JYVÄSKYLÄ STUDIES IN COMPUTING 25

Alexey Tsymbal

Dynamic Integration of Data Mining Methods in Knowledge Discovery Systems

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Agora-rakennuksessa (Ag Aud. 2)
joulukuun 19. päivänä 2002 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in the Building Agora, (Ag Aud. 2), on December 19, 2002 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2002

Dynamic Integration of Data Mining Methods in Knowledge Discovery Systems

JYVÄSKYLÄ STUDIES IN COMPUTING 25

Alexey Tsymbal

Dynamic Integration of Data Mining Methods in Knowledge Discovery Systems

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2002

Editors
Seppo Puuronen
Department of Computer Science and Information Systems, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN 9513913546
ISBN 951-39-1354-6 (PDF)

ISBN 951-39-1372-4 (nid.)
ISSN 1456-5390

ABSTRACT

Tsymbal, Alexey
Dynamic Integration of Data Mining Methods in Knowledge Discovery Systems
Jyväskylä: University of Jyväskylä, 2002, 69 p. (+included articles)
(Jyväskylä Studies in Computing,
ISSN 1456-5390; 25)
ISBN 951-39-1354-6
Finnish summary
Diss.

The purpose of this study is to develop the theoretical background and practical aspects of dynamic integration of data mining methods in knowledge discovery systems. Currently, the most popular approach to integrating a number of predictive data mining methods is majority voting or simply averaging. However, majority voting and other static approaches to integration have the important shortcoming that they do not take into account local expertise. An advanced technique for the dynamic integration of classifiers is developed in this thesis. The main assumption is that each learned model is best suited in some subareas of the whole application domain, where its local error is comparatively less than the corresponding errors of the other available models. Different strategies of dynamic integration are developed and analyzed: (1) Dynamic Selection (DS), (2) Dynamic Voting (DV), and (3) a combination of dynamic voting with dynamic selection – Dynamic Voting with Selection (DVS). It is proposed to apply the developed technique of dynamic classifier integration to local feature selection, by generating the base classifiers on different subsets of features. A number of different distance functions which can be used in dynamic classifier integration are analyzed. The application of dynamic classifier integration in a hierarchical classifier integration scheme called an arbiter tree is also considered. Additionally, it is proposed to use dynamic classifier integration instead of static voting in decision committee learning approaches like bagging and boosting. The application of dynamic classifier integration in ensemble feature selection is considered. Feature selection based ensembles generated with the contextual merit measure and with the random subspace method are also analyzed. Every algorithm developed in this study is evaluated experimentally on a number of datasets from the UCI machine learning repository, and some algorithms are applied to large real-world datasets.

Keywords: ensemble of classifiers, integration of data mining methods, classification, data mining, knowledge discovery in databases, machine learning, feature selection

ACM Computing Reviews Categories

H.2.8. Information Systems: Database Management: *Database Applications, Data Mining*
I.2.6. Computing Methodologies: Artificial Intelligence: *Learning*
I.5.1. Computing Methodologies: Pattern Recognition: *Models*
I.5.2. Computing Methodologies: Pattern Recognition: *Design Methodology*

Author's address Alexey Tsymbal
Dept. of Computer Science and Information Systems
University of Jyväskylä
P.O. Box 35, FIN-40351, Jyväskylä, Finland
alexey@it.jyu.fi

Supervisors Prof. Dr. Seppo Puuronen
Dept. of Computer Science and Information Systems
University of Jyväskylä
P.O. Box 35, FIN-40351, Jyväskylä, Finland

Prof. Dr. Vagan Terziyan
Dept. of Mathematical Information Technology
University of Jyväskylä
P.O. Box 35, FIN-40351, Jyväskylä, Finland

Reviewers Prof. Dr. Peter Kokol
Faculty of Electrical Engineering and Computer Science
University of Maribor
Smetanova ulica 17, 2000 Maribor, Slovenia

Dr. David W. Patterson
Faculty of Informatics
University of Ulster at Jordanstown
Shore Road, Newtownabbey, Co. Antrim
BT37 0QB, Northern Ireland, U.K.

Opponent Prof. Dr. Pádraig Cunningham
Trinity College Dublin
College Green, Dublin 2, Ireland

ACKNOWLEDGEMENTS

I would like to thank all those who helped me to carry out this research. This work was carried out in the Department of Computer Science and Information Systems, at the University of Jyväskylä. I am thankful to my scientific supervisor Prof. Dr. Seppo Puuronen, who never refused to share his time and scientific experience on the direction of the thesis in many fruitful discussions. I am thankful to my scientific supervisor Prof. Dr. Vagan Terziyan, who was also a supervisor of my first scientific project (M.Sc. project) "*Development of a Software Complex Intended for Creation of Automated Diagnostics Systems Using Methods of Multivariate Analysis: Intelligent Data Analysis Subsystem*". This M.Sc. project was the starting point of the present doctoral thesis. Prof. Dr. Vagan Terziyan has played a great role in establishing me as a researcher and encouraging me in my pursuit to continue my studies towards a doctoral degree. I would also like to thank Prof. Dr. Seppo Puuronen and Prof. Dr. Vagan Terziyan for co-operation and co-authoring the articles included in this thesis.

I would like to extend my thanks to the external reviewers of the dissertation, Prof. Dr. Peter Kokol, and Dr. David W. Patterson, and to all the unknown reviewers of the papers, for their constructive comments and suggestions. I am thankful to Prof. Dr. Peter Kokol, and to Prof. Dr. Herna Viktor for their valuable comments to my licentiate thesis.

This research has been initially funded by the Center for International Mobility (CIMO) during December 1997 – May 1998, and since February 1999 by the Graduate School in Computing and Mathematical Sciences (COMAS) of the University of Jyväskylä.

I am thankful to Professor Pekka Neittaanmäki for providing valuable comments and guidance that speeded up the process of the preparation of the present dissertation. I wish to express my appreciation to the staff of the department, and especially Johanna Savela, Kaarina Suonia, Ulla Kahakorpi, Mirja Tervo, Lea Hakala, Tapio Tammi, Tero Vartiainen, Jouko Kääriäinen, Jari Lepistö, and Jukka Järvinen. I thank Professor Seppo Puuronen for his help with the preparation of the Finnish summary.

I would also like to thank the UCI machine learning repository of databases, domain theories and data generators (UCI ML Repository) for the datasets, and the machine learning library in C++ (MLC++) for the source code used in this research.

Special thanks to my dearly beloved wife Iryna Skrypnyk. Last, but not least, I wish to thank my parents, my brother, and all my friends, who provided invaluable moral support.

CONTENTS

1 INTRODUCTION 11

2 RESEARCH BACKGROUND 13
2.1 Knowledge discovery and data mining 13
2.2 An integrated knowledge discovery management system 15
2.3 The task of classification and inductive learning: basic definitions 17
2.4 Learning algorithms 18
2.4.1 Decision tree learning 19
2.4.2 Instance-based learning 19
2.4.3 Bayesian classification 20
2.5 Evaluation of learning algorithms and learned models 21
2.5.1 Estimating the error rate 21
2.5.2 Bias / variance decomposition of error 23
2.5.3 Test of hypotheses 24

3 RESEARCH PROBLEM: INTEGRATION OF MULTIPLE DATA MINING METHODS 26
3.1 Ensembles and two basic aspects of their construction 26
3.1.1 Generation of a set of models to be combined 27
3.1.2 Integration of multiple learned models 29
3.2 Ensemble goodness criteria and their interrelation 32

4 OUR APPROACH TO THE PROBLEM 35

5 RESEARCH DESIGN 39
5.1 Research methods 39
5.2 Experimental design 40
5.3 Datasets used in experiments 41

6 SUMMARY OF THE INCLUDED ARTICLES 46
6.1 “A dynamic integration algorithm for an ensemble of classifiers” 46
6.2 “Local feature selection with dynamic integration of classifiers” 47
6.3 “Distance functions in dynamic integration of data mining techniques” 49
6.4 “Arbiter meta-learning with dynamic selection of classifiers and its experimental investigation” 50

6.4 “Decision committee learning with dynamic integration of classifiers” 52
6.6 “Ensemble feature selection with dynamic integration of classifiers” 53
6.7 “Ensemble feature selection with the simple Bayesian classification in medical diagnostics” 55
6.8 About the joint articles 56

7 CONCLUSIONS 58
7.1 Results and contributions of the thesis 58
7.2 Limitations and research directions 59

REFERENCES 61

FINNISH SUMMARY (YHTEENVETO) 68

ORIGINAL ARTICLES

LIST OF INCLUDED ARTICLES

I Puuronen, S., Terziyan, V. & Tsymbal, A. 1999. A dynamic integration algorithm for an ensemble of classifiers. In Z.W.Ras & A.Skowron (Eds.) *Foundations of intelligent systems: 11th International Symposium ISMIS'99, Warsaw, Poland, LNAI* 1609. Berlin: Springer, 592-600.

II Puuronen, S. & Tsymbal, A. 2001. Local feature selection with dynamic integration of classifiers. *Fundamenta Informaticae* 47(1-2), Special Issue "Intelligent Information Systems", 91-117.

III Puuronen, S., Tsymbal, A. & Terziyan, V. 2000. Distance functions in dynamic integration of data mining techniques. In B.V.Dasarathy (Ed.) *Data mining and knowledge discovery: theory, tools and technology II. Proceedings of SPIE,* Vol. 4057. Bellingham, WA: SPIE, 22-32.

IV Tsymbal, A., Puuronen, S. & Terziyan, V. 1999. Arbiter meta-learning with dynamic selection of classifiers and its experimental investigation. In J.Eder, I.Rozman & T.Welzer (Eds.) *Advances in databases and information systems: 3rd East-European Conference ADBIS'99, Maribor, Slovenia, LNCS 1691*. Berlin: Springer, 205-217.

V Tsymbal, A. 2000. Decision committee learning with dynamic integration of classifiers. In J.Štuller, J.Pokorný, B.Thalheim & Y.Masunaga (Eds.) *Current issues in databases and information systems. Proceedings of ADBIS-DASFAA 2000, Prague, Czech Republic, LNCS 1884*. Berlin: Springer, 265-278.

VI Tsymbal, A., Puuronen, S. & Skrypnyk, I. 2001. Ensemble feature selection with dynamic integration of classifiers. In *Int. ICSC Congress on Computational Intelligence Methods and Applications CIMA'2001, Bangor, Wales, U.K.* 558-564.

VII Tsymbal, A. & Puuronen, S. 2002. Ensemble feature selection with the simple Bayesian classification in medical diagnostics. *Int. Journal of Medical Informatics*. Elsevier Science (submitted as an extended version of the conference article, CBMS'2002).

1 INTRODUCTION

Knowledge discovery or data mining is the process of finding previously unknown and potentially interesting patterns and relations in large databases (Fayyad *et al.*, 1997). Current electronic data repositories are growing quickly and contain a huge amount of data from commercial, scientific, and other domain areas. The capabilities for collecting and storing all kinds of data exceed the abilities to analyze, summarize, and extract knowledge from this data. Numerous data mining methods have recently been developed to extract knowledge from these large databases. Selection of the most appropriate data-mining method or a group of the most appropriate methods is usually not straightforward. Often the method selection is done statically for all new instances of the domain area without analyzing each particular new instance. Usually better data mining results can be achieved if the method selection is done dynamically taking into account characteristics of each new instance.

During the past several years in a variety of application domains researchers have tried to learn how to create and use an ensemble of data mining methods. For example, Dietterich (1997) has presented the integration of multiple classifiers as one of the four most important directions in the machine learning research. The main discovery is that ensembles are often more accurate than the individual methods alone. According to Skalak (1997), the two advantages that can be reached through combining classifiers are: (1) the possibility that by combining a set of simple classifiers, we may be able to perform classification better than with any sophisticated classifier alone, and (2) the accuracy of a sophisticated classifier may be increased by combining its predictions with those made by an unsophisticated classifier.

The basic assumption in this thesis is that each data mining method is best suited inside certain subareas of the whole domain area. Thus if we are able to collect and derive the benefit of the information with respect to these "competence areas" of the mining methods, then better mining results can be achieved. The problem is to estimate these competence areas in a way that helps the dynamic integration of data mining methods. From this point of view data mining using an ensemble of methods has much in common with the problem of using multiple experts (Terziyan *et al.*, 1998). In both situations knowledge is

received from several sources and a mechanism is needed to decide about the use of this knowledge.

In this thesis, the emphasis is on the dynamic integration of classifiers. Classification is a typical data mining task where the value of some attribute for a new instance is predicted based on the given collection of instances for which all the attribute values are known (Aivazyan, 1989). In Chapter 2, the background to the research is considered. First, knowledge discovery and data mining are discussed. An integrated knowledge discovery management system is then presented. The basic definitions for the problem of classification and inductive learning, which are the focus of this thesis, are given; learning algorithms used throughout the thesis are reviewed, and basic approaches for evaluating learned models that are used in experiments are presented. In Chapter 3, the research problem of the thesis is considered, and related work is briefly reviewed. Two basic approaches to the integration of classifiers (combination and selection) suggested by researchers are discussed, and ensemble goodness criteria are considered. Chapter 4 presents our approach to the problem of the integration of multiple data mining methods. In Chapter 5, the research design of the thesis is considered. First, the research methods used are discussed. Then, the experimental design is considered, and information about datasets used in the experiments given. Chapter 6 contains summaries of the articles included in the thesis with each section providing a summary of the corresponding included article. In Chapter 7, main contributions of the thesis are summarized, and limitations of the research and future work are discussed.

2 RESEARCH BACKGROUND

This chapter gives a more detailed overview of the area of knowledge discovery and outlines the necessity for automated discovery algorithms. It overviews the different stages classically associated with the knowledge discovery process and goes on to propose a flexible, modular architecture for an integrated knowledge discovery management system. Next the different learning approaches used in the experimental work including, decision trees, instance based learning and Bayesian classification are explained. This is followed by an overview of the sampling techniques employed in the experiments, including, cross validation, random sampling and bootstrapping. The remainder of the chapter overviews how the results were interpreted statistically using the paired t-test, McNemar's test and the test for the difference of two proportions.

2.1 Knowledge discovery and data mining

Currently, electronic data repositories are growing quickly and contain huge amounts of data from commercial, scientific, and other domains. According to some estimates, the amount of data in the world is doubling every twenty months (John, 1997). The capabilities for collecting and storing all kinds of data exceed the development in abilities to analyse, summarise, and extract knowledge from this data. The commercial need for tools to handle the data glut has risen sharply in the last number of years: a large community of users need tools to mine databases for patterns that can be used to improve business processes, analyze scientific experiments, help doctors better understand the effects of treatments, and so forth (John, 1997).

Knowledge discovery in databases (KDD) is a combination of data warehousing, decision support, and data mining - an innovative approach to information management. KDD is the process of finding previously unknown and potentially interesting patterns and relations in large databases (Fayyad *et al.*, 1997).

Before the terms “knowledge discovery” and “data mining” became popular, researchers in areas such as statistics, machine learning, databases, neural networks, pattern recognition, econometrics, and many others were all working on the same kinds of problems. However, without a term under which to unite, the research suffered from fragmentation. KDD unites all of these disciplines together under the premise that there exists much valuable knowledge in databases, and that due to the tremendous and growing volumes of data involved, advanced computer algorithms running on fast hardware are a more economical way to discover this knowledge than tedious manual searches and queries by human analysts (John, 1997).

KDD is usually defined as “the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data” (Fayyad *et al.*, 1996, 84). KDD is a process comprising of many steps, which involves data selection, data pre-processing, data transformation, data mining (search for patterns), and interpretation and evaluation of patterns. The basic steps of the KDD process are presented in Figure 1 (these steps were defined, for example, in (Fayyad *et al.*, 1997).

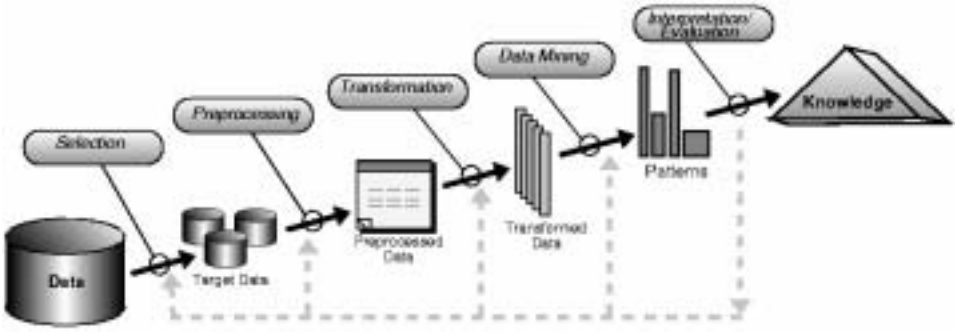

FIGURE 1 Basic steps of the KDD process (Fayyad *et al.*, 1996, 85)

The steps depicted start with the raw data and finish with the extracted knowledge, which was acquired as a result of the KDD process. The set of data mining tasks used to extract and verify patterns in data is the core of the process. Data mining (DM) consists of applying data analysis and discovery algorithms for producing a particular enumeration of patterns (or models) over the data (Brunk *et al.*, 1997). Most of current KDD research is dedicated to the DM step. However, this core area typically takes only a small part (estimated at 15%-25%) of the effort of the overall KDD process. The additional steps of the KDD process, such as data preparation, data selection, data cleaning, incorporating appropriate prior knowledge, and proper interpretation of the results of mining, are also essential to derive useful knowledge from data (Gaul & Säuberlich, 1999).

2.2 An integrated knowledge discovery management system

The current generation of database systems is based mainly on a small number of primitives of Structured Query Language (SQL) that is sufficient to support a vast number of business applications, but not sufficient to capture the emerging family of new applications dealing with knowledge discovery (Imielinski & Mannila, 1996). Most current KDD systems offer only isolated discovery techniques (tree inducers, neural nets, or rule discovery algorithms). Very few systems use a combination of available discovery techniques. Most current KDD systems are based on a loosely coupled architecture, where the database and the data mining subsystems are realised as separate independent blocks. This architecture demands continuous context switching between the data mining engine and the database. Such KDD systems are known as first-generation database mining systems (Imielinski & Mannila, 1996).

Numerous KDD systems have recently been developed. There are two basic branches of modern KDD systems. One branch includes domain-specific tools that support discovery in a single domain only. Such systems commonly use the user's language, and the user needs to know very little about the analysis process itself. The other branch refers to technique-oriented systems (Gaul & Säuberlich, 1999), which use one or several techniques for knowledge discovery and can be applied in different domains. Most of these systems use only a single discovery technique (e.g. decision tree, neural network, instance-based learning, or rule discovery). Very few systems propose a combination of the available techniques. Examples of systems which combine several discovery techniques are considered in (Anand *et al.*, 1997; Brunk *et al.*, 1997).

Nowadays there is a growing need for second-generation database mining systems to manage KDD applications in a similar way SQL-based systems successfully manage business applications. These systems should integrate the data mining and database subsystems and automate all steps of the whole KDD process (Figure 1) as much as possible. In this approach, the data mining engine is usually combined with the execution engine of the database, avoiding context switching between the database and data mining. These systems should be able to discover knowledge by combining several available KDD techniques. An essential part of the integrated KDD-process is the subpart that enables situation-dependent selection of appropriate KDD technique(s). In Figure 2, a general architecture of the integrated KDD-management system is proposed, which enables integration of multiple discovery techniques forming a platform upon which different KDD applications can be built.

Each subsystem presented in Figure 2 is intended to automate one basic step of the KDD process and when added together, the whole KDD process as much as possible. The database management subsystem performs data storage, retrieval, and manipulation using standard SQL operations. It is important that this subsystem should support distributed and heterogeneous data and complex data types, adopt a client / server architecture, and can access databases of standard widely-used formats, e.g. ODBC-compatible databases. For

example, the ODMG-93 standard can be supported as is proposed by Anand *et al.* (1997).

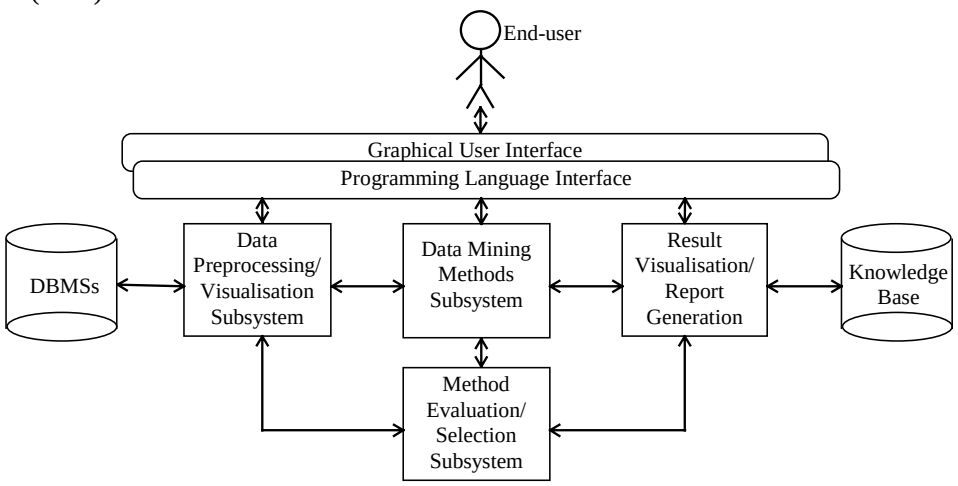

FIGURE 2 Integrated KDD management framework

The data pre-processing and visualisation subsystem provides tools to prepare the data for data mining and visualisation facilities by applying exploratory data analysis before the use of data mining techniques. The data pre-processing tools are used to remove noise, handle missing data fields, reduce data and make data projections.

The data mining subsystem incorporates different mining techniques for creating models and extracting patterns from the data, e.g. classification, clustering, and dependency modelling techniques. It is important that this subsystem be open (extensible), allowing the easy addition of new techniques through a "plug-in" interface as external tools or implemented as internal procedures using a built-in programming language. This allows rapid extension of the system without redeveloping the system core.

The method evaluation/selection subsystem is very important, helping the user to select an appropriate data mining method. In this thesis, the focus is on this subsystem. Different techniques for the integration of data mining methods were considered for example by Koppel & Engelson (1996); Merz (1996); Merz (1998); Merz (1999); Oza & Tumer (1999); Puuronen *et al.* (1999a); Puuronen *et al.* (1999b); Skalak (1997); and Wolpert (1992). An overview of techniques for integration of classifiers is given in (Dietterich, 1997). A technique for advanced dynamic integration of data mining methods is developed in this thesis. This technique is based on the assumption that each method is best suited only within certain subdomains of the whole application domain.

The result visualisation/report generation subsystem includes tools for the visualization of the models built and patterns discovered and for the generation of reports including discovery results. This subsystem helps the user to interpret and evaluate extracted patterns and models, which is essential for

the process of obtaining new knowledge. It also helps the user to determine the patterns that can be considered as new knowledge and to make correct conclusions.

The built-in programming language offered by the programming language interface is needed to provide the extensibility of the system and to make it configurable to different applications. SQL is extended in (Imielinski & Mannila, 1996) to the notion of KDD query language that is proposed for use in knowledge and data discovery management systems, i.e. in second-generation database mining systems. KDD query language can be used as a base for the built-in programming language.

The graphical user interface guides the user through the discovery process, helping to reduce the requirements in the user's expertise in KDD. This module hides the power and complexity of discovery subsystem deep inside the KDD management system. The graphical user interface should allow an end-user without any abilities in programming to use the system.

2.3 The task of classification and inductive learning: basic definitions

A typical data mining task is to explain and predict the value of some attribute of the data given a collection of fields of some tuples with known attribute values (Chan, 1996). This task is often solved with *inductive learning*. The goal of inductive learning is to build a general decision procedure based on a set of examples. The resulting model is then used to make predictions on previously unseen examples. Often an *example (*also called an *instance* or a *case)* is defined by specifying the value of each attribute. This is known as *attribute-value* (or feature-value) notation, and may be written as a row vector using the following notation:

$$\mathbf{x} = [v(x_1), v(x_2),\ldots,v(x_l)],$$

where $v(x_i)$ denotes the value of feature (attribute) x_i, and l is the number of features. Features which take on a value from an unordered set of possible values are called *nominal* or *categorical*. The feature Sex is categorical with the possible values {Male, Female}. Features, such as Age, Weight and Height, which can take on numeric values are said to be *continuous*. Continuous features are used whenever there is a linear ordering on the values, even if they are not truly continuous (*e.g.*, Age). The features used to define **x** may be paired with an extra nominal or continuous attribute y, called the *output attribute*, *class value* (if y is nominal) or *regression value* (if y is continuous).

The range of all possible values of the features of **x** is referred to as the *input space*, *instance space* or *example space*. The range of possible values of y is referred to as the *output space*.

Typically, examples with a given classification or regression value are used for building predictors, and are called *training* or *learning set*, or simply a *dataset*. The predictors are usually applied to examples without an output value,

sometimes called *test* or *unseen* examples. Predictors can perform one of two tasks. When the output is continuous, the task is *regression*, and when the output is nominal, the task is *classification*. The task of classification is the focus of this thesis.

The task of placing an example into one of a finite set of possible categories is called *classification*, i.e.,

$$C(\mathbf{x}) \in range(y),$$

where $range(y)$ denotes the set of possible values for the categorical output attribute y. Examples of the task of classification include the prediction of whether a person has heart disease (as well as many other medical diagnostics problems), technical diagnostics, and weather prediction, etc.

A common measure of a classifier's performance is the *error rate*, and it is calculated as the percentage of misclassified test examples (Merz, 1998). And conversely, *classification accuracy* (also called *generalization performance*) is the percentage of correctly classified test examples. More generally, the accuracy of a classifier is the probability of correctly classifying a randomly selected instance (Kohavi, 1995b).

The task of classification has been defined as the process of predicting the value of the class attribute for an unseen example using a prediction method or a classifier. Classifiers may vary widely from simple rules to neural networks. A classifier can be viewed as a general description or a *model* for a particular classification task. Typically, the construction of a model is guided by a collection of examples for which the correct output value is already known, i.e., the *training data* or *learning data*. The process of building a model from a training data is called *inductive learning* (Merz, 1998). In other words, inductive learning is the task of identifying regularities in some given set of examples with little or no knowledge about the domain from which the examples are drawn (Chan, 1996). The ability of a learned model (a classifier) to make predictions on "future" unseen data, or *generalization performance* is usually measured using a loss function and test data taken from the same distribution.

2.4 Learning algorithms

As it was noted, classifiers may be learnt (trained) using different learning algorithms from rule learning to neural networks. In this section, learning algorithms used throughout the dissertation are briefly described. They are the C4.5 decision tree learning, the *k*-nearest neighbor algorithm, and the Naïve-Bayes learning algorithm. These learning algorithms were used in experiments to build the base classifiers in the ensembles. All of these are well known in the data mining and machine learning communities and represent three completely different approaches to learning from data.

2.4.1 Decision tree learning

Decision tree learning is one of the most widely used inductive learning methods (Breiman *et al.*, 1984; Quinlan, 1986; Quinlan, 1993; Quinlan, 1996). A *decision tree* is represented as a set of nodes and arcs. Each node usually contains a feature (an attribute) and each arc leaving the node is labelled with a particular value (or range of values) for that feature. Together, a node and the arcs leaving it represent a decision about the path an example follows when being classified by the tree.

Given a set of training examples, a decision tree is usually induced in a "top-down" fashion by repeatedly dividing up the examples according to their values for a particular feature. This is known as a "divide and conquer" or "recursive partitioning" approach to learning. Initially all the examples are in one partition and each feature is evaluated for its ability to improve the "purity" of the classes in the partitions it produces. The splitting process continues recursively until all of the leaf nodes are of one class.

In general, decision tree learning algorithms are biased toward producing trees with the fewest number of decision nodes. This bias is known as Occam's razor, which, for the task of learning, states that the simplest decision structure which correctly classifies the data should be accepted. The requirement that all the data be correctly classified may result in an overly complex decision tree. Extra nodes may be added in response to minor variations in the data. The problem of being overly sensitive to minor fluctuations in the training data is known as *overfitting*, and it is a general problem for all learning algorithms. A common strategy for avoiding overfitting in decision trees is to "prune" away subtrees of the decision tree if it improves generalization performance on a too small set of pruning validation examples.

The decision tree learning algorithm used in this research is the C4.5 decision tree learning algorithm (Quinlan, 1993), which is the most widely used decision tree learning approach used today. C4.5 uses gain ratio, a variant of mutual information, as the feature selection measure. C4.5 prunes by using the upper bound of a confidence interval on the resubstitution error as the error estimate; since nodes with fewer instances have a wider confidence interval, they are removed if the difference in error between them and their parents is not significant (Quinlan, 1993).

2.4.2 Instance-based learning

An instance-based learning algorithm stores a series of training instances in its memory and uses a distance metric to compare new instances to those stored. Prediction on the new instance is based on the example(s) closest to it (Aha *et al.*, 1991; Wettscherech, 1994).

The simplest and most well studied instance-based learning algorithm is known as the "*nearest neighbor*" (NN) algorithm and is used to classify points in the instance space. The most elementary version of the algorithm is limited to continuous features with the Euclidean distance metric. To classify an unseen

example, its distance to all the training examples is calculated and the class label corresponding to the closest training example is assigned to the example. A more sophisticated version of the nearest neighbor classifier returns the most frequent class among the *k* closest training examples (denoted *k*-NN) (Aha *et al.*, 1991).

In experiments, we use the PEBLS instance-based learning algorithm (Cost & Salzberg, 1993). This algorithm is a kind of a nearest-neighbor algorithm, which uses a special distance metric called Value Difference Metric (VDM) for nominal features. In PEBLS, the distance between two values of a nominal feature is based on their class separability. A simplified version of the VDM (without weighting) used in our experiments defines the value of the distance function between two values *x* and *y* of a given nominal feature *a* as:

$$vdm_a(x,y) = \sqrt{\sum_{c=1}^{C} \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^2} = \sqrt{\sum_{c=1}^{C} \left| P_{a,x,c} - P_{a,y,c} \right|^2},$$

where $N_{a,x}$ is the number of instances in the training set that have value x for feature a; $N_{a,x,c}$ is the number of instances that have value x for feature a and class c; C is the number of classes in the problem domain; $P_{a,x,c}$ is the conditional probability that the class is c given that feature a has the value x, i.e., $P(c \mid x_a)$. Using the VDM, two values are considered to be closer if they have more similar classifications. Continuous features in PEBLS are discretized into a number of equal-sized discrete ranges, and then these values are treated as nominal.

2.4.3 Bayesian classification

The Naïve-Bayes classifier (John, 1997) uses Bayes rule (or Bayes' theorem) to optimally predict the class of a previously unseen example, given a training set. Bayes' theorem defines how to compute the probability of each class given the instance, assuming the features are conditionally independent given the class. The chosen class is the one that maximizes the probability:

$$P(c_i \mid \mathbf{x}_{test}) = \frac{P(c_i) P(\mathbf{x}_{test} \mid c_i)}{P(\mathbf{x}_{test})}$$

where c_i is the i-th class, $\mathbf{x}_{test}$ is a test example, and $P(A \mid B)$ is the conditional probability of A given B.

If the l input features are independent given the class, $P(\mathbf{x}_{test} \mid c_i)$ can be broken down into the product $P(x_1 \mid c_i) \ldots P(x_l \mid c_i)$, where x_j is the value of the j-th feature in the example $\mathbf{x}_{test}$. Then, the predicted class is the one which maximizes the probability:

$$P(c_i \mid \mathbf{x}_{test}) = \frac{P(c_i)}{P(\mathbf{x}_{test})} \prod_{j=1}^{l} P(x_j \mid c_i)$$

Estimating the probability $P(\mathbf{x}_{test})$ is unnecessary because it is the same for each class c_i. The remaining probabilities can be estimated from the training set. This is known as the naïve Bayesian classifier, referred to Bayes in this thesis.

More sophisticated Bayesian classifiers were developed, e.g. by John (1997), but only the naïve Bayesian classifier is used in the experiments in this dissertation.

The naïve Bayesian classifier relies on an assumption that is rarely valid in practical learning problems: that the features used to derive a prediction are independent of each other, given the predicted value. The earliest known reference to the naïve Bayesian classifier is the book "Perceptrons" by Minsky and Papert (1969).

Due to its perceived limitations, the simple Bayesian classifier has traditionally not been the focus of research. It has sometimes been used as the base against which more sophisticated algorithms are compared. However, it has been recently shown that, for classification problems where the predicted value is categorical, the independence assumption is less restrictive than might be expected (Domingos & Pazzani, 1996; Domingos & Pazzani, 1997; Friedman, 1997). Domingos and Pazzani (1997) have presented a derivation of necessary and sufficient conditions for the optimality of the simple Bayesian classifier showing that it can be optimal even when the independence assumption is violated by a wide margin. They showed that although the probability estimates that the simple Bayesian classifier produces can be inaccurate, the classifier often assigns maximum probability to the correct class.

A large-scale comparison of the simple Bayesian classifier with state-of-the-art learning algorithms on standard benchmark datasets was performed in (Domingos & Pazzani, 1997). The simple Bayesian classifier was found to be superior to each of the other learning algorithms even on datasets with substantial attribute dependencies. Besides, it has advantages in terms of simplicity, learning speed, classification speed, storage space, and incrementality. It also can easily handle a missing feature value of a learning instance allowing its other feature values still to contribute. Nevertheless, the simple Bayesian classifier is limited in expressiveness because it can only create linear frontiers (Duda & Hart, 1973). Therefore, even with many training examples and no noise, it does not approach the maximal accuracy on some problems.

2.5 Evaluation of learning algorithms and learned models

2.5.1 Estimating the error rate

For the purposes of algorithm comparison and selection, as well as for parameter setting, methods of estimating the performance of a set of learned models are needed. The goal of the model selection task is to estimate the generalization performance of a collection of learning algorithms and to select the algorithm with the lowest error estimate (Kohavi, 1995a). This section describes several evaluation methods that will be used throughout this dissertation.

The simplest way to estimate the accuracy is to use the *resubstitution estimate*, in which the model is tested on the same data it was built on (Kohavi, 1995b). However, the resubstitution estimate is a highly optimistic estimate of accuracy because learning algorithms attempt to minimize it. For some learning algorithms, e.g. linear discrimination, where it is usually hard to fit large amounts of data, the resubstitution estimate is reasonable. In (Kohavi, 1995b) it was also noted that for large enough samples, there is no need to look further than the resubstitution estimator when seeking a robust method.

Except for the resubstitution estimate, all methods of evaluating a learned model's performance consist of a sampling process wrapped around the induction process. Typically, a sample of the available examples (all having output values) is placed in a training set and the remaining examples are placed in a test set. This is sometimes called "holdout" approach, or simply test set estimation. The training set is used to derive the learned models (classifiers or regression models). The learned models are then used to make predictions on the test set and generalization performance is measured using the appropriate loss function.

Three major nonparametric statistical methods follow this methodology: (1) cross-validation, (2) random sampling or Monte Carlo cross-validation, and (3) bootstrapping (Merz, 1998). The method of sampling from the available examples is what distinguishes these evaluation methods. That is the focus of the rest of this section.

In *cross-validation* (CV) (Schaffer, 1993; Kohavi, 1995a), sometimes called rotation estimation, the examples are randomly split into v mutually exclusive partitions (the folds) of approximately equal size. A sample is formed by setting aside one of the v folds as the test set, and the remaining folds make up the training set. This creates v possible samples. As each learned model is formed using one of the v training sets, its generalization performance is estimated on the corresponding test partition. Typically CV is used to select a learning algorithm: the learning algorithm with the best average generalization performance is selected as the "winner". In this thesis, CV is also used to collect spatial cross-validation history of a learning algorithm's performance, which can later be used for dynamic selection of learning algorithms. Sometimes, *stratified CV* (Kohavi, 1995a) is used, where the folds are stratified so that they contain approximately the same proportions of classes as the original dataset. It was shown that in many cases stratification gives better estimation (Kohavi, 1995a). Sometimes, multiple runs of cross-validation can be useful for stabilization of the estimations (Kohavi, 1995b).

Random sampling or Monte Carlo cross-validation (Kohavi, 1995a) is a special case of v-fold cross-validation where a percentage of training examples (typically 2/3) are randomly placed in the training set, and the remaining examples are placed in the test set. After learning takes place on the training set, generalization performance is estimated on the test set. This whole process is repeated for many training/test splits (usually 30) and the algorithm with the best average generalization performance is selected. Random sampling is used

in most experiments throughout this dissertation to evaluate developed methods and to compare them with existing ones.

Bootstrapping (Kohavi, 1995a) is the process of sampling with replacement from the available examples to form the training and test partitions. If *M* examples are available, a sample of size *M* is drawn with replacement to form the training set. In the purest form of bootstrapping, the test set is the entire set of available examples. However, several "holdout" methods have been developed recently that form test sets from those examples not appearing in the training set (Kohavi, 1995a; Merz, 1998). In (Kohavi, 1995a) it was shown that on average, cross-validation methods are better than bootstrapping, and could be recommended for accuracy estimation and model selection.

2.5.2 Bias/variance decomposition of error

A number of recent investigations of learning algorithms have successfully analysed error performance in terms of the *bias plus variance decomposition* (Webb, 2000). According to Geman *et al.* (1992), in this decomposition we can view the expected error of a learning algorithm on a particular target function and training set size as having three components: (1) a *bias* term measuring how close the average model produced by the learning algorithm will be to the target function; (2) a *variance* term measuring how much each of the learning algorithm's guesses will vary with respect to each other (e.g., how often they disagree in the case of classification); and (3) a term measuring the minimum classification error associated with the Bayes optimal classifier for the target function (this term is sometimes referred to as the intrinsic target noise).

In the case of classification models, the *bias* can be considered as a measure of the contribution to error of the central tendency or most frequent classification of the learner when trained on different training data. The *variance* can be considered as a measure of the contribution to error of deviations from the central tendency.

The bias/variance analysis is useful in focusing our attention on two significant factors that govern the accuracy of classifiers learned by a learning system. If a learning system, when provided with different training data, develops classifiers that differ in their predictions, then the extent of such variations provides a lower limit on the average error of those classifiers when applied to any subsequent set of test data.

However, preventing such variance between the classifiers will not guarantee the elimination of prediction error. This error is also governed by both the degree to which the correct classifier for an object can differ from that for other objects with identical descriptions (irreducible error) and the accuracy of the learning bias.

A number of different formulations of bias and variance have been proposed in the field of classification learning (Geman, 1992; Breiman, 1996; Webb, 200). Two bias/variance related metrics are used in this study. *Contribution of bias to error* is that portion of the total error, across the distribution of test sets, that is due to errors committed by the central tendency

of the learning algorithm. This is the proportion of classifications that are both incorrect and equal to the central tendency. *Contribution of variance to error* is that portion of the total error across the distribution of test sets that is due to errors that are deviations from the central tendency of the learning algorithm. This is the proportion of classifications that are both incorrect and not equal to the central tendency. These two terms must sum to total error.

These metrics are used to evaluate the extent to which variations in the classifiers formed from one training set to another affect the error of the learning algorithm. High contribution of bias to error indicates high error resulting from the learning bias whereas high contribution of variance to error indicates high error resulting from the algorithm's responsiveness to variations between training sets (Webb, 2000). These two definitions have the desirable properties that: (1) bias is a direct measure of the contribution of the central tendency to total error, (2) variance is a direct measure of the contribution to error of deviations from the central tendency, and (3) the two terms (bias and variance) sum to total error (Webb, 2000).

2.5.3 Tests of hypotheses

The cross-validation methods considered in Section 2.5.1 are commonly used for comparing the generalization performance of two learning algorithms. Sometimes it is necessary to determine how significant the observed differences are. Most studies to date have relied upon the resampled Student's *t*-test (also known as the resampled paired *t*-test) for measuring the significance of the observed difference in generalization performance (Dietterich, 1998).

For the resampled paired *t*-test, a series of (usually) 30 trials is conducted. In each trial, the available sample is randomly divided into a training set of a specified size (e.g., typically two thirds of the data) and a test set. Learning algorithms *A* and *B* are both trained on the training set and the resulting classifiers are tested on the test set. Let $p_A^{(i)}$ (respectively $p_B^{(i)}$) be the observed proportion of test examples misclassified by algorithm *A* (respectively *B*) during trial *i*. If we assume that the 30 differences $p^{(i)} = p_A^{(i)} - p_B^{(i)}$ were drawn independently from a normal distribution, then we can apply Student's *t* test, by computing the statistic

$$t = \frac{\bar{p} \cdot \sqrt{n}}{\sqrt{\frac{\sum_{i=1}^{n} (p^{(i)} - \bar{p})^2}{n-1}}},$$

where $\bar{p} = \frac{1}{n} \cdot \sum_{i=1}^{n} p^{(i)}$, and *n* is the number of trials. Under the null hypothesis, this statistic has a *t* distribution with *n*-1 degrees of freedom. For 30 trials, the null hypothesis can be rejected if $|t| > t_{29,0.975} = 2.04523$.

As discussed in (Dietterich, 1998), there are many potential drawbacks in this approach. First, the individual differences $p^{(i)}$ will not have a normal

distribution, because $p_A^{(i)}$ and $p_B^{(i)}$ are not independent. Second, the $p^{(i)}$'s are not independent, because the test sets in the trials overlap (and the training sets in the trials overlap as well).

Recent studies (Dietterich, 1998; Salzberg, 1999) have shown that the resampled *t*-test and other commonly used significance tests have an unacceptably high probability of detecting a difference in generalization performance when no difference exists (Type 1 error). This is primarily due to the nature of the sampling process in the experimental design and the number of examples available. Some conclusions made in this dissertation are checked also with the McNemar's test, and the test for the difference of two proportions, which are claimed to have acceptable Type 1 error (Dietterich, 1998). In this thesis, the same results were obtained with all the tests, sometimes with different levels of significance.

Let us conclude this chapter by words from (Mitchell, 1997): "No single procedure for comparing learning methods based on limited data satisfies all the constraints we would like. It is wise to keep in mind that statistical models rarely fit perfectly the practical constraints in testing learning algorithms when available data is limited. Nevertheless, they do provide approximate confidence intervals that can be of great help in interpreting experimental comparisons of learning methods".

3 RESEARCH PROBLEM: INTEGRATION OF MULTIPLE DATA MINING METHODS

In this chapter, the research problem of the thesis is considered, and related work is briefly reviewed. Additionally criteria for assessing the competency / accuracy of ensembles are reviewed.

3.1 Ensembles and two basic aspects of their construction

Integration of multiple models (or an ensemble of models) to improve the generalization performance is currently an active research area. Dietterich (1997) showed that this area is currently one of the four most important directions in machine learning research and presented a thorough overview of different approaches. Integration of an ensemble of models has been shown to yield higher generalization performance than the best base model alone in different real-world tasks. The benefits of creating an ensemble of models were successfully presented for both the task of classification and the task of regression (Merz, 1998).

In general the ensemble prediction process (Figure 3) consists of a set of base models h_1, …, h_S, where S is the size of the ensemble formed during *the learning phase*. Each base model in the ensemble (models h_1 … h_S in this case) is trained using training instances of the corresponding training set $\mathbf{T}_i$, $i = 1, \ldots, S$. For the ensemble prediction, the predictions of the base models are combined during *the application phase* in some way $h^* = F(h_1, h_2, \ldots, h_S)$ to produce the final prediction of the ensemble.

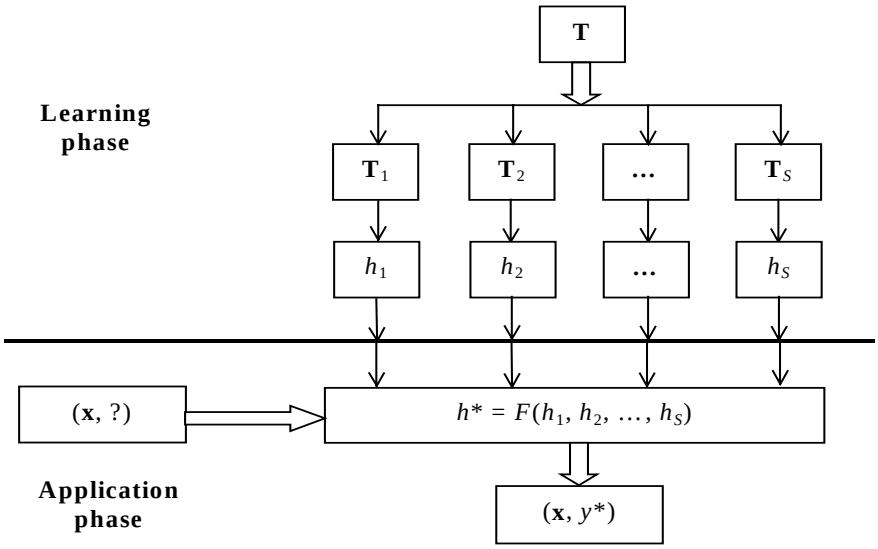

FIGURE 3 The general ensemble prediction process

The task of using multiple models generated with different data mining methods can be broken down into two basic questions: (1) what set of learned models should be generated?; and (2) how should the predictions of the learned models be integrated? (Merz, 1998).

3.1.1 Generation of a set of models to be combined

It was shown by several independent researchers, both in experiments and analytically, that the more diverse the predictions of the combined learned models are, the better will be the integration results achieved (Ali & Pazzani, 1996; Asker & Maclin, 1997; Merz, 1998; Skalak, 1997; Tumer & Chosh, 1996a; Tumer & Ghosh, 1996b). To generate a set of diverse learned models, several approaches have been tried.

One way of generating a diverse set of models is to use learning algorithms with heterogeneous representations and search biases (Merz, 1998), such as decision trees, neural networks, instance-based learning, etc. For example, in (Puuronen *et al.*, 1999a) we use C4.5 decision tree learning, PEBLS instance learning algorithm, and Bayesian learning to generate base classifiers in the ensembles.

Another approach is to use models with homogeneous representations that differ in their method of search or in the data on which they are trained. This approach includes several types of techniques for generating base models, such as learning base models from different subsets of the training data. For

example, two well-known ensemble methods of this type are bagging and boosting (Quinlan, 1996). Bagging builds each base model from data that are drawn from the training data with replacement using bootstrap sampling. On each draw, each training instance has an equal probability of being drawn. The construction of base models is independent of one another. In contrast, boosting builds its base models sequentially. On each trial a boosting technique draws examples following a probability distribution that insures that instances misclassified by a model constructed on a previous trial are more likely to be drawn for purposes of constructing the model on the current trial or, in the case with weighted instances, those examples just receive more weight. In (Tsymbal & Puuronen, 2000; Tsymbal, 2000) we considered a combination of the bagging and boosting approaches for generating base classifiers with our approach for the dynamic integration of classifiers.

The base models with homogeneous representations may be binary classifiers that are integrated to implement a multiclass learner (i.e., where the number of class labels is greater than 2). Each classifier in such an ensemble is learnt to distinguish one class label from the others. For example, Dietterich and Bakiri (1995) map each class label onto a bit string prior to learning. Bit strings for class labels are designed to be well separated, thus serving as error-correcting output codes (ECOC). An off-the-shelf system for learning binary classifications (e.g., 0 or 1) can be used to build multiple classifiers, one for each bit in the output code. An instance is classified by predicting each bit of its output code (i.e., label), and then classifying the instance as the label with the "closest" matching output code.

Another technique for building models with homogeneous representations is the use of different subsets of features (attributes) for each model. For example, in (Oza & Tumer, 1999) base classifiers are built on different feature subsets, where each feature subset includes features relevant for distinguishing one class label from the others (the number of base classifiers is equal to the number of classes). In (Tsymbal & Puuronen, 2000) we built base classifiers on different feature subsets to provide local feature selection using dynamic classifier integration. Finding a set of feature subsets for constructing an ensemble of accurate and diverse base models is also known as ensemble feature selection (Opitz, 1999).

Also, natural randomisation in the process of model search (e.g., random weight setting in the backpropagation algorithm for training neural networks) can be used to build different models with homogeneous representation. The randomisation can also be injected artificially. For example, in (Heath *et al.*, 1996) a randomised decision tree induction algorithm, which generates different decision trees every time it is run, was used for that purpose.

Sometimes, a combination of the techniques considered above can be useful in order to produce the desired characteristics of the generated models. For example, a combination of boosting and wagging (which is a kind of bagging technique) is considered by Webb (2000).

In addition to these general-purpose methods for generating a diverse ensemble of models, there are several techniques that can be applied to the

backpropagation algorithm for training neural networks. For example, Opitz and Shavlik (1996) employ a kind of genetic algorithm to search for a good population of neural network classifiers.

3.1.2 Integration of multiple learned models

This thesis is focused on the second issue, identified in Section 3.1 namely, "how should the predictions of the learned models be integrated?". Brodley and Lane (1996) have shown that only increasing coverage of an ensemble through diversity is not enough to insure increased prediction accuracy. If the integration method does not utilize the coverage, then no benefit arises from integrating multiple classifiers. Thus, the diversity and coverage of an ensemble are not in themselves sufficient conditions for ensemble accuracy. It is also important for ensemble accuracy to have a good integration method that will utilize the diversity of the base models.

A new technique for the integration of multiple data mining methods was developed and analyzed in this work. This technique was applied with the different ensemble generating schemas considered above. This work concentrated on the task of integrating multiple classifiers. The technique proposed in the thesis can be easily extrapolated to other data mining tasks such as regression. Furthermore in this section the problem of the integration of multiple classifiers is considered, and two basic approaches to the integration (combination and selection) suggested by researchers are discussed.

The integration of multiple classifiers, to improve classification results, is currently an active research area in the machine learning (Dietterich, 1997; Mitchell, 1997) and neural networks (Opitz & Maclin, 1999; Baxt, 1992; Hansen & Salamon, 1990; Krogh & Vedelsby, 1995; Opitz & Shavlik, 1996) communities. Deitterich (1997) showed that this area is currently one of the four most important directions in machine learning research and presented a thorough overview of different approaches. Different techniques to integrate an ensemble of classifiers have been considered for example in (Bauer & Kohavi, 1999; Breiman, 1996; Chan & Stolfo, 1997; Domingos, 1998; Kohavi, 1995a; Koppel & Engelson, 1996; Merz, 1998; Oza & Tumer, 1999; Puuronen *et al.*, 1999c; Schapire, 1999; Skalak, 1997; Tsymbal & Puuronen, 2000b; Tsymbal, 2000; Tsymbal *et al.*, 1999; Webb, 2000). The integration of an ensemble of classifiers has been shown to yield higher accuracy than the most accurate base classifier alone in different real-world tasks.

The challenging problem of integration is to decide which one(s) of the classifiers to rely on or how to combine the results produced by the base classifiers. The problem of integration can be defined as follows. Let us suppose that there is a training set **T** and a set of classifiers **C**. Let the training set **T** be $\{(\mathbf{x}_i, y_i), i = 1,...,n\}$, where *n* is the number of the training instances, $\mathbf{x}_i = \{x_j\}, j = 1,...,l$ is the vector of the attribute values of the *i*-th training instance (the values of the attributes are allowed to be continuous or nominal), and $y_i \in \{c_1,...,c_k\}$ is the classification of the *i*-th instance, where *k* is the total number of the classes. Let

the set of classifiers **C** be $\{C_1,...,C_S\}$, where *S* is the number of the *base classifiers*. Each base classifier is either derived using some learning algorithm or a hand-crafted classifier constructed using some heuristic knowledge. A new instance $\mathbf{x}^*$ is an assignment of values to the vector of the attributes $\{x_j\}$. Let each base classifier be able to classify the new instance. Then the problem of integrating the classifiers is to derive a technique to classify the new instance $\mathbf{x}^*$ using the classifiers of the set **C**.

Most existing classifier integration methods can be considered as instantiations of the *stacked generalization* (or *stacking*) framework (Wolpert, 1992) presented in Figure 4. The goal of stacking is to combine the predictions of a set of learned models based on information learned about their particular biases with respect to the training data. The basic idea is to perform another layer of induction over the outputs (or approximated outputs) of the learned models with the goal of learning how to correct for inappropriate biases (Wolpert, 1992).

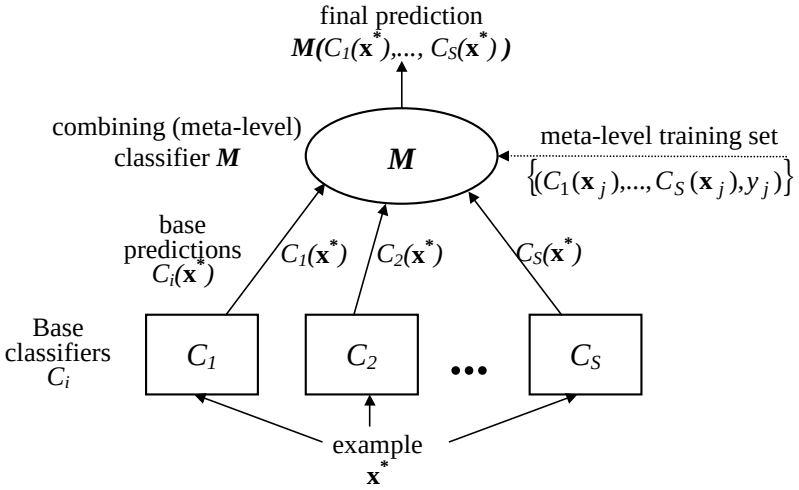

FIGURE 4 The stacked generalization framework

In the basic form of the stacked generalization layered architecture, the base classifiers $\mathbf{C} = \{C_1,...,C_S\}$ form the first layer, and a single combining classifier *M* forms the second layer. The combining classifier *M* is trained using the predictions of the base classifiers $\{C_i(\mathbf{x}_j)\}$ and the training set **T** (Wolpert, 1992). When a new instance $\mathbf{x}^*$ is classified, then first, each of the base classifiers C_i produces its classification result $C_i(x^*)$, and then the combining classifier *M* produces the final classification using these classification results. The technique proposed in this thesis can also be considered as an instantiation of the stacking framework. However, meta-level data, containing spatial information about errors of the base classifiers is used, instead of classifier predictions.

Techniques using two basic approaches have been suggested as a solution to the integration problem: (1) *combination approach,* where the base classifiers

produce their classifications and the final classification is composed using them; and (2) *selection approach*, where one of the classifiers is selected and the final classification is the result produced by it. Several effective techniques for the *combination* of classifiers have been proposed. One of the most popular and simplest techniques used to combine the results of the base classifiers, is simple voting (also called majority voting and select all majority (SAM)) (Bauer & Kohavi, 1999). In the voting technique, the classification of each base classifier is considered as an equally weighted vote for that particular classification. The classification that receives the biggest number of votes is selected as the final classification (ties are solved arbitrarily). Often, weighted voting is used: each vote receives a weight, which is usually proportional to the estimated generalization performance of the corresponding classifier. Weighted voting works usually much better than simple majority voting (Bauer, 1999).

More sophisticated combination techniques, which can be considered as instantiations of the stacking framework, include the SCANN method based on the correspondence analysis and using the nearest neighbor search in the correspondence analysis results (Merz, 1998; Merz, 1999); and techniques to combine minimal nearest neighbor classifiers within the stacked generalization framework (Skalak, 1997). Two effective classifier combination techniques based on the stacked generalization called “arbiter” and “combiner” were presented in Chan (1996). Hierarchical classifier combination has also been considered. Experimental results of Chan (1996); Chan & Stolfo (1997) showed that the hierarchical (multi-level) combination approach, where the dataset was distributed among a number of sites, was often able to sustain the same level of accuracy as a global classifier trained on the entire dataset.

There are still many open questions in the area of combining classifiers; even within such a widely used architecture as stacked generalization. For example, no clear guidelines exist as to how to decide which base classifiers should be used, which features should be used to form the meta-level training set for the combining classifier, and which combining classifier should be used. Different combining classifier algorithms have been considered by various researchers, including the boosting algorithm using weighted voting (Schapire, 1999), ID3 for combining nearest neighbor classifiers (Skalak, 1997), and nearest neighbor classification (Merz, 1999) to search in the space of correspondence analysis results (not directly on the predictions).

A number of *selection* techniques have also been proposed to solve the integration problem. One of the most popular and simplest selection techniques is the cross-validation majority (CVM) (Schaffer, 1993; Kohavi, 1995a). In the CVM, the cross-validation accuracy for each base classifier is estimated using the training set, and then the classifier with the highest accuracy is selected (ties are solved using voting). More sophisticated selection approaches include estimation of the local accuracy of the base classifiers by considering errors made in instances with similar predictions (Merz, 1996), learning a number of meta-level classifiers (“referees”) that predict whether the corresponding base classifiers are correct or not for new instances (each “referee” is a C4.5 tree that recognizes two classes) (Koppel & Engelson, 1996). Todorovski & Dzeroski

(2000) trained a meta-level decision tree, which dynamically selected a base model to be applied to the considered instance, using the level of confidence of the base models in correctly classifying the instance. Applications of classifier selection in medical diagnostics have been considered in (Skrypnyk *et al.*, 1999; Terziyan *et al.*, 1996; Tsymbal *et al.*, 1998). The local accuracy of the base classifiers is predicted there by analyzing the accuracy in near-by instances of the learning set, and in this thesis an advanced version of this classifier integration approach is considered.

The approaches of classifier selection techniques can be divided into two subsets: *static* and *dynamic* selection. The static approaches propose one "best" method for the whole data space, while the dynamic approaches take into account each new instance to be classified. The CVM is an example of the static approach, while the other selection techniques above and the one proposed in this thesis are examples of the dynamic approach.

Techniques for combining classifiers can be static or dynamic as well. For example, widely used weighted voting (Bauer *et al.*, 1999) is a static approach. The weights for each base classifier's vote do not depend on the instance to be classified. In contrast, the reliability-based weighted voting (RBWV) introduced in (Cordella *et al.*, 1999) is a dynamic voting approach. It uses classifier-dependent estimation of the reliability of predictions for each particular instance. Dynamic Voting (DV) and Dynamic Voting with Selection (DVS) techniques, which are focused upon in this thesis, are also examples of dynamic combination approaches. The weights for each base classifier's vote do depend on the instance to be classified (the weights are proportional to estimated classifiers' local accuracies). Usually, better data mining results can be achieved if the classifier integration is done dynamically taking into account the characteristics of each new instance.

3.2 Ensemble goodness criteria and their interrelation

An ensemble is generally more accurate than any of the base classifiers in the ensemble. Both theoretical and empirical research have shown that an effective ensemble should consist of base classifiers that not only have high classification accuracy, but that also make their errors in different parts of the input space (Bauer & Kohavi, 1999; Krogh & Vedelsby, 1995; Opitz & Maclin, 1999; Tumer & Gosh, 1996). Brodley and Lane (1996) show that the main objective when generating the base classifiers is to maximize the *coverage* of the data, which is the percentage of the instances that at least one base classifier can classify correctly. Achieving coverage greater than the accuracy of the best base classifier requires *diversity* among the base classifiers. Several researchers have presented theoretical evidences supporting this claim (Hansen & Solomon, 1990; Krogh & Vedelsby, 1995; Tumer & Gosh, 1996).

For example, Hansen and Salamon (1990) proved that, if the average classification error rate for an instance is less than 50% and the base classifiers in the ensemble are independent in the production of their errors, then the

expected error for that instance can be reduced to zero as the number of base classifiers included into the ensemble goes to infinity, when majority voting is used for integration. Such assumptions rarely hold in practice (for example, an outlier may easily have predicted classification error rate that is higher than 50%), and then the classification error rate over all the instances cannot necessarily be reduced to zero. But if we assume a significant percentage of the instances are predicted with less than 50% average error, gains in generalization will be achieved. A key assumption in this analysis is that the base classifiers should be independent in their production of errors.

Krogh and Vedelsby (1995) have shown that the classification error for an ensemble of neural network base classifiers is related to the generalization error of the base networks and to how much disagreement there is between them. They have proved that $E = \overline{E} - \overline{A}$, where E is the *ensemble generalization error*, $\overline{E}$ is the average of the generalization errors of the base networks weighted by corresponding beliefs, and $\overline{A}$ is the *ensemble ambiguity* measured as the weighted average of the squared differences in the predictions of the base networks and the ensemble. The ambiguity $\overline{A}$ that they used as a way to measure the disagreement, is common when disagreement among numeric predictors such as neural networks is measured.

A similar dependency derived by Tumer and Ghosh (1996) for classifiers that predict the a posteriori probabilities of the output classes is:

$$E = \frac{1 + (S-1) \cdot \sum_{i=1}^{L} P_i \delta_i}{S} \cdot \overline{E}, \qquad (1)$$

where E is the ensemble generalization error (beyond the Bayesian one), S is the size of the ensemble, P_i is the prior probability of class i, δ_i is the *average correlation factor* among the classifiers in the prediction of the a posteriori probabilities of class i, and $\overline{E}$ is the average error of the base classifier. Here the disagreement is measured as the correlation in the predictions of the a posteriori probabilities.

Ho (1998) presents another measure of *agreement* for two base classifiers producing categorical classifications in an ensemble. For two classifiers h_i and h_j, the classifier agreement $s_{i,j}$ is the amount that their decision regions overlap:

$$s_{i,j} = \int_R p(x)dx, \qquad (2)$$

where $R = \{x | h_i(x) = h_j(x)\}$, and $p(x)$ is the probability density function of x. Ho (1998) shows that for some decision trees the agreement $s_{i,j}$ can be calculated exactly in theory. When the exact calculation is infeasible, one may get an approximation by Monte-Carlo estimation, generating random points, or using testing samples that are representative for the given problems.

The most commonly used measure of a base classifier's diversity is the average difference Div_i^* between the base classifier and the ensemble (Brodley & Lane, 1996). Let $Dif(a,b)$ be the difference in two classifications a and b, which

is zero if the classifications are the same and one if they are different, then Div_i^* is:

$$Div_i^* = \frac{\sum_{j=1}^{M} Dif\left(h_i(\mathrm{x}_j), h^*(\mathrm{x}_j)\right)}{M}. \quad (3)$$

where $h_i(\mathrm{x}_j)$ denotes the classification of the instance x_j by the classifier h_i, and M is the number of instances in the test set.

However, this measure depends on the integration procedure. This measure is used mostly for voting or averaging types of integration. For some integration procedures (e.g., static selection) this measure cannot reflect the real disagreement between a classifier and the rest of the classifiers in the ensemble. In this study we experimented with different integration procedures, and thus we needed an integration-independent measure.

We measure the disagreement of a base classifier h_i and the whole ensemble using a diversity Div_i of the classifier on test instances as the average difference in predictions in all the pairs of classifiers including h_i as:

$$Div_i = \frac{\sum_{j=1}^{M} \sum_{k=1,k\neq i}^{S} Dif\left(h_i(\mathrm{x}_j), h_k(\mathrm{x}_j)\right)}{M \cdot (S-1)}, \quad (4)$$

where S denotes the number of the base classifiers. The *total diversity of the ensemble* is the average diversity of its members. It can be shown that in the case of numeric predictors where the simple average is used as the integration procedure, when the function *Dif* is redefined as the squared difference in predictions of two classifiers, the diversity estimate (3) is proportional to the integration-independent measure (4).

One simple alternative metric for determining the agreement of classifiers in an ensemble is *classification overlap*, which is the percentage of test instances that were classified in the same way by each of the classifiers (Brodley & Lane, 1996). In (Cunningham & Carney, 2000) it was proposed to use entropy as the appropriate measure of the *ambiguity* of classification ensembles. However, there was no research showing that this measure is better than the commonly used measure (3).

4 OUR APPROACH TO THE PROBLEM

In this chapter we present the motivation for dynamic integration of learned models, and briefly discuss the proposed approach.

In a similar way as Hansen and Salamon (1990) proved their theorem for the base classifiers with less than 50% classification error rate, it is possible to prove that, for majority voting ensembles, if the average classification error rate for an instance is more than 50% and the base classifiers in the ensemble are independent in the production of their errors, the expected ensemble classification error for that instance will approach 100% as the number of classifiers combined goes to infinity, even when the coverage approaches 100%. Moreover, in some hard learning problems and especially in multi-class problems, average local accuracy below 50% can hold on quite a big percentage of instances. The accuracy in this case can be significantly improved with an integration technique that is capable of identifying the regions of expertise of individual base classifiers instead of simple majority voting (e.g., with dynamic integration).

In Figure 5, a simple example of failure of static voting is considered. Let there be three base classifiers C_1, C_2, and C_3 generated with a boosting approach, and the ensemble is used to classify a hard instance $\mathbf{x}^*$. It is very probable that the first two classifiers will give wrong predictions, and only the last classifier, after the adaptive boosting distribution change, will classify correctly. This is a very common situation with boosting. It seems that the idea of boosting has worked, and the instance is classified correctly by the last base classifier, but the final ensemble prediction will be wrong in most cases, depending on the weights used in the weighted voting-based integration procedure. This example shows the important shortcoming of static voting approaches in that they do not take into account local expertise of the base classifiers, and thus provides the motivation for the use of dynamic integration.

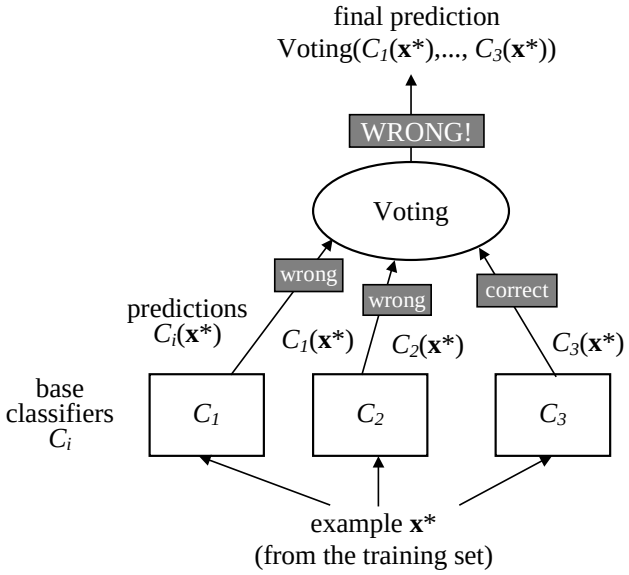

FIGURE 5 An example of classification failure with static voting in boosting

In Figure 6, a simple example of the distribution of errors of a model built by a C4.5 decision tree learning algorithm (Quinlan, 1993) over the instance space is considered. The classification problem includes two features x_1 and x_2, and two classes "class_1" and "class_2". The target function is $x_1 > x_2$. The greyed areas represent instances, which are incorrectly classified by the model. It can be seen that the errors are concentrated in triangular regions in this case.

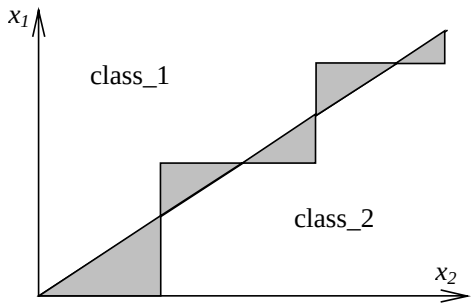

FIGURE 6 An example of the distribution of errors of a C4.5 decision tree

In (Gama, 1999) it was shown that the distribution of the error rate over the instance space is not homogeneous for many types of classifiers. Depending on the classifier, the error rate will be more concentrated on certain regions of the instance space than in others.

The basic idea of *dynamic integration* is that the information about a model's errors in the instance space can be used for learning just as the original instances were used for learning the model. In (Giacinto & Roli, 1999), a

theoretical framework of dynamic classifier selection was presented showing that the accuracy of dynamic selection approaches the accuracy of the optimal Bayesian classifier when the number of instances in the dataset grows.

In this thesis, a dynamic integration approach is presented that estimates the local accuracy of the base classifiers by analyzing their accuracy on nearby instances to the instance to be classified (Puuronen *et al.*, 1999a). Instead of directly applying selection or combination as an integration method, we use cross validation to collect information about the classification accuracies of the base classifiers and use this information to estimate the local classification accuracies for each new instance. These estimates are based on the weighted nearest neighbor classification (WNN) (Cost & Salzberg, 1993).

The proposed dynamic integration technique contains two main phases (Puuronen *et al.*, 1999a; Puuronen & Tsymbal, 2001). First, at the learning phase, the training set is partitioned into folds. During the cross validation run, we estimate the local classification errors of each base classifier for each instance of the training set according to the 1/0 loss function. These local errors together with the features of the instances of the training set form a meta-level training set used by WNN. The learning phase finishes with training the base classifiers on the whole training set. The application phase begins with determining the K-nearest neighborhood for the new instance using a distance metric based on the values of its features. Then, the meta-level classifier (WNN) is used to predict the local classification errors of each base classifier for a new instance using the meta-level training set. In WNN, to weigh the classification errors for an instance k from the K-nearest neighborhood of the test instance, we use a tri-cube function (Hastie & Tibshirani, 1996):

$$w_k = (1-(d_k / d_{K+1})^3)^3, \quad (5)$$

where d_k is the distance from the instance k to the test instance, and d_{K+1} is the distance from the test instance to the first nearest instance not included in the K-neighborhood.

We have proposed three different approaches based on the local accuracy estimates: Dynamic Selection (DS) (Puuronen *et al.*, 1999a), Dynamic Voting (DV) (Puuronen *et al.*, 1999a), and Dynamic Voting with Selection (DVS) (Tsymbal *et al.*, 2001). All these are based on the same local accuracy estimates obtained using WNN and (5). DS simply selects a classifier with the least predicted local classification error, as was also proposed in (Giacinto & Roli, 1999; Woods *et al.*, 1997). In DV, each base classifier receives a weight that is proportional to the estimated local accuracy of the base classifier, and the final classification is produced by combining the votes of each classifier with their weights. In DVS, the base classifiers with highest local classification errors are discarded (the classifiers with errors that fall into the upper half of the error interval of the base classifiers) and locally weighted voting (DV) is applied to the remaining base classifiers.

Our DS integration procedure is similar to DCS_LA (for Dynamic Classifier Selection based on Local Accuracies) presented in (Woods *et al.*, 1997), and to DCS presented in (Giacinto & Roli, 1999). The main differences between our DS and the others are as follows. (1) DCS_LA does not use distances for

weighting the local accuracy estimates. (2) Neither DCS_LA nor DCS use cross-validation to estimate the training local accuracies, estimating them directly on the training or validation datasets. (3) Both DCS_LA and DCS can be based on the local accuracy estimate, which takes into account also the classification produced by the corresponding base classifier. The local accuracy is calculated in this case only for those instances in the neighborhood that are classified into the same class as the test instance by the corresponding classifier. This local accuracy is called the *local class accuracy* in (Woods *et al.*, 1997), and *a posteriori accuracy* in (Giacinto & Roli, 1999; Giacinto & Roli, 2001), and it was shown to be slightly superior with respect to the simple local accuracy (used in our DS, DV, and DVS).

As regarding these differences, it was discussed in (Giacinto & Roli, 2001) that distance-weighted local accuracies are better as they allow us to handle more effectively the problem of the choice of the size of the neighborhood, and provide more robust estimates of local accuracies. The use of cross-validation for building the history of base classifiers' accuracy is necessary in order to get unbiased accuracy estimates, especially when there is not enough data for a separate validation set. Using cross-validation, accuracy estimates are provided for each training instance. By sacrificing efficiency, several cross-validation runs can be used in order to get more exact accuracy estimates. For some classifiers, the use of cross-validation is necessary, as the training set based estimates are too biased. This is especially true of 1-nearest neighbor and some unpruned decision trees that always have zero training-set error. When heterogeneous classifiers are being integrated, cross validation is also desirable as different classifiers have different degree of bias with respect to the resubstitution (training-set-based) error. The use of the a posteriori local accuracy within DS, DV, and DVS is an interesting issue for further research. We presume that the a posteriori local accuracy will be superior with respect to the simple local accuracy also for our DS, DV, and DVS integration procedures.

5 RESEARCH DESIGN

This chapter provides details of the experimental design and a description of the different datasets used in the experiments in terms of how many instances, the number and type of attributes and number of classes, they each have.

5.1 Research methods

Three basic *research approaches* are used in this thesis: the conceptual-theoretical approach, the constructive approach, and the experimental approach. These approaches are tightly connected and are applied in parallel. The theoretical background is exploited during the constructive work and the constructions are used for experimentation. The results of constructive and experimental work are used to refine the theory.

Accordingly, several *research methods* are applied. In the conceptual-theoretical approach, conceptual basics and formalisms of the integration of multiple data mining methods in knowledge discovery systems, and especially dynamic integration, are reviewed and discussed. A technique for the dynamic integration of data mining methods is developed. Formal methods are also used in analysing developed and existing algorithms and meta-level data generation. During the constructive part of the research, software that implements the developed theory is constructed, with an emphasis on the use of meta-level data to integrate different mining techniques in knowledge discovery. The constructive part results in a software prototype that enables one not only to discuss the aspects of the integration of multiple mining methods but also to experiment with them. In the experimental part of the research, widely available benchmark databases (artificial and real-world ones) are used to evaluate characteristics of the developed integration approach in order to receive a deeper understanding about its behaviour in different subject domains. In the next two subsections, the experiments are considered in more detail. First, general issues of the experimental design are considered, and then datasets used in experiments throughout the thesis are discussed.

5.2 Experimental design

In this section, the techniques used in the experiments throughout this dissertation are described.

To compare the developed algorithms and existing ones, 30 runs of random sampling or Monte-Carlo CV (described in Section 2.5.1 of this thesis) are used. In each run the dataset is first split into the training set and the test set by random sampling. Each time about 30 percent of the instances of the dataset are first randomly assigned to the test set. The other 70 percent of the instances are then passed to the learning algorithm where they are divided into 10 folds using stratified random sampling. In the test environment 10-fold cross-validation is applied to build the cross-validation history for the dynamic integration of base classifiers. Cross-validation is not used for generating the performance history in the experiments with the arbiter meta-learning in Article IV (Tsymbal *et al.*, 1999), and decision committee integration (bagging and boosting) in Article V (Tsymbal, 2000), as those base classifiers are being built using one learning algorithm on different subsamples of the initial dataset. In this case, the performace history is simply collected on the whole dataset. In the experiments with the arbiter meta-learning, 10-fold cross-validation was also used to compare classification accuracies of considered learning algorithms. In Article VII (Tsymbal *et al.*, 2002), the original division of the datasets into the training and test sets was used for the sake of performance comparison. Additionally, in each run, 20 percent of the instances of the training set were randomly assigned to the validation set used in the feature subsets refinement.

In the experiments, three base learning algorithms are used: PEBLS (Cost & Salzberg, 1993), C4.5 (Quinlan, 1993), and BAYES (John, 1997), all of which were considered in Section 2.4 of this thesis. PEBLS is an instance-based learning algorithm, which uses a value-difference metric, designed specially for nominal features. The three learning algorithms represent three completely different approaches to learning and are able to provide different learning biases for successful integration. Where homogeneous base classifiers are integrated (feature selection in Article II (Puuronen & Tsymbal, 2000), arbiter meta-learning in Article IV (Tsymbal *et al.*, 1999), bagging and boosting in Article V (Tsymbal, 2000), and ensemble feature selection in Article VI (Tsymbal *et al.*, 2001) and Article VII (Tsymbal & Puuronen, 2002)), the C4.5 decision tree learning is used. In the experiments, the Heterogeneous Euclidean-Overlap Metric was usually used (Wilson & Martinez, 1997). In Article III (Puuronen *et al.*, 2000), six different distance functions are compared in the framework of dynamic integration of classifiers. When needed (as for the PEBLS and BAYES algorithms), the values of continuous features are discretized dividing the interval of the values of the feature into intervals with equal length. The whole experimental environment was implemented within the MLC++ framework (the machine learning library in C++) (Kohavi *et al.*, 1996).

Significance of observed differences in the generalization performance of two algorithms is usually checked with the paired differences *t*-test based on

train / test cross-validation splits (Dietterich, 1998), as described in Section 2.5.3. Some conclusions made in this dissertation are checked also with the McNemar's test, and the test for the difference of two proportions, which are claimed to have acceptable Type 1 error (Dietterich, 1998). In this thesis, the same results were obtained with all the significance checking tests, sometimes with different levels of significance.

5.3 Datasets used in the experiments

The datasets used in the experiments were taken from the University of California at Irvine Machine Learning Repository (Blake *et al.*, 1998), except for the Acute Abdominal Pain (AAP) datasets provided by the Laboratory for System Design, Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia and the Theoretical Surgery Unit, Dept. of General and Trauma Surgery, Heinrich-Heine University Düsseldorf, Germany (Zorman *et al.*, 2001), and the Dystonia dataset taken from the Medical Research Centre of Kharkov Medical University, Ukraine (Terziyan *et al.*, 1998). The main characteristics of the datasets used in experiments throughout the thesis are presented in Table 1. The table includes the name of the dataset, the number of instances included in the dataset, the number of different classes of instances, and the numbers of different kind of features included in the instances.

The Acute Abdominal Pain (AAP) datasets represent the same problem of separating acute appendicitis (class "appendicitis"), which is a special problem of acute abdominal pain, from other diseases that cause acute abdominal pain (class "other diagnoses"). The early and accurate diagnosis of acute appendicitis is still a difficult and challenging problem in everyday clinical routine. Each dataset includes 18 parameters (features) from history-taking and clinical examination.

The Balance dataset was generated to model psychological experimental results. Each example is classified as having the balance scale tiped to the right, tiped to the left, or balanced. The attributes are the left weight, the left distance, the right weight, and the right distance. The correct way to find the class is the greater of (left-distance * left-weight) and (right-distance * right-weight). If they are equal, it is balanced.

TABLE 1 Characteristics of the datasets

Dataset	Instances	Classes	Features	
			Categorical	Numerical
Acute Abdominal Pain I	1251	2	17	1
Acute Abdominal Pain II	2279	2	17	1
Acute Abdominal Pain III	4020	2	17	1
Balance	625	3	0	4

(continues)

TABLE 1 (continues)

Breast Cancer Ljubljana	286	2	9	0
Car Evaluation	1728	4	6	0
Pima Indians Diabetes	768	2	0	8
DNA	3186	3	180	0
Dystonia	150	3	0	7
Glass Recognition	214	6	0	9
Heart Disease	270	2	8	5
Ionosphere	351	2	0	34
Iris Plants	150	3	0	4
LED	300	10	7	0
LED17	300	10	24	0
Liver Disorders	345	2	0	6
Lymphography	148	4	15	3
MONK-1	432	2	6	0
MONK-2	432	2	6	0
MONK-3	432	2	6	0
Soybean	47	4	0	35
Thyroid	215	3	0	5
Tic-Tac-Toe Endgame	958	2	9	0
Vehicle	846	4	0	18
Voting	435	2	16	0
Zoo	101	7	16	0

In the Breast Cancer Ljubljana dataset the task is to determine whether breast cancer will or will not recur. The data were originally obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia.

The Car Evaluation dataset was derived from a simple hierarchical decision model that evaluates cars according to a concept structure. The Car Evaluation dataset contains examples with the structural information removed, i.e., it directly relates a car to the six input attributes: buying, maintenance, doors, persons, lug_boot, and safety. The four classes are “unacceptable”, “acceptable”, “good”, and “very good”.

The task for the Pima Indians Diabetes dataset is to determine whether the patient shows signs of diabetes according to World Health Organization criteria. There are eight continuous features: the number of times pregnant, plasma glucose concentration, diastolic blood pressure, triceps skin fold thickness, 2-hour serum insulin, body mass index, diabetes pedigree function, and age.

The DNA dataset is drawn from the field of molecular biology. Splice junctions are points on a DNA sequence at which “superfluous” DNA is removed during protein creation. The task is to recognize exon/intron boundaries, referred to as EI sites; intron/exon boundaries, referred to as IE sites; or neither. The features provide a window of 60 nucleotides. The

The Dystonia dataset (Terziyan *et al.*, 1998) was taken from the Medical Research Centre of Kharkov Medical University, Ukraine. The task is to distinguish three types of dystonia. Dystonia is a neurological disorder marked by strong involuntary muscle spasms that cause painful and disabling twisting of the body.

In the Glass Recognition dataset the task is to identify which one of the six types of glass is present from chemical elements in a sample.

The task for the Heart Disease dataset is to distinguish the presence or absence of heart disease in patients. The features include: age, sex, chest pain type, resting blood pressure, fasting blood sugar, max heart rate, *etc*.

The Ionosphere dataset includes radar data that was collected by a system in Goose Bay, Labrador. This system consists of a phased array of 16 high-frequency antennas with a total transmitted power in the order of 6.4 kilowatts. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere. Received signals were processed using an autocorrelation function whose arguments are the time of a pulse and the pulse number. There were 17 pulse numbers for the Goose Bay system. Instances in this dataset are described by 2 attributes per pulse number, corresponding to the complex values returned by the function resulting from the complex electromagnetic signal.

The Iris Plants dataset created by R.A. Fisher is perhaps the best known database in the machine learning literature. The task is to classify iris plants into one of three iris plants varieties: Iris Setosa, Iris Versicolour, and Iris Virginica. This is an exceedingly simple domain and very low error rates have already been achieved long ago.

The LED dataset contains data about the LED display problem, where the goal is to learn to recognize decimal digits having information about whether the seven corresponding LED segments are on or off. The LED 17 dataset represents an extension of the LED display problem, with an additional 17 irrelevant attributes being added to the instance space. Their values are randomly assigned the values 0 or 1.

The Liver Disorders dataset was created by BUPA Medical Research Ltd, and the task is to predict liver disorders that might arise from excessive alcohol consumption.

The Lymphography dataset was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. There are 15 categorical and 3 numerical attributes, and the classes being predicted are: “normal find”, “metastases”, “malign lymph”, and “fibrosis”.

The MONK’s problems are a collection of three artificial binary classification problems over the same six-attribute discrete domain (a1,…,a6). All MONK’s datasets contain 432 instances without missing values, representing the full truth tables in the space of the attributes. The ”true” concepts MONK-1, MONK-2, and MONK-3 underlying each MONK’s problem are given by: (a1=a2)or(a5=1) for MONK-1, exactly two of {a1=1, a2=1, a3=1, a4=1, a5=1, a6=1} for MONK-2, and (a5=3 and a4=1)or(a5<>4 and a2<>3) for

MONK-3. MONK-3 has 5% additional noise (misclassifications) in the training set. The MONK's problems were the basis of the first international comparison of learning algorithms (Thrun *et al*., 1991).

The Soybean dataset includes data about the soybean disease diagnosis. This is a small subset of the original Soybean-large database. There are 35 numerical attributes, and 4 classes, representing soybean diseases.

In the Thyroid dataset, five laboratory tests are used to try to predict whether a patient's thyroid is in the class "euthyroidism", "hypothyroidism" or "hyperthyroidism". The diagnosis (the class label) is based on a complete medical record, including anamnesis, scan etc.

The Tic-Tac-Toe Endgame dataset encodes the complete set of possible board configurations at the end of tic-tac-toe games, where "x" is assumed to have played first. The target concept is "win for x" (i.e., true when "x" has one of 8 possible ways to create a "three-in-a-row"). The dataset contains 958 instances without missing values, each with 9 attributes, corresponding to tic-tac-toe squares and taking on 1 of 3 possible values: "x", "o", and "empty".

In the Vehicle dataset, the goal is to classify a given silhouette as one of four types of vehicles ("Opel", "Saab", "Bus", and "Van"), using a set of 18 numerical features extracted from the silhouette. The vehicle may be viewed from one of many different angles. This dataset comes from the Turing Institute, Glasgow, Scotland.

The Voting dataset includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac in 1984. The goal is build a classification model to predict the voting congressman to be either a democrat or a republican.

Zoo is a simple dataset created by Richard S. Forsyth with instances containing 17 Boolean valued-attributes, and representing 7 types of animals.

A survey of widely used learning algorithms (decision trees, neural networks, and rule-based classifiers) on twenty-nine datasets from the UCI machine learning repository is given in Eklund (1999). This survey connects the properties of datasets examined with the selection of learning algorithms. In (Lim *et al.*, 2000) twenty-two decision tree, nine statistical, and two neural network algorithms are compared on thirty-two datasets in terms of classification accuracy, training time, and (in the case of trees) number of leaves. Fifteen of the datasets are available from the UCI repository.

In (Salzberg, 1999) the use of the datasets from the UCI repository was strongly criticized. The main message in (Salzberg, 1999) is: "In conducting comparative studies, classification researchers and other data miners must be careful not to rely too heavily on stored repositories of data (such as the UCI repository) as its source of problems, because it is difficult to produce major new results using well-studied and widely shared data. … Any new experiments on these and other UCI datasets run the risk of finding "significant" results that are no more than statistical accidents." However, we do believe that nevertheless, the UCI datasets provide useful benchmark estimates that can be of great help in experimental analysis and comparisons of

learning methods, even though these results must be looked at carefully before final conclusions are made.

6 SUMMARY OF THE INCLUDED ARTICLES

This chapter provides an overview of the refereed papers included into the present thesis. These examine various aspects of dynamic integration for ensemble classifiers including, the different dynamic integration approaches, local feature selection, the effects of using different distance functions in dynamic integration, arbiter meta-learners and decision committees.

6.1 "A dynamic integration algorithm for an ensemble of classifiers"

Reference: Puuronen, S., Terziyan, V. & Tsymbal, A. 1999. A dynamic integration algorithm for an ensemble of classifiers. In Z.W.Ras & A.Skowron (Eds.) *Foundations of intelligent systems: 11th International Symposium ISMIS'99, Warsaw, Poland, Lecture Notes in Artificial Intelligence, Vol. 1609*. Berlin: Springer, 592-600.

An algorithm for the dynamic integration of classifiers is considered, which is a new variation of the stacked generalization framework, and uses a metric to locally estimate the errors of the base classifiers. Instead of training a meta-level classifier that will derive the final classification using the classifications of the base classifiers as in stacked generalization, we propose to train a meta-level classifier that will estimate the errors of the base classifiers for each new instance and then use these errors to derive the final classification. Our goal is to use each base classifier just in that subdomain for which it is most reliable. Our dynamic approach to the integration of multiple classifiers attempts to create a meta-model that describes subspaces correctly classified by corresponding base classifiers. Our approach is closely related to that considered by Koppel & Engelson (1996). The main difference between these two approaches is in the combining algorithm. In (Koppel & Engelson, 1996) the C4.5 decision tree induction algorithm was used to predict the errors of the base classifiers, whereas we use weighted nearest neighbor classification (WNN)

(Aivazyan, 1989; Cost & Salzberg, 1993), which simplifies the learning phase of the composite classifier. It is enough to calculate the performance matrix for the base classifiers during the learning phase. In the application phase, nearest neighbors of a new instance among the training instances are identified and the corresponding base classifiers' performances are used to calculate the predicted classifiers' performance for each base classifier.

Two variations of the application phase are introduced: dynamic selection (DS) and dynamic voting (DV). In the DS application phase the local classification error is predicted for each base classifier and a classifier with the smallest error is selected to make the final classification. In the DV application phase each base classifier receives a weight that is related to the local classifier's performance and the final classification is determined by combining classifier predictions with their weights. The two variations (DS and DV) of the algorithm are evaluated and compared in experiments with each other, and with weighted voting (WV) and cross-validation majority (CVM).

The experiments were conducted on three datasets from the UCI Machine Learning Repository (Blake *et al.*, 1998) with the paired differences *t*-test based on 30 random train/test splits. With two of these datasets, our algorithm significantly outperformed weighed voting and cross validation majority. The results were also supported by the McNemar's test and a test for the difference of two proportions. However, with the third dataset, results show that our algorithm does not always produce significant benefit over the other two methods. It was seen that even in that domain DS did not perform significantly worse than the other integration methods. It was discovered from the tests also that the accuracy did not vary significantly with different numbers of nearest neighbours.

6.2 "Local feature selection with dynamic integration of classifiers"

Reference: Puuronen, S. & Tsymbal, A. 2001. Local feature selection with dynamic integration of classifiers. *Fundamenta Informaticae* 47(1-2), Special Issue "Intelligent Information Systems". Amsterdam: IOS Press, 91-117. It was published as an extended version of (Tsymbal & Puuronen, 2000b).

Data, analyzed and processed in knowledge discovery and data mining is usually multidimensional and represented by a number of features. When numerous features are used, the learning process may become computationally and analytically unmanageable. For instance, many classification techniques are based on Bayes decision theory or on a nearest neighbor search, which suffer from a drastic increase in both computational complexity and classification error in high dimensions (Hall, 2000). Bellman (1961) was the first to define this phenomenon as the "*curse of dimensionality*", while working on complicated signal processing. Techniques that are efficient in low dimensions

fail to provide meaningful results when the number of dimensions goes beyond a ‘modest’ size.

Multidimensional data in data mining is sometimes feature-space heterogeneous in that different features have different importance in different subareas of the whole space (Atkeson *et al*., 1997; Cardie & Howe, 1997; Dash & Liu, 1997; Domingos, 1997; Liu & Motoda, 1998). Many methods have been proposed for the purpose of feature selection, but almost all of them ignore the fact that some features may be relevant only in context (i.e. in some regions of the instance space).

We consider a dynamic feature selection technique that uses the wrapper approach (Kohavi, 1995b), which has been found to yield better results than the filter approach for feature selection. The evaluation function in our case is the local classification accuracy obtained by applying the classification algorithm with different feature subsets. We propose to apply a technique for the dynamic integration of classifiers (Puuronen *et al*., 1999a). This allows us to determine which classifier, with which feature subset, should be applied for each new instance. This technique can also be applied in the case of implicit heterogeneity when the regions of heterogeneity cannot be defined by a simple dependency. In our method the classifiers included in the ensemble (the base classifiers) are built based on various subsets of the original feature set.

To restrict the number of feature combinations analysed, some recursive partitioning techniques or some heuristic measures can be used to discard features that are locally irrelevant with a high probability. Thus, we propose to combine our wrapper-based method with a filter approach, using it in advance to restrict the possible feature combinations.

We present our experiments with the use of the C4.5 decision tree algorithm as a filter to guide the local feature selection process conducted using the local wrapper approach with the help of the dynamic integration of classifiers. According to the experimental results, the use of the C4.5 algorithm with or without pruning for local feature selection significantly reduces the number of locally analyzed features, often without a loss in the classification accuracy. On average for many of the datasets, the classification accuracy is even higher with the guided feature selection. We have shown that the use of decision trees for guidance (local feature filtering) works quite well with datasets including only categorical features, when with less than half of the features it is possible to reach even 10% higher accuracy than with all the features. When a data set includes also numeric features, then even only 30-50% of the features are enough to reach about 5% smaller accuracy than using all the features. According to the results, the guided feature selection is better on average than the unguided one.

For some datasets, the dynamic integration (local feature selection) is clearly better than the static approaches. The relationship between dynamic selection and dynamic voting is usually similar to the relationship between static selection and static voting. In general, the results achieved are promising and show that local data mining in comparison with mining the whole space can be advantageous. In many cases, the mining accuracy is better and the time

for processing is shorter, due to the fact that only a small set of features are used to classify each new instance.

6.3 "Distance functions in dynamic integration of data mining techniques"

Reference: Puuronen, S., Tsymbal, A. & Terziyan, V. 2000. Distance functions in dynamic integration of data mining techniques. In B.V.Dasarathy (Ed.) *Data mining and knowledge discovery: theory, tools and technology II. Proceedings of SPIE, Vol.4057*. Bellingham, WA: SPIE, 22-32.

One of the most important directions in the improvement of data mining and knowledge discovery is the integration of multiple data mining techniques. In this paper we considered two algorithms that use dynamic integration: (1) an algorithm for the dynamic integration of classifiers, and (2) an algorithm for dynamic (local) feature selection that is based on dynamic classifier integration. Our main assumption in these algorithms is that each data mining technique is the most competent one within certain subareas of the application domain, and we try to estimate these competence areas so that this information could be used in the dynamic integration of the techniques.

These algorithms use an instance-based learning approach to collect information about the competence areas of the mining techniques and apply a distance function to determine how close a new instance is to each instance of the training set. The nearest instance or instances are used to predict the performance of the data mining techniques. Because the quality of the integration depends heavily on the suitability of the distance function used, the goal here is to analyze the characteristics of different distance functions. In this paper we investigate several distance functions: (1) Euclidean distance function, (2) Heterogeneous Euclidean-Overlap Metric (HEOM), (3) Discretized Value Difference Metric (DVDM), (4) Heterogeneous Value Difference Metric (HVDM), (5) Interpolated Value Difference Metric (IVDM), and (6) Windowed Value Difference Metric (WVDM) (Wettecherech, 1994; Wilson & Martinez, 1997). We analyze the effects of the use of different distance functions to the accuracy achieved by dynamic integration when the parameters describing datasets vary.

We consider the algorithm for dynamic (local) feature selection that is based on the dynamic classifier integration. Many complicated and heterogeneous data mining problems exist with a heterogeneous feature-space, where the features that are important for data mining are different in different regions of the feature space (Apte *et al*., 1997). To make the feature selection dynamic and local, we propose to apply the previously considered technique for the dynamic integration of classifiers (Puuronen *et al*., 1999a).

In this paper we also survey the six distance functions that are used in the experiments. We present experiments with the dynamic classifier integration and the dynamic feature selection, using the six distance functions.

From the experimental results with the dynamic classifier integration, one can see that the selection of the distance function really influences the accuracy of the dynamic classifier integration as we expected. Usually the classification accuracy changes on the datasets with different distances in the same way as reported by Wilson & Martinez (1997) for simple nearest neighbor classification. When a distance function performs badly in some domain with simple nearest neighbor classification in comparison with the others, then it usually performs as badly with the dynamic classifier integration.

The influence of the distance function is usually less significant in our case than in the case of direct nearest neighbor classification. It can be explained by the fact that the distance function does not influence on the base classifiers in our dynamic integration, and it influences only at the meta-level, when some base classifier is selected with the best local accuracy. The distance function influences only on defining the neighborhood, which is used for estimating the local accuracies. Dynamic Selection is always better than Dynamic Voting on average. One surprising result is that the Euclidean metric has the best accuracy on average (in combination with Dynamic Selection). It can be explained by the selection of datasets in our experiments.

In the experiments with the dynamic feature selection, the selection of the distance function also influences the accuracy of the dynamic integration, and the trends in the accuracy are usually the same as those presented by Wilson and Martinez (1997). However, IVDM and HVDM are the best metrics in this case, and it supports in a way results obtained in (Wilson and Martinez, 1997). Dynamic Selection is again always better than Dynamic Voting on average, supporting our conclusion from the previous experiment.

6.4 "Arbiter meta-learning with dynamic selection of classifiers and its experimental investigation"

Reference: Tsymbal, A., Puuronen, S. & Terziyan, V. 1999. Arbiter meta-learning with dynamic selection of classifiers and its experimental investigation. In J.Eder, I.Rozman & T.Welzer (Eds.) *Advances in databases and information systems: 3rd East-European Conference ADBIS'99, Maribor, Slovenia, Lecture Notes in Computer Science, Vol. 1691*. Berlin: Springer, 205-217.

In (Chan, 1996; Chan & Stolfo, 1997) the arbiter meta-learning technique was proposed for the parallel integration of multiple classifiers. *Meta-learning* encompasses the use of learning algorithms to learn how to integrate results from multiple learning systems. The whole dataset is partitioned into smaller subsets, and learning algorithms are applied on these subsets. This is followed by a part of the learning phase, which combines the learned results. It was shown in experiments that the accuracy would not suffer in such a scheme, as

one may presume, in comparison with learning from the entire dataset (Chan, 1996).

An *arbiter* is a classifier that is trained to resolve disagreements between the base classifiers. An arbiter is generated using the same learning algorithm that is used to train the base classifiers. When a new instance is classified in the application phase, first the base classifiers and the arbiter generate their predictions. Then an *arbitration rule* generates the final prediction using the predictions made by the base classifiers and the arbiter. One such arbitration rule that could be used to derive the final classification is as follows: return the prediction with the majority of occurrences given by the base classifiers and the arbiter. This arbitration rule is based on the widely used voting principle. In (Chan, 1996; Chan & Stolfo, 1997), a hierarchical (multi-level) meta-learning method called an *arbiter tree* was also considered. An *arbiter tree* is a hierarchical structure composed of arbiters that are computed in a bottom-up, binary-tree way and it can be generalized to arbiter trees of higher orders.

The voting technique used as the arbitration rule, however, has several shortcomings. From our point of view the most important shortcoming is that the voting technique is unable to take into account the local expertise. In order to improve the meta-learning, we propose the use of our dynamic classifier selection as the arbitration rule.

We present an experimental evaluation of the arbiter meta-learning with the dynamic classifier selection. We compare it with the simple arbiter meta-learning. Our experimental results support the findings and conclusions made in (Chan & Stolfo, 1997). All the meta-learning strategies do show a consistent improvement in the classification accuracy over the base classifiers trained on a subset of the training data. Our experimental results show also that both one-level meta-learning (*Dynamic* and *Arbiter*) and hierarchical meta-learning (*Dynamic Tree* and *Arbiter Tree*) are often able to sustain the same level of accuracy as a global classifier trained on the entire dataset. Thus meta-learning over data partitions can maintain or even boost the accuracy of a single global classifier under certain circumstances. For example, it was done by *Dynamic* on the Tic-Tac-Toe dataset, where the best base classifiers on 32 subsets had 73% accuracy, and the global classifier 87% only, but the one-level dynamic arbiter meta-learning classifier had 97% accuracy! It can be seen from the experimental results that this is a very common result.

Our experimental results confirm that maximal parallelism can be effectively exploited by the meta-learning over disjoint data partitions without a substantial loss of accuracy. Hierarchically learned classifiers can work better than single layered meta-learners under certain circumstances. For example, on the MONK-1, MONK-3, and Tic-Tac-Toe datasets the *Arbiter Tree* works significantly better than the *Arbiter*, and on the MONK-2 dataset the *Dynamic Tree* works significantly better than the *Dynamic*.

One can see from the experimental results that in some cases our dynamic meta-learning techniques are better than the corresponding simple meta-learning techniques. For example, on the MONK-1, MONK-3, and Tic-Tac-Toe

datasets the *Dynamic* is significantly better than the *Arbiter*, and on the MONK-2 dataset the *Dynamic Tree* is significantly better than the *Arbiter Tree*.

6.5 "Decision committee learning with dynamic integration of classifiers"

Reference: Tsymbal, A. 2000. Decision committee learning with dynamic integration of classifiers. In J.Štuller, J.Pokorný, B.Thalheim & Y.Masunaga (Eds.) *Current issues in databases and information systems. Proceedings of ADBIS-DASFAA 2000, Prague, Czech Republic, Lecture Notes in Computer Science, Vol. 1884*. Berlin: Springer, 265-278.

Decision committee learning has demonstrated spectacular success in reducing classification error from learned classifiers. These techniques develop a classifier in the form of a committee of subsidiary classifiers. The committee members are applied to a classification task and their individual outputs combined to create a single classification from the committee as a whole. This combination of outputs is usually performed by majority vote. Decision committee learning can be especially recommended for learning tasks where there is no prior opportunity to evaluate the relative effectiveness of alternative approaches, there is no a priori knowledge available about the domain, and the primary goal of learning is to develop a classifier with the lowest possible error (Webb, 2000).

Two decision committee learning approaches, boosting (Schapire, 1990; Schapire, 1997; Schapire, 1999) and bagging (Breiman, 1996), have received extensive attention recently. They repeatedly build different classifiers using a base learning algorithm, such as a decision tree generator, by changing the distribution of the training set. Bagging learns the constituent classifiers from bootstrap samples drawn from the training set. Boosting learns the constituent classifiers sequentially. The weights of training examples used for creating each classifier in boosting are modified based on the performance of the previous classifiers in such a way as to make the generation of the next classifier concentrate on the training examples that are misclassified by the previous classifiers. Both boosting and bagging are generic techniques that can be employed with any base learning algorithm (Quinlan, 1996).

The voting technique used to combine the outputs of committee members in bagging and boosting, however, has an important shortcoming. It is unable to take into account local expertise. When a new instance is difficult to classify, then the average classifier will give a wrong prediction, and the majority vote will more probably result in a wrong prediction. Instead of voting, dynamic integration of classifiers (Puuronen *et al.*, 1999a) can be used to overcome this problem. Dynamic integration of classifiers is based on the assumption that each committee member is best suited inside certain subareas of the whole feature space.

A technique which is a combination of dynamic voting and dynamic selection strategies, known as DVS (Dynamic Voting with Selection), is considered. According to this strategy, first, the local errors for the committee classifiers are estimated as usual. Then, the classifiers with high local errors are discarded (the classifiers with local errors that fall into the upper half of the error range of the committee). Next, locally weighted voting (DV) is applied as before to the restricted set of classifiers.

In this work, experiments using the dynamic classifier integration algorithm to combine classifiers generated with AdaBoost and Bagging decision committee learning approaches, are presented. The experiments are conducted on nine datasets taken from the UCI machine learning repository (Blake *et al.*, 1998). From the experimental results one can see that on each dataset considered at least some dynamic integration strategy works better than either Bagging or AdaBoost. Dynamic integration outperforms static voting in Bagging on 6 out of 9 datasets, and static voting in AdaBoost on 4 out of nine datasets. Commonly this holds true on the datasets for which dynamic integration is preferable to static integration (voting or cross-validated selection) as was shown by previous experiments with different ensemble generation techniques.

Dynamic integration strategies DV and DVS are better than Bagging and AdaBoost also on average. In general, the results achieved are promising and show that boosting and bagging have often significantly better accuracy with dynamic integration of classifiers than with simple voting. More experiments supporting these conclusions (with bigger committee sizes – 10 and 25, besides 5) are presented in (Tsymbal & Puuronen, 2000a).

6.6 "Ensemble feature selection with dynamic integration of classifiers"

Reference: Tsymbal, A., Puuronen, S. & Skrypnyk, I. 2001. Ensemble feature selection with dynamic integration of classifiers. In *Int. ICSC Congress on Computational Intelligence Methods and Applications CIMA'2001*, *Bangor, Wales, U.K.* 558-564.

Among successful ensemble generation approaches are instance sampling methods, and methods manipulating either the input features or the output targets. We manipulate the set of input features so that for training each base classifier only a subset of features is used. Into each feature subset only the features with the highest contextual merit measure (CM-measure) (Hong, 1997) values are selected, as defined by a preset threshold value. Each feature subset in the ensemble is selected to distinguish one class from the others in a multi-class problem, so that each ensemble includes as many base classifiers as there are classes.

The main assumption of the CM-measure-based heuristic is that features important for classification should be significantly different in their values to predict instances from different classes. The CM-measure is robust to both problems of class heterogeneity and feature-space heterogeneity (Hong, 1997). The CM-measure assigns a merit to a feature taking into account the degree to which the other features are capable to discriminate between the same instances as the given feature. In an extreme situation, if two instances of different classes differ in only one feature, then that feature is particularly valuable for classification and additional merit is assigned to it.

CM-measure was used in (Apte *et al.*, 1997) as a main component of the technique for explicit splitting of the feature space. In our study we directly apply the CM-measure for feature selection. We slightly modify the original CM-measure (Hong, 1997) so as to implement ensemble feature selection for multi-class problems.

In this paper we analyze and experiment with five different ensemble integration strategies with an emphasis on dynamic integration, applying them to the ensembles generated with the CM-based heuristic: cross-validation majority (CVM), weighted voting (WV), dynamic selection (DS), dynamic voting (DV), and dynamic voting with selection (DVS). We conduct experiments on seven multi-class problems from the UCI machine learning repository (Blake *et al.*, 1999).

The goal of the experiments is to investigate how the construction of the base classifiers with the CM-measure heuristic takes place with both static and dynamic integration approaches. In order to find the circumstances under which a particular integration method has an advantage in using the CM-based heuristic, we mark out several parameters to be adjusted. First, we analyze the threshold for the feature merit values, which defines the numbers of selected features. Then we investigate the influence of the use of cross validation in the evaluation of the base classifiers in dynamic integration. And finally, the optimal number of nearest neighbors in dynamic integration is analyzed.

As the experiments have shown, each dataset has its own optimal CM-measure threshold, but on an average, accuracy achieved with the 0.5 threshold is significantly less than with the 0.25 threshold. This can be explained by the fact that deleting extra features can lead to irreparable loss of information in some cases. On the other hand, including more features than necessary does not have such negative effect because C4.5 decision trees have embedded feature selection.

The best number of neighboring instances taken into account in dynamic integration depends on the dataset and whether or not cross validation was used. The behaviour of the dynamic strategies is changed with the use of cross validation. When no cross validation is used, for the best DS strategy, the best average accuracy is achieved with 7 neighbors, and afterwards the accuracy drops by more than 1 percent. When cross validation is used, the highest average accuracy is achieved with 63 neighbors.

The use or non-use of cross validation in dynamic integration leads to two different cases with their own advantages and shortcomings. In the first case,

we avoid the overly optimistic bias of the training set estimation by sacrificing the processing time, and in the second case we have exact information about the training set errors of the base classifiers, but the general error estimation gets overly optimistic.

6.7 "Ensemble feature selection with the simple Bayesian classification in medical diagnostics"

Reference: Tsymbal, A. & Puuronen, S. 2002. Ensemble feature selection with the simple Bayesian classification in medical diagnostics. *Int. Journal of Medical Informatics.* Elsevier Science (submitted as an extended version of the conference article, CBMS'2002).

One effective approach for generating an ensemble of accurate and diverse base classifiers is to use ensemble feature selection (Opitz, 1999). By varying the feature subsets used to generate the base classifiers it is usually possible to promote the diversity and produce base classifiers which make their classification errors in different subareas of the instance space. While traditional feature selection algorithms have the goal of finding the best feature subset that is germane to both the learning task and the selected inductive learning algorithm, the task of ensemble feature selection has the additional goal of finding a set of feature subsets that will promote disagreement among the base classifiers (Opitz, 1999).

Most of ensemble research is concentrated on base classifiers that apply a decision tree or a neural network approach. Ensembles of simple Bayesian classifiers have not been widely studied mainly because of their background assumption: that the features used to derive a classification are independent of each other, given the predicted value. Another reason is that the simple Bayesian classifier is extremely stable, and most ensemble techniques aim at reducing variance thus being unable to take the full advantage of the integration of simple Bayesian classifiers (Bauer & Kohavi, 1999). However, it has been recently shown that the simple Bayesian classifier (1) can be optimal even when the independence assumption is violated by a wide margin (Domingos & Pazzani, 1997), (2) can be effectively used with boosting (an ensemble technique which performs also bias reduction) (Elkan, 1997), and (3) can be successfully applied with ensemble feature selection (Pedersen, 2000).

The algorithm EFS_SBC (Ensemble Feature Selection with the Simple Bayesian Classification) introduced in (Tsymbal *et al.*, 2002) constructs an ensemble of simple Bayesian classifiers in random subspaces and uses hill-climbing search in a refinement cycle for improving the accuracy and diversity of the base classifiers. The algorithm is composed of two main phases: (1) construction of the initial ensemble in random subspaces; and (2) iterative refinement of the ensemble members. EFS_SBC uses the same fitness function as was used in (Opitz, 1999) in a genetic algorithm, where the fitness of a feature

subset was selected to be proportional to the classification accuracy and diversity of the corresponding classifier. The iterative refinement is based on a hill-climbing search. For all the feature subsets, each feature is tried to be switched (included or deleted). If the resulting feature subset produces better performance on the validation set, that change is kept. This process is continued until no further improvements are possible. The process usually terminates after no more than four passes through the feature set.

We have applied EFS_SBC to three large datasets in the domain of acute abdominal pain (AAP) (Zorman *et al.*, 2001), comparing the diagnostic accuracy, sensitivity, and specificity of EFS_SBC and the original simple Bayesian technique. These datasets all represent the same problem of separating acute appendicitis (class "appendicitis"), which is a special problem of acute abdominal pain, from other diseases that cause acute abdominal pain (class "other diagnoses").

In the experiments, in many cases the EFS_SBC ensembles of simple Bayesian classifiers had higher accuracies than the single "global" simple Bayesian classifier. For all of the three datasets, most of the integration techniques in EFS_SBC performed significantly better than the single simple Bayes. For the datasets AAP I and AAP III, the best results were shown by DVS, and for the dataset AAP II the best technique was DS.

Dynamic integration was in general much better than static integration for all the three datasets, better utilizing the diversity of the base classifiers. The best integration strategy on average was DVS. The achieved average of sensitivity and specificity rivalled the best previously published results for these datasets. Using information collected from the refinement cycles, we analyzed the importance of each of the features on each of the three datasets.

Two main conclusions made in (Zenobi & Cunningham, 2001) were: (1) the ensembles based on diversity have lower generalization error, and (2) the base classifiers produced focusing on diversity have less features on average than those based on error only, indicating that these base classifiers can be considered as local learners. Our experiments supported these two conclusions. In addition, we also have shown that the degree of importance of accuracy and diversity when building ensembles is different for different datasets.

6.8 About the joint articles

The present introductory part and Article V (Tsymbal, 2000) have been written solely by the author. The author of this thesis is the principal author of Article IV (Tsymbal *et al.*, 1999), Article VI (Tsymbal *et al.*, 2001), and Article VII (Tsymbal & Puuronen, 2002). The other joint papers: I (Puuronen *et al.*, 1999a), II (Puuronen & Tsymbal, 2001), and III (Puuronen *et al.*, 2000) have been written in close collaboration by the authors. All the articles included have been refereed by at least two international reviewers and published. Articles I, II, IV-VII are full-paper refereed, and Article III is extended abstract refereed. All the

included papers have been edited and revised for final publication by the author. Articles I, IV, V and VII have been presented by the author personally at the corresponding conferences and symposiums.

The software implementation of the experimental settings used and described in Arcticles I-VII, and most of the contents of experimental sections in those articles also represent the independent work done by the author. Reviews of related work in the included joint papers (e.g. Chapter 2 in Article I, Sections 2 and 4 in Article III) were also done mainly by the author.

7 CONCLUSIONS

This chapter summarises the novel contributions of the thesis, outlines some limitations and provides analysis of proposed future directions for the research.

7.1 Contributions of the thesis

This thesis presents a technique for dynamic integration of data mining methods in knowledge discovery systems. The main contributions of the present thesis are:

1. The problem of integration of multiple classifiers is considered, and a framework for the integration of classifiers, which includes two basic approaches to the integration (combination and selection), is proposed. In this framework, integration methods are also divided into static and dynamic. Relevant work in the integration of classifiers, and especially in dynamic integration, is reviewed.

2. A technique for the dynamic integration of data mining methods is developed. This technique is based on the assumption that each data mining method is best suited inside certain subareas of the whole domain area. Several possible strategies using this technique are analyzed: Dynamic Selection, Dynamic Voting, and Dynamic Voting with Selection. The experiments show that DVS is a very stable strategy that combines advantages of DS and DV. Combinations of our integration technique together with different algorithms for generating base classifiers are analyzed: bagging and boosting, arbiter meta-learning, and base classifiers built on various feature subsets.

3. An experimental investigation of the proposed technique for dynamic integration of data mining methods is made. The influence of distance function selection on the accuracy of dynamic classifier integration is evaluated. All the proposed combinations of the dynamic integration technique together with different algorithms for generating base classifiers are also experimentally evaluated. The experiments are conducted on benchmark datasets from the UCI machine learning repository, and on some real-world datasets.

4. A possible application of the dynamic integration technique within an integrated knowledge discovery management system is analyzed. A structure for the system is proposed, objectives of each subsystem are discussed, and guidelines are given on the implementation of each subsystem. The method evaluation/selection subsystem, which is the focus of the thesis, is very important, helping a user to select an appropriate data mining method. This subsystem can be based on the technique for the dynamic integration of data mining methods, which is considered in this dissertation.

5. The EFS_SBC (Ensemble Feature Selection with the Simple Bayesian Classification) algorithm is developed. It constructs an ensemble of simple Bayesian classifiers in random subspaces and uses hill-climbing search in a refinement cycle for improving the classification accuracy and diversity of the base classifiers. It was applied to datasets from the UCI machine learning repository, and to three large datasets of acute abdominal pain, rivalling the best previously published accuracy, sensitivity and specificity in many cases. It was shown that dynamic integration was in general much better than static integration for these medical datasets, and the best integration strategy was DVS.

7.2 Limitations and future work

A technique for the dynamic integration of data mining methods in knowledge discovery systems is proposed in this dissertation. Classification is a typical data mining task where the value of some attribute for a new instance is predicted based on the given collection of instances for which all the attribute values are known. A focus in the thesis is on the dynamic integration of classifiers. However, the considered algorithm for the dynamic integration of classifiers can be applied also to other data mining problems, and especially to regression, with some minor changes. This is an interesting topic for further research.

In this thesis, experimental results on benchmark datasets have been promising, but further experiments on real problems are still needed to evaluate the practical value of the proposed technique, and to analyze the characteristics of the domains that can benefit of the technique. This concerns all the combinations of the integration techniques with different algorithms for generating base classifiers: bagging and boosting, arbiter meta-learning, and base classifiers built on various feature subsets.

An application of the dynamic integration of classifiers to local feature selection is another potentially interesting and promising topic for future research, where only preliminary experiments on benchmark datasets were made. The issue of dynamic integration for parallel distributed learning is also an important and interesting topic for future work especially when considering the increasing size of datasets produced currently in modern scientific experiments. The implementation of the parallel distributed learning technique

Dynamic Tree with the Grid Computing technology, which is a computing infrastructure that provides dependable, consistent, pervasive and inexpensive access to computational capabilities, could be one step in that important direction.

The developed dynamic integration approach takes into consideration (local) accuracy of the components being integrated. However, other characteristics such as time taken, and memory used by the base models can also be important for an integration procedure in some cases as considered for example in (Nakhaeizadeh, 1997). Extension of the dynamic integration algorithm to take into account different integrated components' characteristics is also a potential topic for future research. Besides, it is also important to make an analysis of the proposed integration techniques with respect to such characteristics as time taken and memory used, and not only the classification accuracy, which was the focus of the present thesis.

The proposed dynamic integration techniques are less efficient than the static ones with respect to the time taken and memory used. Dynamic integration requires normally extra steps such as building the cross-validation history during the learning phase, and a nearest neighbor search for finding the nearest neighborhood, during the application phase. Extra memory is used for storing the meta-level methods' performance information or the cross-validation history. Indexing techniques can be used within dynamic integration to improve its efficiency by speeding up the nearest neighbor search, and this is an important topic for further research.

Another interesting topic for future research is the use of the a posteriori local accuracy (Giacinto & Roli, 1999; Giacinto & Roli, 2001) or the local class accuracy (Woods *et al.*, 1997) within DS, DV, and DVS, as discussed in Chapter 4.

REFERENCES

Aha, D.W., Kibler, D. & Albert, M.K. 1991. Instance-based learning algorithms. Machine Learning 6, 37-66.

Aivazyan, S.A. 1989. Applied statistics: classification and dimension reduction. Moscow: Finance and Statistics.

Ali, K. & Pazzani, M. 1996. Error reduction through learning multiple descriptions. Machine Learning 24, 173-202.

Anand, S.S., Scotney, B.W., Tan, M.G., McClean, S.I., Bell, D.A., Hughes, J.G. & Magill, I.C. 1997. Designing a kernel for data mining. IEEE Expert: Intelligent Systems & Their Applications 12(2), 65-74.

Apte, C., Hong, S.J., Hosking, J.R.M., Lepre, J., Pednault, E.P.D. & Rosen, B.K. 1997. Decomposition of heterogeneous classification problems. In X.Hiu, P.Cohen & M.Berthold (Eds.) Advances in intelligent data analysis (IDA-97), LNCS 1280. Berlin: Springer, 17-28.

Asker, L. & Maclin, R. 1997. Ensembles as a sequence of classifiers. In M.E.Pollack (Ed.) Proc. 15th Int. Joint Conf. on Artificial Intelligence, Nagoya, Japan. San Francisco, CA: Morgan Kaufmann, 860-865.

Atkeson, C.G., Moore, A.W. & Schaal, S. 1997. Locally weighted learning. Artificial Intelligence Review 11(1-5), 11-73.

Bauer, E. & Kohavi, R. 1999. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Machine Learning 36, 105-139.

Baxt, W. 1992. Improving the accuracy of an artificial neural network using multiple differently trained networks. Neural Computation 4, 772-780.

Bellman, R. 1961. Adaptive control processes: a guided tour. Princeton: Princeton University Press.

Blake, C.L., Keogh, E. &. Merz, C.J. 1999. UCI repository of machine learning databases [http:// www.ics.uci.edu/ ~mlearn/ MLRepository.html]. Dept. of Information and Computer Science, University of California, Irvine, CA.

Breiman, L., Friedman, J., Olshen, R. & Stone, C. 1984. Classification and regression trees. Belmont, MA: Wadsworth International Group.

Breiman, L. 1996. Bagging predictors. Machine Learning 24, 123-140.

Brodley, C. & Lane, T. 1996. Creating and exploiting coverage and diversity. In Proc. AAAI-96 Workshop on Integrating Multiple Learned Models, Portland, OR. 8-14.

Brunk, C., Kelly, J. & Kohavi, R. 1997. MineSet: an integrated system for data mining. In D.Heckerman, H.Mannila & D.Pregibon (Eds.) Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD-97), Newport Beach, CA, USA. Menlo Park, CA: AAAI Press, 37-42.

Cardie, C., & Howe, N. 1997. Improving minority class prediction using case-specific feature weights. In D.H.Fisher (Ed.) Proc. 14th Int. Conf. on Machine Learning. San Francisco, CA: Morgan Kaufmann, 57-65.

Chan, P. 1996. An extensible meta-learning approach for scalable and accurate inductive learning. Dept. of Computer Science, Columbia University, New York, NY. PhD Thesis. (Technical Report CUCS-044-96).

Chan, P. & Stolfo, S. 1997. On the accuracy of meta-learning for scalable data mining. Intelligent Information Systems 8, 5-28.

Cordella, L.P., Foggia, P., Sansone, C., Tortorella, F. & Vento, M. 1999. Reliability parameters to improve combination strategies in multi-expert systems. Pattern Analysis and Applications 2(3), 205-214.

Cost, S. & Salzberg, S. 1993. A weighted nearest neighbor algorithm for learning with symbolic features. Machine Learning 10(1), 57-78.

Cunningham, P. & Carney, J. 2000. Diversity versus quality in classification ensembles based on feature selection. In R.L.deMántaras & E.Plaza (Eds.): Proc. ECML 2000 11th European Conf. On Machine Learning, Barcelona, Spain, LNCS 1810. Berlin: Springer, 109-116.

Dash, M. & Liu, H. 1997. Feature selection for classification. Intelligent Data Analysis 1(3), 131-156.

Dietterich, T.G. & Bakiri, G. 1995. Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2, 263-286.

Dietterich, T.G. 1997. Machine learning research: four current directions. AI Magazine 18(4), 97-136.

Dietterich, T.G. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation 10(7), 1895-1923.

Domingos, P. 1997. Context-sensitive feature selection for lazy learners. Artificial Intelligence Review 11(1-5), 227-253.

Domingos, P. 1998. Knowledge discovery via multiple models. Intelligent Data Analysis 2, 187-202.

Domingos, P. & Pazzani, M. 1996. Beyond independence: conditions for the optimality of the simple Bayesian classifier. In L.Saitta (Ed.) Proc. 13th Int. Conf. on Machine Learning. San Francisco, CA: Morgan Kaufmann, 105-112.

Domingos, P. & Pazzani, M. 1997. On the optimality of the simple Bayesian classifier under zero-one loss. Machine Learning 29 (2,3), 103-130.

Duda, R. & Hart, P. 1973. Pattern classification and scene analysis. New York: Wiley.

Eklund, P.W. 1999. Comparative study of public domain supervised machine-learning accuracy on the UCI database. In B.V. Dasarathy (Ed.) Data mining and knowledge discovery: theory, tools, and technology. Proceedings of SPIE, Vol. 3695. Bellingham, WA: SPIE, 39-50.

Elkan, C.1997. Boosting and naïve Bayesian learning. Dept. of CS and Engineering, Un-ty of California, San Diego, USA. Tech. Report CS97-557.

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. 1996. Knowledge discovery and data mining: towards a unifying framework. In E.Simoudis, J.Han & U.M.Fayyad (Eds.) Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD-96), Portland, OR. Menlo Park, CA: AAAI Press, 82-88.

Fayyad, U.M., Piatetsky-Shapiro, G. & Smyth, P. 1997. From data mining to knowledge discovery: an overview. In U.M.Fayyad, G.Piatetsky-Shapiro, P.Smyth & R.Uthurusamy (Eds.) Advances in knowledge discovery and data mining. Cambridge, MA: AAAI/MIT Press, 1-29.

Friedman, J.H. 1997. On bias, variance, 0/1-loss, and the curse of dimensionality. Data Mining and Knowledge Discovery 1 (1), 55-77.

Gama, J.M.P. 1999. Combining classification algorithms. Dept. of Computer Science, University of Porto, Portugal. PhD thesis.

Gaul, W. & Säuberlich, F. 1999. Classification and positioning of data mining tools. In W.Gaul & H.Locarek-Junge (Eds.) Classification in the information age. Berlin: Springer, 143-152.

Geman, S., Bienenstock, E. & Doursat, R. 1992. Neural networks and the bias/variance dilemma. Neural Computation 4, 1-58.

Giacinto, G. & Roli, F. 1999. Methods for dynamic classifier selection. In Proc. ICIAP '99, 10th International Conference on Image Analysis and Processing. Los Alamitos, CA: IEEE CS Press, 659-664.

Giacinto, G. & Roli, F. 2001. Dynamic classifier selection based on multiple classifier behaviour. Pattern Recognition 34 (9), 1879-1881.

Hall, M.A. 2000. Correlation-based feature selection of discrete and numeric class machine learning. In Proc. Int. Conf. On Machine Learning (ICML-2000). San Francisco, CA: Morgan Kaufmann, 359-366.

Hansen, L. & Salamon, P. 1990. Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 993-1001.

Hastie, T. & Tibshirani, R. 1996. Discriminant adaptive nearest-neighbor classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (6), 607-616.

Heath, D., Kasif, S. & Salzberg, S. 1996. Committees of decision trees. In B.Gorayska & J.Mey (Eds.) Cognitive technology: in search of a humane interface. Amsterdam: Elsevier Science, 305-317.

Ho, T.K. 1998. The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(8), 832-844.

Hong, S.J. 1997. Use of contextual information for feature ranking and discretization. IEEE Transactions on Knowledge and Data Engineering 9(5), 718-730.

Imielinski, T. & Mannila, H. 1996. A database perspective on knowledge discovery. Communications of the ACM 39(11), 58-64.

John, G.H. 1997. Enhancements to the data mining process. Dept. of Computer Science, Stanford University, Stanford, USA. PhD Thesis.

Kohavi, R. 1995a. A study of cross-validation and bootstrap for accuracy estimation and model selection. In C. Mellish (Ed.) Proc. 14th Int. Joint Conf. on Artificial Intelligence IJCAI-95. San Francisco, CA: Morgan Kaufmann, 1137-1145.

Kohavi, R. 1995b. Wrappers for performance enhancement and oblivious decision graphs. Dept. of Computer Science, Stanford University, Stanford, USA. PhD Thesis.

Kohavi, R., Sommerfield, D. & Dougherty, J. 1996. Data mining using MLC++: a machine learning library in C++. In M.G.Radle (Ed.) Proc. 8th IEEE Conf. on Tools with Artificial Intelligence. Los Alamitos, CA: IEEE CS Press, 234-245.

Koppel, M. & Engelson, S. 1996. Integrating multiple classifiers by finding their areas of expertise. In AAAI-96 Workshop On Integrating Multiple Learning Models for Improving and Scaling Machine Learning Algorithms, Portland, OR. 53-58.

Krogh, A. & Vedelsby, J. 1995. Neural network ensembles, cross validation, and active learning. In G.Tesauro, D.Touretzky & T.Leen (Eds.) Advances in neural information processing systems, Vol. 7. Cambridge, MA: MIT Press, 231-238.

Lim, T.-S., Loh, W.-Y. & Shih, Y.-S. 2000. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Machine Learning 40, 203-229.

Liu, H. & Motoda, H. 1998. Feature selection for knowledge discovery and data mining. Boston: Kluwer.

Merz, C.J. 1996. Dynamical selection of learning algorithms. In D.Fisher & H.-J.Lenz (Eds.) Learning from data, artificial intelligence and statistics. New York: Springer.

Merz, C.J. 1998. Classification and regression by combining models. Dept. of Information and Computer Science, University of California, Irvine, USA. PhD Thesis.

Merz, C.J. 1999. Using correspondence analysis to combine classifiers. Machine Learning 36(1-2), 33-58.

Minsky, M. & Papert, S. 1969. Perceptrons; an introduction to computational geometry. Cambridge MA: MIT Press.

Mitchell, T. 1997. Machine Learning. New York: McGraw-Hill.

Nakhaeizadeh, G. & Schnabl, A. 1997. Development of multi-criteria metrics for evaluation of data mining algorithms. In D.Heckerman, H.Mannila & D.Pregibon (Eds.) Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD-97), Newport Beach, CA, USA. Menlo Park, CA: AAAI Press, 37-42.

Opitz, D. 1999. Feature selection for ensembles. In Proc. 16th National Conf. on Artificial Intelligence. Menlo Park, CA: AAAI Press, 379-384.

Opitz, D. & Maclin, D. 1999. Popular ensemble methods: an empirical study. Journal of Artificial Intelligence Research 11, 169-198.

Opitz, D. & Shavlik, J. 1996. Generating accurate and diverse members of a neural network ensemble. In D.Touretzky, M.Mozer & M.Hassemo (Eds.) Advances in neural information processing systems, Vol.8. Cambridge MA: MIT Press, 535-541.

Oza, N. & Tumer, K. 1999. Dimensionality reduction through classifier ensembles. Computational Sciences Division, NASA Ames Research Center, Moffet Field, CA. Technical report NASA-ARC-IC-1999-126.

Pedersen, T. 2000. A simple approach to building ensembles of naive Bayesian classifiers for word sense disambiguation. In Proc. 1st Annual Meeting of

the North American Chapter of the Association for Computational Linguistics, Seattle, WA. 63-69.

Puuronen, S., Terziyan, V. & Tsymbal, A. 1999a. A dynamic integration algorithm for an ensemble of classifiers. In Z.W.Ras & A.Skowron (Eds.) Foundations of intelligent systems: 11th Int. Symp. ISMIS'99, Warsaw, Poland. LNAI 1609. Berlin: Springer, 592-600.

Puuronen, S., Terziyan, V. & Tsymbal, A. 1999b. Dynamic integration of data mining methods using selection in a knowledge discovery management system. In M.Mohammadian (Ed.) Computational intelligence for modelling, control & automation: intelligent image processing, data analysis & information retrieval. Amsterdam: IOS Press, 272-277.

Puuronen, S., Terziyan, V., Katasonov, A. & Tsymbal, A. 1999c. Dynamic integration of multiple data mining techniques in a knowledge discovery management system. In B.V.Dasarathy (Ed.) Data mining and knowledge discovery: theory, tools and technology. Proceedings of SPIE, Vol. 3695. Bellingham, WA: SPIE, 128-139.

Puuronen, S., Tsymbal, A. & Terziyan, V. 2000. Distance functions in dynamic integration of data mining techniques. In B.V.Dasarathy (Ed.) Data mining and knowledge discovery: theory, tools and technology II. Proceedings of SPIE, Vol. 4057. Bellingham, WA: SPIE, 22-32.

Puuronen, S. & Tsymbal, A. 2001. Local feature selection with dynamic integration of classifiers. Fundamenta Informaticae 47(1-2), special issue "Intelligent Information Systems", 91-117.

Quinlan, J.R. 1986. Induction of decision trees. Machine Learning 1, 81-106.

Quinlan, J.R. 1993. C4.5 programs for machine learning. San Mateo CA: Morgan Kaufmann.

Quinlan, J.R. 1996. Bagging, boosting, and C4.5. In Proc. 13th National Conf. on Artificial Intelligence AAAI-96, Portland OR. Menlo Park, CA: AAAI Press, 725-730.

Salzberg S.L. 1999. On comparing classifiers: a critique of current research and methods. Data Mining and Knowledge Discovery 1, 1-12.

Schaffer, C. 1993. Selecting a classification method by cross-validation. Machine Learning 13, 135-143.

Schapire, R.E. 1990. The strength of weak learnability. Machine Learning 5(2), 197-227.

Schapire, R.E. 1997. Using output codes to boost multiclass learning problems. In D.H.Fisher (Ed.) Proc. 14th Int. Conf. on Machine Learning. San Francisco, CA: Morgan Kauffman, 313-321.

Schapire, R.E. 1999. A brief introduction to boosting. In T.Dean (Ed.) Proc. 16th Int. Joint Conf. on Artificial Intelligence. San Francisco, CA: Morgan Kauffman, 1401-1406.

Skalak, D.B. 1997. Combining nearest neighbor classifiers. Dept. of Computer Science, University of Massachusetts, Amherst MA. PhD Thesis.

Skrypnyk, I., Terziyan, V., Puuronen, S. & Tsymbal, A. 1999. Learning feature selection for medical databases. In Proc. 12th IEEE Symp. on Computer-

Based Medical Systems CBMS'99, Stamford CT, USA. Los Alamitos, CA: IEEE CS Press, 53-58.

Terziyan, V., Tsymbal, A., Tkachuk, A. & Puuronen, S. 1996. Intelligent medical diagnostics system based on integration of statistical methods. Informatica Medica Slovenica. Journal of Slovenian Society of Medical Informatics 3(1-3), 109-114.

Terziyan, V., Tsymbal, A. & Puuronen, S. 1998. The decision support system for telemedicine based on multiple expertise. Int. J. of Medical Informatics 49(2), 217-229.

Thrun, S.B., Bala, J, Bloedorn, E., et al. 1991. The MONK's problems – a performance comparison of different learning algorithms. Carnegie Mellon University, Pittsburg PA. Technical report CS-CMU-91-197.

Todorovski, L. & Dzeroski, S. 2000. Combining multiple models with meta decision trees. In D.A.Zighed, J.Komorowski & J.ŭytkow (Eds.) Principles of data mining and knowledge discovery. Proc. PKDD 2000, Lyon, France, LNAI 1910. Berlin: Springer, 54-64.

Tsymbal, A., Puuronen, S. & Patterson, D. 2002. Ensemble feature selection with the simple Bayesian classification. Information Fusion, Special Issue "Fusion of Multiple Classifiers". (to appear).

Tsymbal, A., Puuronen, S. & Skrypnyk, I. 2001. Ensemble feature selection with dynamic integration of classifiers. In Int. ICSC Congress on Computational Intelligence Methods and Applications CIMA'2001, Bangor, Wales, U.K. 558-564.

Tsymbal, A., Puuronen, S. & Terziyan V. 1998. Advanced dynamic selection of diagnostic methods. In Proc. 11th IEEE Symp. on Computer-Based Medical Systems CBMS'98, Lubbock TX. Los Alamitos, CA: IEEE CS Press, 50-54.

Tsymbal, A., Puuronen, S. & Terziyan, V. 1999. Arbiter meta-learning with dynamic selection of classifiers and its experimental investigation. In J.Eder, I.Rozman & T.Welzer (Eds.) Advances in databases and information systems: 3rd East-European conf. ADBIS'99, LNCS 1691. Berlin: Springer, 205-217.

Tsymbal, A. & Puuronen, S. 2000a. Bagging and boosting with dynamic integration of classifiers. In D.A.Zighed, J.Komorowski & J.ŭytkow (Eds.) Principles of data mining and knowledge discovery. Proc. PKDD-2000, Lyon, France, LNAI 1910. Berlin: Springer, 116-125.

Tsymbal, A. & Puuronen, S. 2000b. Local feature selection with dynamic integration of classifiers. In Z.W.Ras & S.Ohsuga (Eds.) Foundations of intelligent systems: 12th int. symp. ISMIS'2000, Charlotte, NC, USA, LNAI 1932. Berlin: Springer, 417-425.

Tsymbal, A. 2000. Decision committee learning with dynamic integration of classifiers. In J.Štuller, J.Pokorný, B.Thalheim & Y.Masunaga (Eds.) Current issues in databases and information systems, Proc. ADBIS-DASFAA 2000, Prague, Czech Republic, LNCS 1884. Berlin: Springer, 265-278.

Tsymbal, A. & Puuronen, S. 2002. Ensemble feature selection with the simple Bayesian classification in medical diagnostics. Int. Journal of Medical

Informatics. (submitted as an extended version of the conference article, CBMS'2002).

Tumer, K. & Ghosh, J. 1996a. Classifier combining: analytical results and implications. In Proc. AAAI-96 Workshop on Integrating Multiple Learned Models, Portland, OR. 15-21.

Tumer, K. & Ghosh, J. 1996b. Error correlation and error reduction in ensemble classifiers. Connection Science 8(3,4), Special Issue on Combining Artificial Neural Networks: Ensemble Approaches, 385-404.

Webb, G.I. 2000. MultiBoosting: a technique for combining boosting and wagging. Machine Learning 40(2), 159-196.

Wettscherech, D. 1994. A study of distance-based machine learning algorithms. Dept. of Computer Science, Oregon State University, Corvallis, OR, USA. PhD Thesis.

Wilson, D.R. & Martinez, T.R. 1997. Improved heterogeneous distance functions. Journal of Artificial Intelligence Research 6(1), 1-34.

Wolpert, D. 1992. Stacked generalization. Neural Networks 5(2), 241-259.

Woods, K., Kegelmeyer, W.P. & Bowyer, K. 1997. Combination of multiple classifiers using local accuracy estimates. IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (4), 405-410.

Zenobi, G. & Cunningham, P. 2001. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In L.D.Raedt & P.A.Flach (Eds.) Proc. ECML 2001 12th European Conf. On Machine Learning, LNCS 2167. Berlin: Springer, 576-587.

Zorman, M., Eich, H.-P., Kokol, P. & Ohmann, C. 2001. Comparison of three databases with a decision tree approach in the medical field of acute appendicitis. In V.Patel, R.Rogers & R.Haux (Eds.) Proc. 10th World Congress on Health and Medical Informatics Medinfo'2001, Vol.2. Amsterdam: IOS Press, 1414-1418.

YHTEENVETO (FINNISH SUMMARY)

Elektronisesti tallennetun tiedon määrän huima kasvuvauhti on johtanut tilanteeseen, jossa tietokannat sisältävät valtavat määrät eri sovellusalueiden tietämystä. Tiedon louhinnassa pyritään löytämään suuresta tietomassasta mielenkiintoisia säännönmukaisuuksia, joiden olomassaolosta ei olla etukäteen tietoisia. Tämänhetkiset käytettävissä olevat tiedon keräämis- ja tallennusmahdollisuudet kuitenkin ylittävät käytettävissä olevat mahdollisuudet analysoida, yhdistää ja tiivistää tietämystä tästä valtavasta tietomassasta. Viime aikoina onkin kehitetty lukuisia tiedonlouhintamenetelmiä tietämyksen muodostamiseksi laajoista tietokannoista. Sopivimman tiedonlouhintamenetelmän tai menetelmäryhmän valinta ei ole yleensä kuitenkaan mikään suoraviivainen tehtävä. Usein menetelmä valitaan staattisesti niin, että samaa menetelmää käytetään kaikille sovellusalueen uusille tapauksille analysoimatta sitä, kuinka hyvin se soveltuu kunkin uuden tapauksen luokitteluun. Yleensä on kuitenkin mahdollista saavuttaa parempi luokitustulos, jos menetelmän valinta suoritetaan dynaamisesti ottaen huomioon kunkin uuden tapauksen ominaispiirteet

Eräänä mahdollisuutena tutkijat ovat jo useiden viime vuosien ajan yrittäneet kehitellä ratkaisua, jossa muodostetaan useamman luokittelijan joukko luokittelutyyppisen tehtävän suorittamiseksi. Tutkimusten päätuloksena on havaittu, että yhdistämisellä voidaan taitavasti toimien saavuttaa yksinkertaisten luokittelijoiden tuloksia yhdistämällä parempi tulos kuin millään yksittäisellä hyvälläkään luokittelijalla yksinään. Sillä voidaan myös parantaa yksittäisen hyvän luokittelijan tuottamaa tulosta ottamalla samalla huomioon useampien heikompien luokittelijoiden tuottamat tulokset. Lähestymistavan edut ovat olleet todettavissa sekä luokittelu- että regressiotehtävissä.

Eri menetelmillä tuotettujen useampien luokittelijoiden käyttöön liittyvät keskeiset ongelmat voidaan tiivistää kahteen peruskysymykseen: 1) Millainen opittujen mallien joukko pitäisi luokittelijoina tuottaa? ja 2) Kuinka opittujen mallien tuottamat tulokset yhdistetään? Tässä väitöskirjassa keskitytään näistä jälkimmäiseen kysymykseen. Aikaisemmin on osoitettu, että luokittelijajoukon kattavuuden kasvattaminen pelkästään joukkoon sisältyvien luokittelijoiden erilaisuutta lisäämällä ei ole riittävä takaamaan parempaa luokittelun lopputulosta ellei lopputuloksen muodostava yhdistämismenetelmä kykene käyttämään kattavuutta hyväkseen.

Väitöskirjatyön perusolettamus on, että kukin luokittelija on paras sovellusalueen tietyllä osa-alueella. Parempi lopputulos on mahdollista saavuttaa, jos näiden asiantuntemusalueiden tietämys kyetään keräämään ja käyttämään hyväksi luokittelijoiden antamien tulosten yhdistämisessä kokonaistulokseksi. Keskeiseksi tutkimusongelmaksi muodostuukin näiden alueiden arviointi yhdistämisen kannalta hyödynnettävissä olevalla tavalla.

Tämän väitöskirjatutkimuksen tavoitteena onkin kehittää luokittelijoiden dynaamisen integroinnin teoreettista taustaa ja soveltamista. Tällä hetkellä suosituimmat integrointitavat ovat luokittelijajoukkoon kuuluvien luokittelijoiden

enemmistön tarjoaman ratkaisun tai keskimääräisen ratkaisun valinta. Näiden staattisten integrointitapojen heikkoutena on kuitenkin se etteivät ne kykene ottamaan huomioon luokittelijoiden asiantuntemusalueita. Työssä kehitetään ja analysoidaan kolmea dynaamisen integroinnin menettelytapaa: 1) dynaamista valintaa, 2) dynaamista äänestystä ja 3) dynaamisen valinnan ja äänestyksen yhdistelmää. Työssä analysoidaan myös luokittelijoiden integroinnissa käytettäviä etäisyysmittoja ja dynaamisen lähestymistavan soveltamista päätöskomiteoihin perustuvien lähestymistapojen ja piirteiden osajoukkojen käytön yhteydessä.

Väitöskirjassa sovelletaan käsitteellis-teoreettista, konstruktiivista ja kokeellista tutkimusotetta tiukasti toisiinsa kytkettyinä tuottaen teoreettiseen taustaan pohjautuen ohjelmistokonstruktioita, joiden avulla suoritettujen kokeilujen perusteella pyritään kehittämään taustalla olevaa teoriaa.

Keskeiset työssä saavutetut tulokset ovat:

1. Viitekehyksen muodostaminen useamman luokittelijan integroinnille. Viitekehyksen puitteissa tarkastellaan sekä staattisesta että dynaamisesta integroinnista julkaistua materiaalia.

2. Kehitetään uusi dynaaminen integrointitapa, joka perustuu luokittelijoiden asiantuntemusalueiden erilaisuusolettamukselle. Integrointitavan sovelluksina analysoidaan kolmea toteutustapaa: dynaamista valintaa, dynaamista äänestystä ja näiden kahden yhdistelmää yhdistettynä useampien olemassa olevien yksittäisten perusluokittelijoiden tuottamistapojen kanssa. Empiirisissä kokeiluissa on todettu dynaamisen valinnan ja äänestyksen yhdistämisellä saavutettavan hyvin vakaa ja niiden molempien vahvat ominaisuudet hyödyntävä integrointitulos.

3. Työssä esitettyä dynaamista integrointia ja erilaisten etäisyysmittojen käyttöä on kokeellisesti testattu sekä tutkijoiden yleisesti käyttämillä testitietokannoilla että muutamalla muulla tietokannalla. Testauksissa on myös käytetty useita yksittäisten perusluokittelijoiden tuottamistapoja.

4. Esitetään dynaamista integrointia soveltavan tiedonlouhintajärjestelmän rakenne ja sen osien toteutuksen suuntaviivat. Tällaisen järjestelmän keskeinen, tiedon louhinnan välineiden arviointia ja valintaa suorittava osajärjestelmä voi perustua työssä esitettyyn dynaamiseen integrointiin.

5. Kehitetään algoritmi piirteiden valitsemiseksi yksinkertaisten Bayesian –tyyppisten luokittelijoiden joukon muodostamiseksi. Menetelmässä tuotetaan ensimmäinen luokittelijoiden joukko satunnaisuutta hyödyntäen. Muodostettua luokittelijajoukkoa kehitetään asteittain lisäämällä siihen sisältyvien luokittelijoiden erilaisuutta kunnes luokittelun kokonaistulos ei pienellä muutoksella enää parane.

JYVÄSKYLÄ STUDIES IN COMPUTING

1 ROPPONEN, JANNE, Software risk management - foundations, principles and empirical findings. 273 p. Yhteenveto 1 p. 1999.
2 KUZMIN, DMITRI, Numerical simulation of reactive bubbly flows. 110 p. Yhteenveto 1 p. 1999.
3 KARSTEN, HELENA, Weaving tapestry: collaborative information technology and organisational change. 266 p. Yhteenveto 3 p. 2000.
4 KOSKINEN, JUSSI, Automated transient hypertext support for software maintenance. 98 p. (250 p.) Yhteenveto 1 p. 2000.
5 RISTANIEMI, TAPANI, Synchronization and blind signal processing in CDMA systems. - Synkronointi ja sokea signaalinkäsittely CDMA järjestelmässä. 112 p. Yhteenveto 1 p. 2000.
6 LAITINEN, MIKA, Mathematical modelling of conductive-radiative heat transfer. 20 p. (108 p.) Yhteenveto 1 p. 2000.
7 KOSKINEN, MINNA, Process metamodelling. Conceptual foundations and application. 213 p. Yhteenveto 1 p. 2000.
8 SMOLIANSKI, ANTON, Numerical modeling of two-fluid interfacial flows. 109 p. Yhteenveto 1 p. 2001.
9 NAHAR, NAZMUN, Information technology supported technology transfer process. A multi-site case study of high-tech enterprises. 377 p. Yhteenveto 3 p. 2001.
10 FOMIN, VLADISLAV V., The process of standard making. The case of cellular mobile telephony. - Standardin kehittämisen prosessi. Tapaustutkimus solukkoverkkoon perustuvasta matkapuhelintekniikasta. 107 p. (208 p.) Yhteenveto 1 p. 2001.
11 PÄIVÄRINTA, TERO, A genre-based approach to developing electronic document management in the organization. 190 p. Yhteenveto 1 p. 2001.
12 HÄKKINEN, ERKKI, Design, implementation and evaluation of neural data analysis environment. 229 p. Yhteenveto 1 p. 2001.
13 HIRVONEN, KULLERVO, Towards Better Employment Using Adaptive Control of Labour Costs of an Enterprise. 118 p. Yhteenveto 4 p. 2001.
14 MAJAVA, KIRSI, Optimization-based techniques for image restoration. 27 p. (142 p.) Yhteenveto 1 p. 2001.
15 SAARINEN, KARI, Near infra-red measurement based control system for thermo-mechanical refiners. 84 p. (186 p.) Yhteenveto 1 p. 2001.
16 FORSELL, MARKO, Improving Component Reuse in Software Development. 169 p. Yhteenveto 1 p. 2002.
17 VIRTANEN, PAULI, Neuro-fuzzy expert systems in financial and control engineering. 245 p. Yhteenveto 1 p. 2002.
18 KOVALAINEN, MIKKO, Computer mediated organizational memory for process control. Moving CSCW research from an idea to a product. 57 p. (146 p.) Yhteenveto 4 p. 2002.
19 HÄMÄLÄINEN, TIMO, Broadband network quality of service and pricing. 140 p. Yhteenveto 1 p. 2002.
20 MARTIKAINEN, JANNE, Efficient solvers for discretized elliptic vector-valued problems. 25 p. (109 p.) Yhteenveto 1 p. 2002.
21 MURSU, ANJA, Information systems development in developing countries. Risk management and sustainability analysis in Nigerian software companies. 296 p. Yhteenveto 3 p. 2002.
22 SELEZNYOV, ALEXANDR, An anomaly intrusion detection system based on intelligent user recognition. 186 p. Yhteenveto 3 p. 2002.
23 LENSU, ANSSI, Computationally intelligent methods for qualitative data analysis. 57 p. (180 p.) Yhteenveto 1 p. 2002.
24 RYABOV, VLADIMIR, Handling imperfect temporal relations. 75 p. (145 p.) Yhteenveto 2 p. 2002.
25 TSYMBAL, ALEXEY, Dynamic integration of data mining methods in knowledge discovery systems. 69 p. (170 p.) Yhteenveto 2 p. 2002.