4</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Sensitivity-based pruning approach was chosen due to its simplicity and good reported performance. Its idea is to remove single neuron and assess the ANN performance change, thus the least sensitive (important) neuron can be found and removed.

2.1. Sensitivity-Based Pruning Algorithm

Experiments with basic sensitivity-based pruning approach have shown that it is prone to local minimums, thus improved algorithm version has been developed. To escape local minimums, the algorithm [26] has been equipped with three improvements. The main steps of the developed algorithm are:

1) save the ANN state;
2) determine the least sensitive neuron (or weight), remove it;
3) retrain ANN;
4) assess classification performance degradation; if it is acceptable, continue pruning, otherwise rollback last neuron (or weight) removal; if several consecutive rollbacks have occurred (threshold is reached), then stop pruning and return the last best known saved ANN.
Steps 1, 3 and 4 are the improvements proposed within the scope of the Thesis. They allowed to prune more neurons (or weights) and control classification degradation. To validate the proposed improvements, a pruning process visualization was created, see Figure 2.1

Fig. 2.1. Neuron pruning experiments for the Ionosphere dataset.

### 2.2. Validation of the Developed Pruning Algorithm

To validate the developed algorithm, an experiments plan has been developed. The goal was to validate the developed algorithm and compare neurons vs. weights pruning. In the first experiments series (Table 2.2), nodes pruning algorithm was applied to neurons in hidden layers only. It is seen, that in all but three cases the testing error of pruned ANN was smaller than that of unpruned ANN. The developed pruning algorithm has shown strong performance both in terms of ANN simplification and its generalization improvement.

#### Results of Pruning Experiments

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MLP train avg.</th>
<th>MLP test avg.</th>
<th>Pruned train</th>
<th>Pruned test avg.</th>
<th>Architecture before/after pruning (hidden nodes in 2 hidden layers)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ionosphere</td>
<td>10.83 %</td>
<td>10.83 %</td>
<td>5.39 %</td>
<td>10.44 %</td>
<td>15–15 / 5.4–3.8</td>
</tr>
<tr>
<td>Monks-1</td>
<td>20.16 %</td>
<td>29.68 %</td>
<td>18.47 %</td>
<td>24.35 %</td>
<td>15–15 / 5–3</td>
</tr>
<tr>
<td>Monks-2</td>
<td>36.82 %</td>
<td>36.55 %</td>
<td>31.83 %</td>
<td>32.58 %</td>
<td>15–10 / 5–3.1</td>
</tr>
<tr>
<td>Monks-3</td>
<td>6.64 %</td>
<td>2.80 %</td>
<td>5.98 %</td>
<td>2.85 %</td>
<td>15–15 / 1.7–1.1</td>
</tr>
<tr>
<td>WPBC</td>
<td>0.00 %</td>
<td>0.00 %</td>
<td>0.00 %</td>
<td>0.00 %</td>
<td>10–10 / 1–1</td>
</tr>
<tr>
<td>WDBC</td>
<td>3.89 %</td>
<td>4.04 %</td>
<td>3.03 %</td>
<td>3.69 %</td>
<td>30–30 / 23.2–17.1</td>
</tr>
<tr>
<td>Pima</td>
<td>23.02 %</td>
<td>23.56 %</td>
<td>25.94 %</td>
<td>26.81 %</td>
<td>10–10 / 2.8–3.1</td>
</tr>
<tr>
<td>Haberman</td>
<td>26.13 %</td>
<td>26.57 %</td>
<td>28.50 %</td>
<td>28.20 %</td>
<td>15–15 / 2.3–3.7</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>24.62 %</td>
<td>24.61 %</td>
<td>16.29 %</td>
<td>15.82 %</td>
<td>30–30 / 26.8–28</td>
</tr>
</tbody>
</table>
Experiments were conducted to validate the developed algorithm in application to nodes and weights pruning [25] applied to the input and hidden neurons, see Table 2.3 for results.

In Table 2.2 and Table 2.3 one can see that pruned ANN in two cases has a minor rise in classification error and in the majority of cases its error rate is lowered. In four cases (Table 2.3), weights pruning was better than nodes pruning. In the remaining five experiment sets, nodes pruning proved to be a better option. Error after weights pruning on the Monks-1 dataset is 1.81% in contrast to 13.22% after nodes pruning, this is the only case with such drastic difference.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MLP train avg.</th>
<th>MLP test avg.</th>
<th>Pruned weights train avg.</th>
<th>Pruned weights test avg.</th>
<th>Pruned nodes train avg.</th>
<th>Pruned nodes test avg.</th>
<th>Pruned weights / pruned nodes counts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haberman</td>
<td>25.99 %</td>
<td>26.78 %</td>
<td>24.39 %</td>
<td>24.91 %</td>
<td>24.98 %</td>
<td>26.17 %</td>
<td>54.9/23.8</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>10.83 %</td>
<td>10.83 %</td>
<td>4.21 %</td>
<td>10.25 %</td>
<td>4.55 %</td>
<td>9.22 %</td>
<td>34.1/34.3</td>
</tr>
<tr>
<td>Monks-1</td>
<td>21.51 %</td>
<td>32.74 %</td>
<td>0.83 %</td>
<td>1.81 %</td>
<td>6.83 %</td>
<td>13.22 %</td>
<td>45.9/22.4</td>
</tr>
<tr>
<td>Monks-2</td>
<td>38.46 %</td>
<td>36.04 %</td>
<td>12.47 %</td>
<td>12.21 %</td>
<td>11.26 %</td>
<td>10.25 %</td>
<td>16.8/20.1</td>
</tr>
<tr>
<td>Monks-3</td>
<td>6.56 %</td>
<td>2.88 %</td>
<td>5.16 %</td>
<td>3.45 %</td>
<td>3.33 %</td>
<td>5.76 %</td>
<td>32.4/29.3</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>24.58 %</td>
<td>24.61 %</td>
<td>14.83 %</td>
<td>16.38 %</td>
<td>14.30 %</td>
<td>15.57 %</td>
<td>10.5/8.3</td>
</tr>
<tr>
<td>Pima</td>
<td>23.93 %</td>
<td>24.56 %</td>
<td>21.64 %</td>
<td>23.74 %</td>
<td>22.12 %</td>
<td>23.05 %</td>
<td>56.0/22.7</td>
</tr>
<tr>
<td>WDBC</td>
<td>4.16 %</td>
<td>4.33 %</td>
<td>1.83 %</td>
<td>2.63 %</td>
<td>1.77 %</td>
<td>2.93 %</td>
<td>23.3/18.5</td>
</tr>
<tr>
<td>WPBC</td>
<td>0.00 %</td>
<td>0.00 %</td>
<td>0.00 %</td>
<td>0.17 %</td>
<td>0.00 %</td>
<td>0 %</td>
<td>153.6/50.0</td>
</tr>
</tbody>
</table>

Pruning has been shown to be a useful trained ANN generalization improving step. The overall current chapter contributions are listed below.

- Developed pruning algorithm is presented, it is based on sensitivity measure with retraining, metric worsening threshold, and pocket memory, allowing pruning procedure to successfully escape local minimums.
- Conducted experiments have proven algorithm utility in simplifying the neural net structure and rising its generalization abilities.
- Input and hidden layer neurons pruning have shown lower classification errors, when compared to only hidden layers pruning.
- Experiments have shown that in general case nodes pruning is preferable over weights pruning as it requires a smaller amount of computations. The only exception would be a necessity to get the lowest possible error rate in which case weights pruning should be applied.

The current chapter accomplished task number two – the study of ANNs pruning approaches. The pruning approach was selected and based on the developed pruning algorithm.
3. DECISION TREE EXTRACTION FROM MULTILAYER PERCEPTRON

While frequently outperforming other types of classifiers artificial neural networks (ANN) black-box nature limit their usage. Therefore, knowledge extraction (KE) from trained ANN can help to uncover new knowledge, validate and productionalize or embed ANN-based classifiers.

3.1. Knowledge Extraction Overview

Description of various ANN types and architectures can be found in [12], [38], [56], [70], [92], [98], [120]. Due to the widespread usage of fully connected neuron layers, the current chapter concentrates on knowledge extraction from FFNNs. Non-linearity introduced by hidden layers is what makes FFNN flexible in modelling input data, but hard to understand classification decision. R. Setiono [96] was one of the first researchers who proposed to work not with weights of the neural network, but with neuron ‘statistics’ – neurons output values obtained over the training data set. As a preliminary KE step, it is proposed to prune neurons to minimize the number of neurons output values sets to be processed (see Chapter 2) – this potentially can minimize the number of ‘rules’ that will be extracted. The next stage involves neurons output values discretization to find bounds or, in the case of several neurons, regions from input space where all input vectors belonging to that region are classified as belonging to the same class. Later on, the developed algorithm (Fig. 3.1) can be applied to extract a classification decision tree from such discretized neurons.

![Fig. 3.1. The high-level knowledge extraction process.](image)

The developed algorithm is decompositional as it is using intrinsic knowledge about neurons outputs in the neural network. Due to the knowledge extraction algorithm specifics a binary classification decision tree was chosen as a rules representation form. Its visualization allows for straightforward reasoning about the classification process. Decision trees can be
directly, in a quazi-optimal way, mapped into If–Then rules as well.

Figure 3.1 provides a high-level overview of the knowledge extraction routine, along with required and optional pre-processing steps. Steps to which the current Thesis contributes to the development of new algorithms are highlighted.

In cases when the input data amount is prohibitively large, it is possible to: 1) work with the subset of the data; and 2) perform neurons output values discretization. After output values discretization step, output values clusterization needs to be performed that will further shrink possible neuron output values.

### 3.2. Knowledge Extraction Algorithm

Many of the KE algorithms described in the literature are extending the original NeuroRule algorithm [68]. NeuroRule and its derivatives have a very common workflow. The steps are neurons output signals non-linearity break-up via outputs clusterization (built up of quantization tables), afterwards all neurons starting from output layer are replaced with the sets of If–Then rules, finally all of these rules are merged and pruned to get final If–Then rules set. Instead of performing such rules merging, a solution was proposed to utilize only input neurons quantization tables for classification decision tree construction. In the proposed algorithm, to extract the decision tree from ANN, only quantization table borders are used (points at which class belongingness decision potentially changes) as a candidate split-points to estimate Information-Gain or GINI value on the full training dataset. This improvement lowers algorithm complexity and potentially amount of computations. To assure that the decision tree is accurately describing original ANN, the decision tree extraction dataset is generated, such that it contains points nearest to the quantized ANN classification boundary, see step 3 in Figure 3.2. A comparison of existing and proposed approaches can be seen in Figure 3.3.

1. Trained MLP
2. Activation values discretization / clusterization
3. Candidate-points for classification decision tree
4. Classification decision tree build up
5. Built classification decision tree

Fig. 3.2. Decision tree built from clusters boundaries formed using input neurons outputs.
The main steps of the proposed algorithm for the decision tree extraction are (see steps 2-4 in Fig. 3.2):

1) pruned ANN classifier non-linearity is broken via neurons outputs clusterization;
2) candidate split-points are acquired from input layer neurons quantization tables;
3) a modified decision tree algorithm uses found candidate cluster boundaries from input layer neurons as a candidate split-points (while using full training dataset to calculate GINI or Information-Gain) to build a classification decision tree.

![Diagram](https://via.placeholder.com/150)

**Fig. 3.3.** Existing and proposed approaches for rules extraction.

The described algorithms were implemented as an extension to the well-known deep learning package *nn* of *Torch7* library in *Lua* programming language [31]. Additionally, algorithms were reimplemented in Python programming language extending *PyTorch* deep learning library.

### 3.3. Developed Algorithm Validation

To validate the developed algorithm experiments have been performed on nine well-known UCI repository datasets. Table 3.1 holds the results of the experiments. For the C4.5 algorithm, results were taken from [2], [10], [17], [40], [74]. An example of the decision boundary for the pruned feedforward neural network is depicted in Figure 3.4(a), the decision boundary for the extracted binary decision tree can be seen in Figure 3.4(b). Average size decision tree, see Figure 3.5(a) and minimal and maximal, see Figure 3.5(b) extracted decision trees for Ripley and 3-class Iris datasets are presented in Figure 3.5.
Table 3.1

Accuracies and Leafs Counts for MLP, Pruned MLP, Extracted Tree, and C4.5

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MLP train/test</th>
<th>Proposed methods</th>
<th>C4.5/J48</th>
<th>C4.5 leaves</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Pruned MLP train/test</td>
<td>Extracted tree train/test</td>
<td>Extracted tree leaves $^{\text{min}}_{\text{max}}$</td>
</tr>
<tr>
<td>Iris</td>
<td>0.9911/0.9667</td>
<td>0.9652/0.9667</td>
<td>0.9689/0.9533</td>
<td>4.2$^3_5$</td>
</tr>
<tr>
<td>Pima diabetes</td>
<td>0.7319/0.7332</td>
<td>0.7253/0.7279</td>
<td>0.7433/0.7423</td>
<td>2$^2_2$</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>1.0000/0.9087</td>
<td>0.9552/0.9544</td>
<td>0.9546/0.9059</td>
<td>21.6$^{12}_{26}$</td>
</tr>
<tr>
<td>Ripley</td>
<td>0.8560/0.8920</td>
<td>0.8640/0.8982</td>
<td>0.8631/0.8946</td>
<td>12.6$^4_2$</td>
</tr>
<tr>
<td>Haberman</td>
<td>0.7509/0.7381</td>
<td>0.7542/0.7547</td>
<td>0.7567/0.7446</td>
<td>2$^2_2$</td>
</tr>
<tr>
<td>Monks-1</td>
<td>1.0000/1.0000</td>
<td>1.0000/1.0000</td>
<td>1.0000/1.0000</td>
<td>8$^6_0$</td>
</tr>
<tr>
<td>Monks-2</td>
<td>0.6923/0.6736</td>
<td>0.7160/0.6435</td>
<td>0.7041/0.5949</td>
<td>14$^{14}_{14}$</td>
</tr>
<tr>
<td>Monks-3</td>
<td>0.9754/0.9259</td>
<td>0.9508/0.9722</td>
<td>0.9508/1.0000</td>
<td>6$^5_0$</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>0.8051/0.8001</td>
<td>0.8006/0.8099</td>
<td>0.8092/0.8102</td>
<td>2$^2_2$</td>
</tr>
</tbody>
</table>

If there are a small number of cluster boundaries to be used as split points for decision tree building, then computational complexity is rather small. Experimental results show that extracted tree classification accuracy is directly related to a neural network accuracy, which is used as a knowledge source, as well as to the number of input neurons output values clusters boundaries that are used to build a tree. Authors of [74] have stated that they were able to get MLP accuracy of 0.9876, which is much higher than in presented experiments. This suggests that in the case of better trained MLP, the extracted tree would have higher classification accuracy. Of course, if comprehensibility and simplicity of the extracted decision tree are of higher importance, then more aggressive pruning can be applied and accuracy will be lower. But still, to get the best results it is important to have a trained ANN with the highest possible classification performance.

![Fig. 3.4. Ripley test dataset split into two classes by (a) pruned neural network and (b) extracted decision tree.](image-url)
Table 3.1 shows that the developed decision tree extraction algorithm is outperforming the C4.5 algorithm in the majority of cases in terms of classification accuracy. Cases when it shows lower performance are those were the original ANN has shown initial poor classification results. Rules-wise count both algorithms are on par, but having higher accuracy allows to further lower rules count, which gives the developed approach an edge over the C4.5 algorithm.

![Decision Tree Diagrams](image)

Fig. 3.5. Sample decision tree extracted for (a) Ripley data set and (b) decision trees of varying depth extracted for Iris data set.

Chapter 3 has reviewed knowledge extraction approaches in application to ANN; justified knowledge extraction from ANN in the form of a binary classification decision tree; justified the selection of Torch7 as a base for the deep learning framework for proposed NNKX [31] implementation; presented and experimentally validated the developed algorithm for classification decision tree extraction. The proposed algorithm has a lower classification error than C4.5 in all tested datasets, where a trained ANN had error lower than the C4.5 classifier. Experiments validated the developed approach. The extracted decision tree has high classification accuracy and low complexity, with that research task number three stated as accomplished.
4. OPTIMIZATION-BASED METHODS FOR RULE EXTRACTION

Chapter 3 introduced the decision tree extraction approach applicable to a fully connected trained feed-forward artificial neural network (ANN). But when it comes to training a classification model, it can be a case that the selected ANN will not show the best results. Therefore, Chapter 4 aims at the development of two alternative optimization-based approaches, one that allows the extraction of elliptical rules from radial basis function (RBF) neural networks and another approach allowing acquiring hyper-polytope classifier in input data space and approximating it using If–Then rules. Hyperpolytopes acquisition is a separate problem and two approaches to acquiring them are presented.

4.1. Elliptical Rules Extraction From RBF Neural Networks

Apart from fully connected ANNs with sigmoidal activation functions, RBF neural network (RBFNN) can be used as an alternative classifier. Elliptical rules are more expressive than If–Then rules. Therefore, optimization-based pedagogical approach allowing to extract elliptical rules from RBFNN was developed and evaluated [21], [27].

Optimization Problem

The extraction of elliptical rules from the trained RBFNN can be treated as a non-convex optimization problem of finding ellipsoids of maximum volume inscribed into the input space area defined by RFBNN classification decision boundary. Let us denote an ellipsoid as

$$\varepsilon = \{Bu + d \mid \|u\|_2 \leq 1\},$$

(4.1)

where $B$ is a symmetric positive definite matrix; $u$ is a unit ball (set of points of distance one from a fixed central point); $d$ is a vector representing ellipsoid center; $\|u\|_2$ denotes the Euclidean norm, i.e., $\|u\|_2 = (u^T u)^{1/2}$.

Ellipsoid $\varepsilon$ is a unit ball under affine transformation. In such a formulation, the ellipsoid volume is proportional to $\det B$ [32]. Thus, the optimization problem can be posed as

$$\max \log(\det(B)), \text{ s.t. } \text{RBFNN} \supseteq \varepsilon,$$

(4.2)

The described problem allows finding the first ellipsoid inscribed into the RBFNN decision boundary. A multi-start search will help in dealing with local optimums due to non-convexity of optimization problem. In most cases, it will be insufficient to represent RBFNN with a single ellipsoid; thus, iterative search for additional ellipsoids is required. To find other ellipsoids it is sufficient to look for the newly inscribed ellipsoid (potentially overlapping previously found ellipsoids) with maximum volume not covered by the already found ellipsoids:

$$\max(\varepsilon_{\text{vol}} - E_{\text{vol}}) - P, \text{ s.t. } \text{RBFNN} \supseteq \varepsilon,$$

(4.3)

where $\varepsilon_{\text{vol}}$ is the volume of newly found ellipsoid; $E_{\text{vol}}$ is the volume of already existing (previously found) ellipsoids; and $P$ is penalty term, see description below. Introduced penalty term $P$ calculates the minimal distance (Eq. 4.4) between the candidate ellipsoid...
center and the border of a set formed by the intersection of all previously found ellipsoids.

\[ P = \min(\text{dist}(\epsilon_{\text{center}}, E_{\text{surf}}): E_{\text{surf}} \in (E_1 \cup \ldots \cup E_N)), \] (4.4)

where \( \epsilon_{\text{center}} \) is the center of newly found candidate ellipsoid; and \( E_{\text{surf}} \) is the surface formed by the intersection of already found ellipsoids. Introduction of \( P \) term ensures that on each iteration optimization objective will find a new ellipsoid, which will cover the largest possible portion of the volume not yet covered by existing ellipsoids.

**Experiments and Results**

Experiments have been conducted on a synthetic two-dimensional Ripley dataset (to aid visual analysis), which can be found in the UCI dataset repository [49]. The algorithm described in [37] was used in RBFNN initialization to construct several neural networks containing a variable number of neurons. In experiments only closed RBFNN defined classification boundaries were observed, which may be seen in figures. Looking at the algorithm, one can notice \( \text{maxEllipsoidsCount} \) variable. It was initialized with a number of neurons in the subject RBFNN, the only exception was a network with 9 neurons for which the maximum number of ellipsoids to be extracted had been set to 7. A number of neurons in RBFNN was chosen to be 2, 6, 7, and 9. Overall visual analysis confirms that the algorithm works as expected, while experimental results (see Table 4.1) show excellent performance of the extracted elliptical rules.

<table>
<thead>
<tr>
<th># of neurons in RBFNN</th>
<th>RBFNN train accuracy</th>
<th>RBFNN test accuracy</th>
<th>Ellipsoid train accuracy(_{\text{std.dev.}})</th>
<th>Ellipsoid test accuracy(_{\text{std.dev.}})</th>
<th>Ellipsoids number mean(_{\text{min}})</th>
<th>Ellipsoids number mean(_{\text{max}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 neurons</td>
<td>0.852</td>
<td>0.911</td>
<td>0.8400(_{0.000})</td>
<td>0.8870(_{0.000})</td>
<td>2.2(_{\frac{1}{2}})</td>
<td>2.6(_{\frac{1}{2}})</td>
</tr>
<tr>
<td>6 neurons</td>
<td>0.868</td>
<td>0.905</td>
<td>0.8680(_{0.002})</td>
<td>0.9032(_{0.005})</td>
<td>4.4(_{\frac{1}{5}})</td>
<td>4.6(_{\frac{1}{5}})</td>
</tr>
<tr>
<td>7 neurons</td>
<td>0.876</td>
<td>0.905</td>
<td>0.8760(_{0.000})</td>
<td>0.9031(_{0.002})</td>
<td>5.1(_{\frac{1}{7}})</td>
<td>5.7(_{\frac{1}{7}})</td>
</tr>
<tr>
<td>9 neurons</td>
<td>0.868</td>
<td>0.905</td>
<td>0.8728(_{0.005})</td>
<td>0.9039(_{0.001})</td>
<td>6.8(_{\frac{1}{7}})</td>
<td>7.1(_{\frac{1}{7}})</td>
</tr>
</tbody>
</table>

Experiment results show that the extracted ellipsoids have almost identical accuracy rates to original RBFNN while having an equal or smaller count of ellipsoids (in comparison to RBF neurons used in NN). Thus, the proposed algorithm works as expected. An important point to mention is computational complexity, as the algorithm uses RBFNN to check whether an ellipsoid fully lies within the RBFNN decision boundary. This is a computationally intensive operation, but it is parallelizable.

Summarized list of current sub-chapter contributions is as follows: optimization problem formulation, including a specific penalty; equipped objective function; and programmatic realization end experimental validation of the proposed algorithm. It was shown that a small number of found ellipsoids could perform classification with a small drop of classification accuracy. This proves the proposed approach to be feasible, especially for RBFNN with small input vector dimensionality.
4.2. Rules Extraction Using a Piece-Wise Approximation Algorithm

The current sub-chapter presents an application of optimization techniques for If–Then rules extraction from the piece-wise linear classifier. This direction was chosen because as it was already shown in [127], non-linear sigmoidal neurons decision boundary can be approximated by piece-wise linear functions. The developed algorithm is a generalization of the previously developed algorithm [80], which was developed to extract If–Then rules from linear support vector machine (SVM) classifier.

Overview of the Approach

To test If–Then rules extraction algorithm, a piece-wise polytope classifier was developed instead of piece-wise approximations of sigmoidal neurons outputs. The core idea of the developed piece-wise classifier is to build convex polytopes defined by hyperplanes around clusters of data points. Afterwards If–Then rules are extracted from such convex polytopes. Alternatively MSM-T classifier can be used as a source of polytopes [28]–[30], [79]. The developed algorithm is recursive, its main steps are:

1) acquisition of hyperpolytope classifier (Fig. 4.1(a));
2) allocation of best (by volume or by covered points count) If–Then rule (Fig. 4.1(b));
3) recursion start: splitting space into non-covered sub-spaces; finding the best If–Then rule in each of the found sub-spaces (Fig. 4.1));
4) checking recursion depth limit, if not reached, repeat step 3, otherwise stop (Fig. 4.1(b)).

Fig. 4.1. If–Then Rule extraction algorithm iterations.

If–Then Rules Extraction Algorithm

The described algorithm follows the defined linear programming (LP) problem [28]–[30], [79] and defines two vertices, namely representing lower and upper bounds of the found
hypercube. Having a single hypercube inscribed into the polytope can be insufficient in terms of classification fidelity. To overcome this undesirable result, the recursive search should be applied to search for additional hypercubes inscribed into the remaining regions of the polytope. This process could be repeated recursively so that ongoing search for the smaller uncovered regions will generate more and more rules that will asymptotically approximate the original polytope classifier with the desired level of classification fidelity. Such remaining regions of interest could be defined as follows:

\[ I^l_i = \{ x \in \mathbb{R}^n, \text{ s.t.,} \begin{align*} l^*_j < x_j & \leq u^*_j \forall 1 \leq j \leq i, \\ x_i & \leq l^*_i, \end{align*} \]  
\[ I^u_i = \{ x \in \mathbb{R}^n, \text{ s.t.,} \begin{align*} l^*_j & \leq x_j < u^*_j \forall 1 \leq j \leq i, \\ x_i & \geq u^*_i, \end{align*} \]  

(4.5)

where \( I^l_i, I^u_i \) are polytope regions that are surrounding extracted rule for the \( i \)-th dimension; and \( l, u \) are upper and lower bounds of the currently processed hypercube (rule). Presented in Equation 4.5 rule inequalities are satisfied for the first \( i - 1 \) dimensions of \( x \), the inequality that relates to the \( i \)-th dimension is not satisfied, and the rest dimensions are free and should not be constrained. To support the recursive search, it is important to guarantee that new recursively inscribed hypercubes will not intersect with each other. Consider dimensions \( i, j \) with \( j > i \). For each \( x \in I_j \), we have \( l^*_i < x_i < u^*_i \), and for each \( x \in I_i \), we have \( x_i \leq l^*_i \) or \( x_i \geq u^*_i \). Hence, \( I_i \) are non-intersecting, and the rules that are acquired for each \( I_i \) differ in terms of approximated polytope region. It should be noted that polytopes could have intersections between each other, thus the extracted rules (hypercubes) could be intersecting. The optimization of extracted rules is not part of the current effort.

**Experiments and Results**

For the developed rule extraction approach verification and testing several public UCI datasets [49] were selected. The verification of the proposed method is covered in [28]–[30], [79]. Datasets for the experiments were selected based on popularity criteria.

Before the actual rule extraction algorithm can take place, polytopes should be created. To acquire polytopes from which If–Then rules will be extracted it is possible to use the MSM-T classifier, which is as a classification decision tree with optimal splits that are not parallel to axes. Another option is to use a proposed piece-wise linear classifier, see [79] for details of the proposed method. For datasets that were not originally separated into validation and training sets 10-fold cross-validation was performed, and averaged classification accuracy was collected. In the case of Monks dataset training and validation, datasets were already provided and 10 experiments were conducted.

It can be seen in Table 4.2 that all datasets, except “Balance-Scale”, are not so nicely separable using Linear SVM. On the other hand, an approximation of a nonlinear decision surface gives a necessary boost of the classification accuracy for the polytope classifier and for extracted rules. Here C4.5 fails to perform good classification. Multi-Surface Method Tree (MSM-T) method falls behind SVM methods and Polytopes. The rules extracted from MSM-T show high classification error. Empirically it was found that the increase in recursion depth for rules extraction from MSM-T helps to lower the classification error.
The rows highlighted in grey are the results shown by developed classifiers. Bold highlights the best overall classification result on test set across all datasets (columns). Green, yellow and red are highlighting explainable classifiers accuracies. Here one can see that the rules extracted from polytope classifier have higher accuracy than rules extracted from MSM-T. Overall SVM with RBF kernel outperforms other classifiers. Among all classifiers, only three classifiers (C4.5, Rules (MSM-T) and Rules (polytopes)) are explainable. Among them, the rules extracted from hyper-polytopes (lower row) have the highest accuracies. This proves that the chosen approach can be successfully applied to extract precise rules.

The current chapter presents two approaches developed in the scope of optimization-based knowledge extraction. These approaches complement and serve as an alternative methods of knowledge extraction in cases when the decision tree extraction from trained ANN has shown poor results. Also, in cases when ANN to be described by the set of the rule is RBFNN and we are willing to lower the number of rules – elliptical rules are a better option. In case when one has hyperpolytopes to be described with If–Then rules, the developed approach provides yet another alternative to classification decision tree extraction.

Experimental validation has shown that If–Then rules can be extracted from hyper-polytopes using the developed algorithm as a convex-optimization problem solution. It was shown that extracted If–Then rules have high accuracy, thus can be used to describe original classifier and serve as yet another tool in machine learning practitioner toolbox. Also, a non-convex optimization problem was posed, and a new algorithm developed that supports the extraction of Elliptical rules. The above said allows concluding that the research task related to development and assessment of optimization-based methods for If–Then and elliptical rule extraction from trained FFNN and RBFNN is accomplished.

Research task number four, related to the development and assessment of optimization-based methods for If–Then and elliptical rules extraction from trained FFNN and RBFNN is accomplished.
5. KNOWLEDGE EXTRACTION METHODOLOGY

Based on accomplished work and acquired results, a methodology for rules extraction is developed and experimentally validated in the current chapter. Unified workflow for knowledge representation and model selection, knowledge extraction, assessment, and refinement is presented. Review of knowledge representation and extraction from artificial neural networks and other types of classifiers covered in Chapter 1 serves as a basis for the methodology, which presents a pruning algorithm, see Chapter 2, a novel algorithm for binary classification decision tree extraction from fully connected multilayer ANN, covered in Chapter 3. Finally, optimization-based method for rules extraction using convex optimization problem is covered in Chapter 4. The same chapter presents a novel developed (non-convex optimization) method for elliptical rules extraction from RBFNN classifier. The proposed workflow [18]–[20] is experimentally validated and guidelines on how to acquire simple or precise rules are given.

5.1. Methodology Development

Methodology developed as part of the dissertation divides knowledge extraction (KE) process into four main stages.

1. Knowledge representation schema and classifier type selection.
2. Classifier training and preparation for KE steps.
3. Knowledge extraction.
4. Extracted knowledge assessment and refinement.

The first step assumes knowledge schema selection. Elliptical rules are more expressive than decision tree or If–Then rules, but less comprehensible. So, if the overall goal is to understand how classification is performed elliptical rules might be a suboptimal choice. On the other side decision tree can be extracted only from ANN, hence if the best performing classifier is piece-wise hyperpolytopes then If–Then rules will have to be used as knowledge representation.

The second and third steps are straightforward. In case the classifier is already pre-trained KE can be performed right away. In the case of ANN pruning can be applied, which can be used to influence on precision and number of extracted rules.

The fourth step is knowledge assessment and refinement. If extracted rules are too complex, several possibilities to mitigate that are listed below.

- Rules can be merged and pruned.
- A greater degree of pruning should be applied on ANN, or lower amount of ellipsoids or smaller depth of If–Then rules recursion should be set.
- The classifier could be retrained to have altered training parameters and hyperparameters and KE steps should be re-run.
- As an alternative, if dataset size allows it, smaller sub-space region, were used classifier performance is poor, can be used to train a separate classifier to run KE on it.

All developed algorithms are combined into a unified methodology, see Figure 5.1.
Fig. 5.1. Methodology for knowledge extraction. Contribution of the Thesis is highlighted in grey.
5.2. Precise vs. Comprehensive Rules

To validate the proposed workflow and assess guidelines for acquiring precise or simple rules, an experiments plan has been developed and executed on medium and small sized datasets. To overcome large computational requirements needed to process real-life medium-sized (~50,000 records) Adult Census dataset (taken from UCI repository), two modifications were incorporated into the KE algorithm:

- neurons output discretization via outputs rounding to $n$ digits for all neurons become a mandatory step, disregarding possible performance degradation;
- neurons outputs clusterization was performed on a data subset (in experiments, it was 15% of the training set).

The experiments goal is to prove that the parameter controlling allowed performance metric degradation during neurons outputs clusterization phase allows controlling the extracted decision trees complexity and classification performance. The second question was to understand how pruning, as a preceding step, influences the extracted decision tree complexity and performance.

Results gathered in Table 5.1 hold means over ten experiments for each dataset and show that aggressive pruning coupled with large performance degradation threshold for neurons outputs clustering results in lower classification performance and smaller decision trees. Light pruning and small threshold give larger, but more precise decision trees.

<table>
<thead>
<tr>
<th>ANN pruning level</th>
<th>Neurons output values clusterization</th>
<th>Characteristics</th>
<th>Datasets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggressive pruning</td>
<td>ANN</td>
<td>Discretized ANN Accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td>Low perf. degradation clustering threshold</td>
<td>Discretized ANN accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extracted tree accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extracted tree rules count / depth</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td>Medium perf. degradation clustering threshold</td>
<td>Discretized ANN accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extracted tree accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extracted tree rules count / depth</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td>Minimal pruning</td>
<td>ANN</td>
<td>Discretized ANN accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td>Low perf. degradation clustering threshold</td>
<td>Extracted tree accuracy, %</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ripley Test</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extracted tree rules count / depth</td>
<td>Adult Train</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Adult Test</td>
</tr>
</tbody>
</table>

Table 5.1: Knowledge Extraction Parameters Influence on Extracted Rules Complexity and Accuracy
Experiments on the Ripley dataset help to understand how the clusterization threshold parameter influences the extracted decision tree. As a starting point, Figure 5.2(a) displays how the initial classification boundary produced by trained and slightly pruned ANN looks like. The result of discretization and clusterization using small (Fig. 5.2(b)) and large (Fig. 5.2(c)) allowable performance degradation threshold is seen in Figure 5.2. Vertical and horizontal lines show clusters boundaries.

Fig. 5.2. Classification boundaries for the Ripley dataset.

A small number of points in Figure 5.2(c) is due to the utilization of quantization tables that are replacing neurons output values. Figure 5.3 shows the classification boundaries of decision trees extracted from (quantized) ANN using neurons outputs clusterization.

Fig. 5.3. Decision tree classification boundary of an ANN with (a) small and (b) large clustering parameter controlling performance degradation.

Experiments have shown that the primary tool for controlling the extracted decision tree complexity is the clusterization phase performance degradation threshold. Next auxiliary parameter belongs to discretization – how big rounding should be, to acquire a smaller count of neurons outputs and lower computational costs during the clusterization phase (at the cost of some classification performance degradation). Lastly, pruning itself can control the rules complexity and classification performance, but the pruning effect has smaller importance on extracted tree complexity.

In regards to accuracy, pruning plays an important role as a regularizer preventing
overfitting. In any case, when non-agressive pruning is applied, the extracted tree will have classification performance similar to ANN it is being extracted from.

The current chapter summarizes the conducted work and presents a unified workflow with recommendations for KE. This workflow and recomendations underline the research work accomplished in previous chapters and provides guidelines for knowledge extraction with experimental validation results. Guidelines are provided for the selection of classifiers and corresponding knowledge extraction algorithms, taking into account dataset characteristics. Workflow for dealing with overly complex or too simple extracted knowledge is presented and experimentally validated. The contribution of current work can be summarized as follows:

- proposed methodology – general workflow of knowledge extraction from trained ANN or hyper-polytope classifier;
- Recommendations formulated within the scope of methodology on classification model selection based on dataset characteristics;
- as part of the methodology, based on experimental validation proposed recommendations on pruning and knowledge extraction parameters selection, described parameters influence on extracted knowledge complexity and performance;
- defined assessment procedure suggesting how knowledge should be assessed and what corrective actions can be performed to fix problems (if any are found).

Classification decision tree extraction is the most general way of acquiring knowledge from a neural network. However, according to the “No Free Lunch” theorem, there is no single method, which will be equally good for all datasets. Thus, alternative algorithms for extraction of elliptical and If–Then rules were developed and proved to be usable. All listed improvements have become parts of the overall knowledge extraction workflow. The work presented in this chapter accomplishes task number five stated as a research task.
RESULTS AND CONCLUSIONS

The Doctoral Thesis is devoted to knowledge extraction from trained artificial neural networks (ANN). Within this work, an analysis of existing approaches to knowledge representation, ANN pruning, and knowledge extraction was performed. As a result of this analysis, a methodology for knowledge extraction was developed. This methodology lists typical knowledge representation schemes, and provides guidelines for selecting best knowledge representation. In the case when knowledge is extracted from ANN, due to lack of ready to use algorithms and low performance of available implementations, new pruning approach was developed. An algorithm for classification decision tree extraction from ANN was developed. To cover more use-cases, alternative optimization-based knowledge extraction approaches were developed. These approaches allow extraction of If–Then rules from the classifier described by hyperpolytopes and Elliptical rules from Radial-Basis function neural network (RBFNN). All developed approaches are united into single workflow along with recommendations in regards to choosing specific workflow steps. The decision tree classifier is supplied with experimentally proven recommendations allowing user to get either complex and precise or more comprehensible and less precise decision trees.

The aim of the Doctoral Thesis is to develop algorithms for ANN pruning and knowledge extraction from trained ANN and unify them into knowledge extraction methodology. Thesis accomplishments are as follows.

1. A review and analysis of existing scientific literature covering theory and algorithms for knowledge representation and extraction is performed. As a result, the research discovered pros and cons of existing approaches, which allowed to define requirements for knowledge extraction workflow.

2. Artificial neural networks pruning algorithms are reviewed, and new ANN pruning algorithm based on sensitivity measure with retraining and pocket memory is developed and evaluated. Recommendations for choosing weights vs. nodes pruning are developed. The algorithm ability to overcome local minimums is proven experimentally.

3. A new decompositional approach for classification decision tree extraction from trained multilayer perceptron is developed. The algorithm extends Torch7 based nn deep learning package with an additional neural network.

4. Optimization based pedagogical approaches for oblique If–Then and elliptical rules extraction from a set of convex hyper-polytopes and RBF neural network are developed and evaluated.

5. Generalized methodology for knowledge extraction is developed and evaluated. The methodology includes suggestions on the model selection (MLP, RBFNN, and convex polytopes), which influences knowledge representation and extraction approach. Methodology contains workflow, which guides knowledge extraction, assessment, and refinement depending on the selected classification model. The experimental validation of methodology testifies the conclusions.
The performance of all developed algorithms was evaluated and analysed to prove the proposed hypotheses. Based on the conducted research, several conclusions can be made.

- The developed pruning algorithm based on sensitivity analysis successfully escapes from local minimums and allows to control classification error rise. In the scope of sensitivity-based pruning in some cases, weights pruning can produce results better than neurons pruning. Although neurons pruning is more welcome as it is a less computationally intensive method and in general produces results on par with weights pruning, these findings prove the first hypothesis.

- Usage of retraining and memory-pocket trick are simple yet effective algorithmic improvements that, when used with sensitivity-based pruning algorithm, produce good pruning results – these findings contribute to the first hypothesis as well.

- In the scope of decision tree extraction from MLP, usage of input layer neurons output values classification decision boundaries (acquired via neuron output values clusterization) instead of replacement of all neurons with rules produces a good classification decision tree. Such method is simpler in terms of computational resources in comparison to the description of all neurons via sets of rules with subsequent rule clustering, merging and pruning (to get required rules for the input layer). Additionally, the developed approach for classification decision tree extraction allows to control the extracted tree complexity and classification accuracy – this proves the second hypothesis.

- Based on experiments results involving the developed approach, a conclusion can be made that optimization-based approach can be used for If–Then rules extraction from convex polytopes. On non-HPC hardware, this method is applicable to datasets with less than eleven attributes. Hence, this method is usable on subsets of input datasets as an alternative method in case the extraction of classification decision tree from MLP produces poor classification decision tree for a specific sub-region. The experiments prove that the extracted If–Then rules effectively approximate the input space regions bounded by hyper-polytopes – this proves the third hypothesis.

- Based on experiments involving the developed algorithm, a conclusion can be made that the optimization-based approach can be used for Elliptical rule extraction from RBFNN for datasets with less than four dimensions (on non-HPC hardware). This method is applicable as a way to replace large sub-trees in a decision tree with more expressive elliptical rules. The experiments prove that the extracted elliptical rules effectively approximate RBFNN and achieve similar classification accuracy – this proves the fourth hypothesis.

All posed theoretical questions are experimentally proven via proposed and developed approaches and methodology. As a result, in the scope of Thesis rules extraction methodology is developed and experimentally evaluated. The methodology allows performing a selection of knowledge representation schema and classification method, knowledge extraction, its assessment, and refinement. Further research directions can include research of ways to introduce reproducibility into ANN pruning and rules extraction, as well as neurons output values clusterization speedup.
BIBLIOGRAPHY


42


