HENRY JOUTSIJOKI

Variations on a Theme

The Classification of Benthic Macroinvertebrates

ACADEMIC DISSERTATION
To be presented, with the permission of
the board of the School of Information Sciences of the University of Tampere,
for public discussion in the Auditorium Pinni B 1100,
Kanslerinrinne 1, Tampere, on November 9th, 2012, at 12 o'clock.

UNIVERSITY OF TAMPERE

UNIVERSITY
OF TAMPERE

ACADEMIC DISSERTATION
University of Tampere
School of Information Sciences
Finland

Distribution
Bookshop TAJU
P.O. Box 617
33014 University of Tampere
Finland

Tel. +358 40 190 9800
taju@uta.fi
www.uta.fi/taju
http://granum.uta.fi

Cover design by
Mikko Reinikka

Acta Universitatis Tamperensis 1777
ISBN 978-951-44-8952-5 (print)
ISSN-L 1455-1616
ISSN 1455-1616

Acta Electronica Universitatis Tamperensis 1251
ISBN 978-951-44-8953-2 (pdf)
ISSN 1456-954X
http://acta.uta.fi

Tampereen Yliopistopaino Oy – Juvenes Print
Tampere 2012

Abstract

This thesis focused on the classification of benthic macroinvertebrates by using machine learning methods. Special emphasis was placed on multi-class extensions of Support Vector Machines (SVMs). Benthic macroinvertebrates are used in biomonitoring due to their properties to react to changes in water quality. The use of benthic macroinvertebrates in biomonitoring requires a large number of collected samples. Traditionally benthic macroinvertebrates are separated and identified manually one by one from samples collected by biologists. This, however, is a time-consuming and expensive approach. By the automation of the identification process time and money would be saved and more extensive biomonitoring would be possible. The aim of the thesis was to examine what classification method would be the most appropriate for automated benthic macroinvertebrate classification. Two datasets were used in the thesis. One dataset contained benthic macroinvertebrate images from eight taxonomic groups and the other images from 50 species of benthic macroinvertebrates. The thesis produced several novel results. Firstly, a new tie situation resolving strategy was introduced when one-vs-one SVM together with majority voting method was used. Secondly, a novel approach to parameter selection for SVMs was proposed. Thirdly, a new approach to class division problem in Half-Against-Half SVMs was developed by applying Scatter method. Lastly, a new classification method called Directed Acyclic Graph k-Nearest Neighbour was introduced. In this thesis altogether four multi-class extensions of support vector machines and 12 other classification methods were used. SVMs were tested with seven kernel functions, and several feature sets were used in the tests. SVMs were very suitable for the benthic macroinvertebrate classification. With the smaller dataset one-vs-one method achieved over 97% accuracy and half-against-half support vector machine achieved around 96% accuracy. Eleven classification methods other than multi-class support vector machines were tested with the smaller dataset. Of these methods the best ones were Quadratic Discriminant Analysis, Multi-Layer Perceptron and Radial Basis Function network. These methods attained around 94% accuracy. The larger dataset was tested with two classification methods. The accuracies achieved with these methods were around 80%. According to the classification results support vector machines are suitable for automated benthic macroinvertebrate classification when a proper feature set, kernel function and optimal parameter values are found.

Keywords: Machine learning · Support Vector Machine · Classification · Benthic macroinvertebrate · Water quality

Acknowledgements

I wish to thank my supervisor, Professor Martti Juhola, Ph.D., for his excellent guidance and all the support which he has given during these three years. Moreover, I am grateful to Jyrki Rasku, Ph.D., who suggested that I should undertake Ph.D. studies in Computer Sciences. This thesis would not have been completed without the financial support of Tampere Graduate School in Information Science and Engineering (TISE) which enabled me to conduct full-time research of three years.

I am also very grateful to the Finnish Environmental Institute, Jyväskylä, for the preprocessed data and to Kristian Meissner, Ph.D., for all the help with the biological issues. Moreover, I wish to thank Tuomas Turpeinen, M.Sc., for his help with the technical details concerning datasets and the preprocessing of the data.

I have had a great privilege to work in the School of Information Sciences and at the Data Analysis Research Group (DARG). More specifically, Jorma Laurikkala, Ph.D., Kati Iltanen, Ph.D., Jyri Saarikoski, M.Sc., and Kirsi Varpa, M.Sc., have given ample help. I wish also to thank other members of DARG. Furthermore, I thank Turkka Näppilä, M.Sc., for the conversations. I wish also to thank the helpful administration.

I changed my major subject from Mathematics to Computer Sciences in 2010. The change from theoretical Mathematics to more practical Computer Sciences was not easy, but knowledge of Mathematics helped me very much. Especially, I want to thank Docent Jorma Merikoski, Ph.D., who was the supervisor of my Master's Thesis and Licentiate Thesis for all his guidance. I wish to thank Docent Pentti Haukkanen, Ph.D., Antti Kuusisto, Ph.D., Mika Mattila, M.Sc., Jonni Virtema, M.Sc., and Juha Jokinen, LL.M., for all the interesting discussions during these years. Furthermore, I wish to thank Arto Luoma, Ph.D., for the statistical guidance. I wish to give special thanks to Jori Mäntysalo from who I have asked several questions concerning LaTeX and Matlab.

Finally, I want to thank Virginia Mattila, M.A., who kindly helped me with my English language. I want to thank also Docent Juhani Sarsila, Ph.D., for all the knowledge imparted concerning the Latin language and classical heritage. I also want to thank my parents and my sister for all their support during the studies.

Grates agendae

Exitus acta probat, ut visum est Ovidio, poetae illi Romano antiquo. Gratias quam maximas ago Professori Martti Juhola, qui operis mei praeses nominatus cursum meum per omnes hos annos consilio et ratione moderatus est. Magno gaudio affectus gratias agere volo etiam Domino Jyrki Rasku; qui me prudentia usus admonuit, ut dissertationem doctoralem de scientia computatoria peragrarem. Crediderim thesin meam in lucem numquam proferri potuisse, nisi Schola Tamperensis Scientiae Informaticae (TISE) mihi benevola subsidium tribuisset; quo factum est, ut per tres annos thesi scribendae me totum dedere possem.

Ab imo pectore gratias ago et Instituto a Rebus Circumiectalibus Fennico, in urbe Granivico sito, pro praeprocessu, ut dicitur, datorum, et Domino Kristian Meissner pro auxilio in rebus ad biologicam pertintentibus. Laudibus dignissimus est etiam Dominus Tuomas Turpeinen pro auxilio mihi lato in singulis rebus technicis, quae ad compositionem ac praeprocessum pertinent datorum.

Iure praecipuo donatus in Schola Scientiarum Informaticarum inque Grege ad Analysin Datorum Investigandorum (DARG) laboravi. Adiuverunt me ibi inprimis Dominus Jorma Laurikkala, Domina Kati Iltanen, Dominus Jyri Saarikoski, Domina Kirsi Varpa. Aliis quoque sociis Gregis illius nomine DARG gratias agendas debeo. Pro disputationibus frequentibus Domino Turkka Näppilä grates ago. Humillimo animo et administrationi eiusque servis fidelibus iustas ago gratias.

Accidit, ut anno 2010 mathematica theoretica relicta in scientias computatorias, ut ad usum magis accommodatas, transirem. Quam disciplinam mutandam, ut confiteor, vix facilem tunc temporis sentiebam. Sed paulo post persuasum est mihi, quanta utilitate mathematica mihi fuisset essetque futura. Plurimi aestimo gratibusque orno Dominum Jorma Merikoski, praesidem sive moderatorem operum mihi peragenti studia primo ad magistri deinde licentiati examina pertinentia. Memoria gratissima repeto disputationes iucundas cum sociis meis, quorum in numero sunt Domini Pentti Haukkanen, Antti Kuusisto, Mika Mattila, Jonni Virtema, Juha Jokinen. In rebus statisticis mihi magnopere auxiliatus est Dominus Arto Luoma, cui debitas ago gratias. Hoc loco non debet omitti Dominus Jori Mäntysalo. Is sententiam persaepe rogatus me certiorem fecit, quid notiones LaTeX et Matlab pro parte quadam reconditae sibi vellent.

Denique grates ago Dominae Virginia Mattila, quae benignissime me adiuvit, ut scientia linguae Anglicae mihi melior redderetur. Laude non immerita dignus est etiam Dominus Juhani Sarsila, quo duce linguae Latinae necnon culturae ex antiquitate traditae studui. Ex eo tempore, quo studiis

academicis me dedideram, praesidio mihi fuerunt et mei parentes et soror mea. His vero sincereque gratias ab imo corde ago.

Table of Contents

List of Abbreviations

Abbreviation	Description
ACC	Accuracy
DAGSVM	Directed Acyclic Graph Support Vector Machine
DAGKNN	Directed Acyclic Graph k-Nearest Neighbour
DDAG	Decision Directed Acyclic Graph
HAH SVM	Half-Against-Half Support Vector Machine
k-NN	k-Nearest Neighbour
LDA	Linear Discriminant Analysis
LS-SVM	Least Squares Support Vector Machine
MLP	Multi-Layer Perceptron
MMDC	Minimum Mahalanobis Distance Classifier
MNLR	Multinomial Logistic Regression
NB	Naïve Bayes
OVA	One-vs-All
OVO	One-vs-One
QDA	Quadratic Discriminant Analysis
QP	Quadratic Programming
RBFN	Radial Basis Function network
SOM	Self-Organizing Map
SVM	Support Vector Machine

Publications

I. H. Joutsijoki and M. Juhola. Kernel selection in multi-class support vector machines and its consequence to the number of ties in majority voting method. *Accepted to Artificial Intelligence Review*, 2011.

II. H. Joutsijoki and M. Juhola, Comparing the one-vs-one and one-vs-all methods in benthic macroinvertebrate image classification. In *Proceedings of the 7th International Conference on Machine Learning and Data Mining (MLDM 2011)*. Springer *Lecture Notes in Artificial Intelligence*, 6871, pp. 399–413, 2011.

III. H. Joutsijoki. Half-against-half multi-class support vector machines in classification of benthic macroinvertebrate images. In *Proceedings of 2012 International Conference on Computer and Information Science (ICCIS2012)*, IEEE, Vol. 1, pp. 414–419, 2012.

IV. H. Joutsijoki and M. Juhola. A survey of classification methods in automated taxa identification of benthic macroinvertebrates. Submitted to *Neural Computing and Applications*.

V. H. Joutsijoki and M. Juhola. DAGSVM vs. DAGKNN: An experimental case study with benthic macroinvertebrate dataset. In *Proceedings of the 8th International Conference on Machine Learning and Data Mining (MLDM 2012)*. Springer *Lecture Notes in Artificial Intelligence*, 7376, pp. 439–453, 2012.

Chapter 1

Introduction

Automated taxa identication of insects has long been a dream of taxonomists. However, it has encountered many obstacles. Gaston and O'Neill [20] considered the following to be some of the reasons why automated species identification has not come into general use:

- It is too difficult.
- It is too labour intensive.
- It is too threatening.
- It is too different.
- It is too costly.

Despite the problems encountered, some systems have been developed for automated species identification. ABIS, for instance, was made for automated bee identification [20, 42]. DAISY system was developed for general-purpose identification and it has been applied to several arthropod classification [20, 42, 51]. Moreover, ALIS [20] was made for leafhopper identification and SPIDA [42] for spider identification. The most recent system, BugID [50, 69], has been developed for benthic invertebrate identification.

This thesis focused on the classification of benthic macroinvertebrates. Generally speaking, automated benthic macroinvertebrate classification has gained scant attention [76] compared to applications such as fingerprint identification, handwritten digit recognition, or face recognition. However, there are studies on aquatic insect classification. In [42, 43, 50, 53, 69] stonefly identification was examined when in [44] insect species classification from EPT orders (Ephemeroptera, Plecoptera, and Trichoptera) was tested. Moreover,

in [33, 37, 38, 39, 76, 89] the classification of benthic macroinvertebrates was investigated.

The classification of benthic macroinvertebrates reverts to image classification. Preprocessing of images such as feature extraction using ImageJ [32] and other image processing stages are beyond the scope of this thesis and the classification itself was made according to completely preprocessed data, where the features were extracted from images. In this thesis two preprocessed datasets were used. One dataset contains data from 1,350 images, which are altogether from eight taxonomical groups of benthic macroinvertebrates. The other dataset includes data from 4,868 images, which are from 50 taxonomic groups of benthic macroinvertebrates. The smaller dataset was also used in [33, 37, 38, 39, 89]. In [37, 38, 39] the emphasis was on the use of artificial neural networks such as Multi-Layer Perceptron (MLP) and Radial Basis Function network (RBFN). Support Vector Machines were applied to benthic macroinvertebrate classification in [33, 38]. Bayes classifier, decision tree, random forest and random Bayes forest were used in [89].

The thesis consists of five publications, where in Publications I-IV smaller dataset was used and in Publication V the larger dataset was examined. This thesis addressed three research problems. Since the main focus was on applying SVM to benthic macroinvertebrate classification, the first research problem was to compare SVM with other classification methods. SVM was used in Publications I-III and V, whereas Publication IV addressed other classification methods. In Publication IV 11 classification methods were investigated in benthic macroinvertebrate classfication. These were k-Nearest Neigbour method (k-NN), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Minimum Mahalanobis Distance Classifier (MMDC), Classification Tree (CT), Multinomial Logistic Regression (MNLR), Naïve Bayes (NB), K-Means, Self-Organizing Map (SOM), MLP and RBFN.

The second research problem was to investigate which multi-class extension of SVM would be the most suitable for benthic macroinvertebrate classification. Existing multi-class extensions for SVM were investigated in Publications I-III and V. In Publication I One-vs-One (OVO) method [70] was applied together with majority voting method. Publication II was a straightforward continuation to Publication I. Publication II compared OVO and One-vs-All (OVA) [70] methods with each other. Because OVO with a majority voting method sometimes produces ties, a new tie situation solving strategy was introduced in Publication I. The solving strategy used k-NN with $k = 1$ to solve the tie situations. In OVA method tie situations are also possible, thus the same solving strategy was also used to resolve ties in Publication II. Moreover, in Publication II a new approach to parameter

selection was introduced which was an alternative to typically used accuracy.

In Publication III HAH SVM [46] was applied to the benthic macroinvertebrate classification. This was the first time that HAH SVM was used in this application. The greatest theoretical problem in HAH SVM is to find the optimal class division in a node and in [46] only a suboptimal solution based on hierarchical clustering was given. In Publication III the class division problem was solved by means of Scatter method [34, 72, 73] and it was used in every node. Platt et al. [63] introduced DAGSVM and it was applied to benthic macroinvertebrate classification in Publication V, investigating the larger dataset. DAGSVM was used for the smaller dataset in [33]. Moreover, a classification method called DAGKNN was introduced and combined the DDAG learning architecture and k-NN classification method.

The third research problem was to find the right kernel function for SVMs and the most suitable feature set to the benthic macroinvertebrate classification. In Publications I-III and V seven kernel functions were used. These were: linear, polynomial kernel functions (degrees 2-5), Radial Basis Function and Sigmoid kernel function. The smaller dataset contained 25 features and the larger dataset 32 features. Several feature subsets were used in Publications I-V. In Publication I 15D (the union of statistical and geometrical features) and 24D feature sets were used. Statistical features (7D), geometrical features (8D) and a random eight-feature subset from 15D were tested in Publication II. In Publication III four different feature sets were used and these were 7D, 15D, 17D (randomly chosen) and 25D. For Publication IV only 15D feature set was used and in Publication V the 50 species dataset was divided into ten species groups and for each group the feature selection was made by Scatter method. The use of Scatter method in the feature selection was a novel approach in this application. In [37, 38, 39, 89] 15D feature set was used as a basis for the experimental tests.

Benthic macroinvertebrate classification is a demanding task. The differences between images may be very small and the positions and the sizes may vary in each image. Moreover, images may contain overlapping or damaged specimens with lack of antennae or legs for instance. Hence the classification of benthic macroinvertebrates requires a lot of the classification methods themself. They need to be reliable and at the same time efficient.

The automated classification of benthic macroinvertebrates is not only an interesting application from the perspective of data mining and machine learning, but it would be a valuable tool in practice. Benthic macroinvertebrates are organisms without a backbone. They inhabit the bottom substrates of their habitats for at least part of their life cycle [86]. Common habitats for benthic macroinvertebrates are rivers, streams, lakes and ponds. The practical importance of benthic macroinvertebrates occurs in several sit-

uations. Firstly, benthic macroinvertebrates are an important food source for fish, and, are therefore indirectly important for humans. Secondly, benthic macroinvertebrates are commonly used in biomonitoring, which means the use of biological responses to assess changes due to anthropogenic causes [86]. The aim of biomonitoring regarding benthic macroinvertebrates is to assess water quality. The reason behind the use of benthic macroinvertebrates in biomonitoring can be explained by the fact that several species of benthic macroinvertebrates are sensitive to changes in water quality, making them excellent indicators of the health of freshwater ecosystems. Moreover, even the most minute changes in water quality can be seen in them after long periods of time, so they can be used for long-term biomonitoring.

The use of benthic macroinvertebrates in water quality assessment requires that the number of samples collected from the freshwater areas is large. Collection of benthic macroinvertebrate samples itself is a relatively easy task and inexpensive [42]. The most laborious task is their sorting and identification. The specimens are manually separated from the samples and identified individually. This traditional human-made classification is time-consuming [76]. The human-made identification process may take several days or weeks. Moreover, it demands special expertise (taxonomists) since many benthic macroinvertebrate species are difficult to distinguish from each other. Hence the total costs of biomonitoring can be very high (see for example [2]). By the automation of the benthic macroinvertebrate classification costs can be significantly reduced and it would enable large-scale biomonitoring [42].

The introductory part of this thesis is divided as follows. Chapter 2 depicts the biological motivation for the thesis. It gives a brief overview of benthic macroinvertebrates and their use in biomonitoring. Chapter 3 concentrates on several key issues in machine learning, such as feature selection and evaluation. In Chapter 4 Support Vector Machine is introduced in binary classification problems and its multi-class extensions, used in Publications I-III and V, are presented. Results from the individual publications are considered in Chapter 5. Chapter 6 is reserved for discussion and conclusions and Chapter 7 clarifies personal contribution of the publications.

Chapter 2

Biological Background

2.1 Benthic Macroinvertebrates

Freshwater ecosystems include a wide diversity of organisms. One large and important group in freshwater ecosystems is benthic macroinvertebrates. The name benthic macroinvertebrate can be divided into three parts linguistically. Firstly, benthic means bottom. Secondly, macro means large and, thirdly, invertebrate signifies being without a backbone. Hence, benthic macroinvertebrates could be characterized as animals without a backbone living on the river beds. However, this is not a very precise definition. A more accurate definition is given in [86]:

> Benthic macroinvertebrates are organisms without backbones that inhabit the bottom substrates of their habitats, for at least part of their life cycle.

Benthic macroinvertebrates are a diverse group of organisms. There are thousands of species of benthic macroinvertebrates and the exact number of species is impossible to determine since from time to time new species are found.

Some classification of benthic macroinvertebrates can be made by comparing the length of their life cycles. One of the largest and the most long-lived benthic macroinvertebrate is the river pearl mussel which can have a life cycle of several decades [66]. Crayfish instead may have a life cycle of several years [66], but the typical life cycle of benthic macroinvertebrates ranges from 1-2 years [76]. The life cycle of most benthic macroinvertebrates can be divided into two groups [17]:

1. Complete metamorphosis
2. Incomplete metamorphosis.

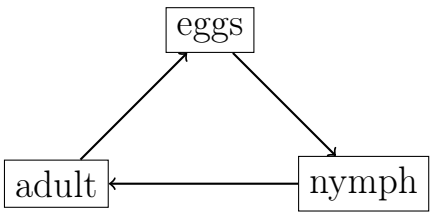
In complete metamorphosis insects at the larvae stage differ significantly from adults. The transformation to adult occurs at the pupae stage. A diagram of the phases of complete and incomplete metamorphosis can be found in Figure 2.1.

Benthic macroinvertebrates can be classified with four different approaches [66]:

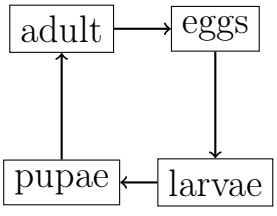
1. Structural, i.e., morphological properties
2. Functional feeding groups
3. Habitat requirements
4. Biome.

Morphological properties are the most common and the oldest way to classify benthic macroinvertebrates and the taxonomy of benthic macroinvertebrates has been developed from this approach [66]. Taxonomy originated in ancient Greece and its modern formulation goes back to the 18th century, when Linnaeus introduced the basis of binomial classification (genus is presented first and then the species) which is still used in biology [21]. It has been used ever since with some modification during the centuries. When a detailed classification of benthic macroinvertebrates is made, the concept of taxon is needed.

Taxon (plural taxa) refers to any attained determination unit (taxonomic category) for animals (or plants) in taxonomy [66]. Taxonomy is a hierarchical structure of different taxa. The goal in identification is to achieve maximum accuracy. The most accurate level in the taxonomy of animals is species. In plant taxonomy it is possible also to determine subspecies. In practice the identification of benthic macroinvertebrates is made by biologists or taxonomists specialized in insect identification. For example, *Hydropsyche siltalai* has been identified to species level, but often the identication is left



(a) Incomplete metamorphosis



(b) Complete metamorphosis

Figure 2.1: Stages of incomplete and complete metamorphosis.

Table 2.1: Hierarchy of taxonomic ranks and an example *Hydropsyche siltalai* of how to use them.

Taxon	Taxon in Latin	Example
Kingdom	Regnum	*Animalia*
Phylum	Phylum, divisio	*Arthropoda*
Class	Classis	*Insecta*
Order	Ordo	*Tricophtera*
Family	Familia	*Hydropsychoidea*
Genus	Genus	*Hydropsyche*
Species	Species	*Hydopsyche siltalai*

at a more general level such as genus or family, since the differences between species (or genera) may be very slight and the exact species (genus) may be impossible to determine. Table 2.1 presents the hierarchy of taxonomy [17] and an example of its use in the case of the *Hydropsyche siltalai* species. In Table 2.1 species level is the most accurate taxon and kingdom is the most general taxon.

A commonly seen abbreviation in benthic macroinvertebrate identification or more generally in insect identification is sp. (species) or in plural spp. (species pluralis). This means that the identification has been left at genus rank. For instance, the smaller dataset of benthic macroinvertebrates examined contains one taxonomic group, *Isoperla* sp., where this abbreviation occurs. This means that although the genus *Isoperla* is known, the exact species could not be determined. If the abbreviation spp. occurs, it means that from the same genus several species have been found without being more accurately determined.

A noticeable structural property is also the size of an animal. Hence, a dichotomy between the benthic macroinvertebrates and benthic meioinvertebrates can be established [66]. The difference between these two concepts is that benthic macroinvertebrates are that subset of the benthic animals which can be seen with the naked eye. Benthic meioinvertebrates can be seen only with a microscope and they may consist of a totally separate species or the early stages of benthic macroinvertebrates [66]. The size difference between benthic macroinvertebrates and benthic meioinvertebrates is that the former are collected with a 500μm sieve [66] and the latter are with finer sieves. Figure 2.2 presents an example image from eight taxonomic groups of benthic macroinvertebrates. The images are from the smaller dataset.

Functional feeding groups are another way to classify benthic macroinvertebrates and they are divided into four categories [17, 45, 66]:

1. Shredders
2. Collectors
3. Scrapers
4. Predators.

Shredders consume coarse non-living organic matter such as leaves [17, 66]. Shredders break down the organic matter into a convient form so it can be handled by other benthic macroinvertebrate groups. The main growth period is in late autumn and in winter when there is an abundance of food available in the rivers [66]. Shredders are found in the upper course of a river more often than the lower course [66]. Examples of shredders are, for instance, *Limnephilidae* and *Lepidostomatidae* families, which belong to the *Trichoptera* order, i.e., to the order of caddisflies [66].

Figure 2.2: An example image from eight taxonomic groups of benthic macroinvertebrates. The order of taxonomic groups from top left to bottom right is *Baetis rhodani, Diura nanseni, Heptagenia sulphurea, Hydropsyche pellucidulla, Hydropsyche siltalai, Isoperla* sp., *Rhyacophila nubila* and *Taeniopteryx nebulosa.*

Collectors can be divided into two groups:

1. Gathering collectors
2. Filtering collectors.

Gathering collectors collect fine particulate non-living organic matter from the river bed on which surface bacteria live [66]. Gathering collectors are common in middle and lower courses where there is usually an abundance of fine pieces of organic matter. Several mayflies, for example, belong to the group of gathering collectors [66]. Filtering collectors catch non-living and living fine organic matter which drift with the current [66]. Moreover, filtering collectors have several kinds of structural and behavioural adaptations which help them in nutrition. As an example, one of the most important filtering collector family in Finnish rivers is *Hydropsychidae* [66, 85]. Species *Hydropsyche pellucidulla* and *Hydropsyche siltalai*, for instance, belong to the *Hydropsychidae* family and are in the smaller dataset.

Scrapers feed on benthic algae from the solid substrates of the river bed [66]. Scrapers' richness is at its highest point in summer when there is an abundance of benthic algae. Scrapers are usually adapted in their morphological and behavioural properties to stay in place at the solid surfaces despite a strong current [66]. Many mayflies and snails are scrapers. The last functional feeding group is predators. They feed on other invertebrates either seizing them directly or with the assistance of traps [66]. Some stonefly and caddisfly species are predators. For the predators the larvae of blackflies and chironomids are important food source [66, 86].

The third alternative for the classification of benthic macroinvertebrates is the habitat requirements. There are various requirements for a habitat in the case of benthic macroinvertebrates. In [66] five of these were listed:

1. Lithophilious
2. Psammophilious
3. Burrowing
4. Xylophilous
5. Phytophilous.

Some stonefly species favour lithophilious habitat while some dragonfly species favour more xylophilous habitat [66]. The diversity exploration of river habitats is dependent on the situation and different kinds of subhabitats can be distinguished. From vegetation, for instance, smaller habitats can be distinguished such as mosses, benthic algae or reeds and each one of these smaller habitats has its own species. If we want to determine the habitat at its most general level, we can use the concept of biome, which is the fourth possible way to classify benthic macroinvertebrates [66]. Nowadays, within a biome ecoregions are separated. As an example in Finland, which belongs to the

boreal region, three ecoregions can be distinguished. These are the southern, middle, and nothern boreal regions [66].

2.2 Biomonitoring

Environmental issues have become an important part of modern society. Continually growing needs in all sectors of society pose a challenge for the environment to adapt to these changes. Water is at the centre of our lives but it is often taken for granted. Water is essential for all organisms, including humans, to survive [58]. Earth's water supplies can be divided roughly into two parts: salt water areas and freshwater areas. Freshwater is in the minority when taking account all water resources in Earth. Oki and Kanae state in [58] that only about 2.5% of Earth's water resources are freshwater. Moreover, a major part of the freshwater resources is stored as glaciers or located in deep groundwater [58]. Thus only a small fraction of freshwater is at our disposal. Due to the global population explosion, industry and agriculture, for instance, there is a constantly growing need for freshwater.

The rising trend in water demand has inconvenient consequences. Illegal dumping, oil emissions and pollution such as heavy metals, are a constant threat to freshwater ecosystems such as rivers, lakes, ponds and streams. Hence biological monitoring or more simply biomonitoring is needed. Biomonitoring can be defined as the use of biological responses to assess changes of anthropogenic origin [86]. When benthic macroinvertebrates are used in biomonitoring, the aim is to assess water quality.

Water quality assessment can be done through short-term or long-term studies. From short-term studies chemical samples are typically used. However, chemical samples give researchers only a snapshot of the water quality [76]. Due to the intermediate length of life cycle benthic macroinvertebrates are suitable for long-term biomonitoring [76]. According to Voelz [83] long-term studies are crucial for ascertaining the differences between natural and anthropogenic changes in water quality. Benthic macroinvertebrates reveal some biochemical, genetic, morphological or physiological changes when they are affected by anthropogenic stressors [86]. This is one reason why benthic macroinvertebrates should be used in biomonitoring. In the literature are numerous studies using benthic macroinvertebrate communities as indicators of environmental stressors [76] (see for example [11, 12, 22, 23, 28, 52, 57, 83]).

More specificlly, benthic macroinvertebrates have several advantages for use in biomonitoring. Vuori et al. [84] as well as [23], [66] and [86] listed some of the advantages of using benthic macroinvertebrates in biomonitoring:

1. Benthic macroinvertebrates are small enough to be easily collected.

Sampling requires only few people and minimal equipment.

2. Benthic macroinvertebrates can be defined fairly easily in several taxonomical groups.

3. Benthic macroinvertebrates are an important food source for fish thus having an essential meaning for people.

4. Sampling methods are highly standardized and different methods have been developed for special needs.

5. Benthic macroinvertebrates are common in most aquatic habitats.

6. There are relatively ample information on the responses of different environmental pressures to the occurrences of different species and benthic macroinvertebrate commmunities.

7. There are a large number of benthic macroinvertebrate species. Thus different benthic macroinvertebrate communities are diverse.

8. Benthic macroinvertebrates generally have limited mobility so they are indicators of localized environmental conditions.

9. Benthic macroinvertebrates are often suitable for experimental studies.

10. Benthic macroinvertebrates have a relatively long life cycle so they can be used in long-term biomonitoring. Even a short-time low dissolved oxygen level or low pH-value in a freshwater ecosystem can be seen in them after a long period of time.

Although several advantages have been listed for the use of benthic macroinvertebrates in biomonitoring, there are also disdvantages. In [66],[84] and [86], a few of the disadvantages are noted:

1. Benthic macroinvertebrates do not respond to all environmental pressures.

2. Quantitative sampling requires a large number of samples to achieve reliable results.

3. Processing of samples and identification of benthic macroinvertebrates are laborious tasks and these processes need special expertise.

4. The occurrence and the abundance of benthic macroinvertebrates are seasonal, especially in the case of aquatic insects.

5. The spectrum of different biological and diversity indices is wide and can be a source of dissatisfaction in experimental results.

6. Some benthic macroinvertebrates may enter the sampling area due to drifting.

7. Calculated biological indices from the data are geographically restricted.

8. The value of diversity indices is heavily dependent on the sampling method.

The best seasons for benthic macroinvertebrate sampling are in the spring or autumn if the sampling is done in inland waters [66]. If sampling is done in maritime areas, summer is also a possible season for sampling [66]. Different seasons have their own advantages for sampling. In spring samples also include larger benthic macroinvertebrates due to overwintering and these specimens can better explain winter phenomena [66]. Moreover, larger benthic macroinvertebrates are easier to identify. If sampling is performed in autumn, benthic macroinvertebrates maturing in summer can be obtained as samples.

Sampling methods are divided into qualitative and quantitative methods [66]. When using quantitative methods, the number of benthic macroinvertebrates is found per known area. In other words, the density of benthic macroinvertebrates can be evaluated [66]. A common way to use quantitative method in practice is to use Surber sampler. Of the qualitative methods one of the most used methods is kick-net method, which is applicable to various surfaces [66]. Usually kick-net sampling goes in practice as follows: A biologist disturbs the substratum by kicking and, thus, all the material together with benthic macroinvertebrates drifts into the net where the benthic macroinvertebrates are separated [66]. This way the kicking area can be fixed and the kicking time can be made constant [66]. Generally speaking, sampling methods are highly standardized and the regulations are precise (see, for example [35]).

Proper preprocessing of samples can make the later stages easier [66]. One problem is to separate the non-important living and non-living material from a sample, but usually the most time-consuming stage is the separation of specimens from samples. This is done manually one by one and needs special attention. The process can be made easier using subsampling [66], but the workload of this stage can still be onerous. When the specimens have been preserved, they are identified manually one by one. The identification process and the specimen separation are the two most laborious and cost-intensive stages in the benthic macroinvertebrates classification. Fully

automated benthic macroinvertebrate identification (see [50, 69]) of all possible species of benthic macroinvertebrates is probably only a dream, but if the major part of specimens could be identified automatically, the costs could be significantly reduced. The remaining specimens not automatically identified could be identified manually afterwards.

There is a wide variety of criteria and indices when benthic macroinvertebrates are used in water quality assessment. Dahl, Johnson and Sandin [12] used 84 single metrics alone in detecting organic pollution of streams. Vuori et al. [84] propose five general criteria for assessment:

1. Taxonomic composition
2. Abundances
3. The absence of important taxonomic groups
4. The proportion of sensitive and non-sensitive taxa
5. Species diversity.

Taxonomic composition refers to species and species groups that exist in a particular bounded environment [84]. Abundance can be described with the number of individuals of a species in a sample ($\frac{ind.}{sample}$) by density ($\frac{ind.}{m^2}$) or by relative proportion of species [84]. The total abundance is not a very sensitive indicator of anthropogenic impacts in less disturbed aquatic ecosystems [84]. However, the influence of severe stress, regulation or high content of toxic substances typically clearly reduce the abundance [84]. Variables that account for both taxa abundances and taxonomic composition (e.g. Percent model affinity) are more sensitive than variables representing only species composition [66].

Another good indicator of anthropogenic pressure is the absence of important taxonomic groups typically occurring occur in natural state lake sites. This may indicate changes in the functionality of an ecosystem [84]. Benthic macroinvertebrate species tolerate different environmental pressure in different ways. The proportion of sensitive and non-sensitive taxa is closely related to community composition and distinct tolerance of species with respect to environmental factors. A common measure to describe the relation between sensitive and non-sensitive taxa is the EPT ratio [66]. This represents the proprotion of EPT taxa (*Ephemeroptera*=mayfly, *Plecoptera*=stonefly, *Trichoptera*=caddisfly) and other taxa. EPT order is often used because the families within these orders are dominant in clean waters [86] and the deterioration of water quality can be seen in them quickly. Generally speaking, benthic macroinvertebrates represent a wide spectrum of pollutant tolerance

(see, for example [86]). Diversity can be measured by using species richness [84]. The most simplest to measure species richness is to use species density in which the number of species is proportioned within a fixed area. Moreover, the abundances between species can be measured with evenness [84].

Benthic macroinvertebrates communities are used routinely in numerous assessments [66]. The responses of benthic macroinvertebrate communities are practicable as a coarse level indicator of environmental changes. Benthic macroinvertebrates can be used as an indicator of the presence of toxic substances in water system stress with three ways [66]:

1. Measuring the residues of toxic compounds from benthic macroinvertebrates.

2. Analysing the health of populations using biomarkers.

3. Testing the level of pollutants in the water or sediment in laboratory and field experiments.

Morphological deformations have proved to be useful biomarkers (see, for instance [85]). The simplest way to ascertain the injurious effects of toxic substances using benthic macroinvertebrates is to investigate the frequency of deformed individuals in the polluted areas compared to some non-polluted reference target [66].

Chapter 3

Machine Learning

3.1 Definition

> We are overwhelmed with data. The amount of data in the world, in our lives, seems to go on and on increasing - and there's no end in sight. As the volume of data increases, inexorably, proportion of it that people understand decreases, alarmingly. Lying hidden in all this data is information, potentially useful information, that is rarely made explicit or taken advantage of. A scientist's job (like a baby's) is to make sense of data, to discover the patterns that govern how the physical world works and encapsulate them in theories that can be used for predicting what will happen in new situations. [87]

Data mining is defined as a process of discovering patterns in data [87]. Machine learning algorithms have a great practical value in many application domains and one of these applications is data mining [54]. Classification is an important subset of data mining and applying machine learning algorithms to these tasks is nowadays active. Machine learning algorithms seem to have great practical value but how can we define machine learning exactly? Mitchell [54] defines it first broadly by saying that machine learning includes any computer program that improves its performance at some task through experience. More specifically, machine learning can defined as follows [54]:

> A computer program is said to learn from experience E with respect to some class of tasks T and performance P, if its performance at tasks in T, as measured by P, improves with experience E.

By adapting the aforementioned definition, the classification of benthic macroinvertebrates could be defined as follows:

- Task T: classifying benthic macroinvertebrate taxa within images.
- Performance measure P: percentage of benthic macroinvertebrates correctly classified.
- Training experience E: a database of benthic macroinvertebrate images with given taxa.

In other words, in classification problems, generally speaking, the aim of using machine learning algorithms is to find a model from the available data that can predict the class labels for the unknown objects. Moreover, the set of class labels is limited. Classification is predicting a label from the predefined set of classes for the unknown objects. Commonly used machine learning methods for classification are decision trees [87], artificial neural networks [26], Bayesian learning [8], SVM [7, 70] and clustering algorithms such as SOM and K-Means [8].

Learning paradigms (machine learning algorithms) can be divided into supervised, unsupervised and reinforcement learning. Unsupervised learning, quite often referred to as clustering, involves a process that automatically reveals a structure in data and needs no supervision [8]. In other words, when considered a classification problem, we do not know the class labels of the examples and the goal is to collect similar cases to their own groups. Typical examples of unsupervised learning are K-Means [8] and SOM [26]. Supervised learning is the opposite of unsupervised learning. In supervised learning we have a collection of data and their characterization as discrete labels [8]. In other words, when supervised learning is used we have data and the class labels of examples are known. Support Vector Machines [7, 70] and neural networks [26] are typical examples of supervised learning. Reinforcement learning is a learning paradigm between unsupervised and supervised learning. It is guided by signals that could be sought as a generalization of the more detailed supervision signals used in supervised learning [8]. Probably the most familiar reinforcement learning algorithm is Q-learning algorithm [54]. However, reinforcement learning is beyond the scope of this thesis.

3.2 Preprocessing

3.2.1 Image Processing

Before any classification method can be applied to a dataset, preprocessing of the data is needed. Preprocessing is a multi-stage process which in the case of benthic macroinvertebrate classification begins when the biologists have

collected the samples from rivers. A thorough preprocessing is a requirement for successful classification. The first actual preprocessing stage is when the benthic macroinvertebrates are separated from the samples. When the specimens have been separated and preserved, taxonomists identify them in the traditional way. After this specimens are scanned onto a computer in single species batches using a flatbed scanner [89].

Now the actual preprocessing of the images can be done. For the datasets of the thesis, segmentation for each batch-scan picture file was performed in four steps [89]. These were:

1. An estimation of the image background was obtained by median filtering an image with a kernel size larger than the radius of the largest object in the image.

2. The background image was used to normalize the background value of each pixel in the original image to the average of the background level.

3. A global threshold was used to generate a mask that separated the specimens from the background in the normalized image. This procedure enables the detection of individual specimens as connected areas in the mask. An automatic technique was used in the choice of threshold value.

4. If the specimens were touching or overlapping, the segmentation was resolved manually.

After segmentation of all greyscale batch-scan picture files, feature extraction from the images was done [89]. Using ImageJ [32] an individual was automatically separated from the background of the image and the features were extracted [89]. This procedure is explained in [89], where a smaller dataset (1350 images) was used. In the smaller dataset case 25 features were extracted using ImageJ and in the larger dataset 32 features were extracted. Table 3.1 shows all the features extracted from the images using ImageJ. In Table 3.1 the first five columns present those features occurring in the smaller dataset, and the larger dataset contained all the features presented in Table 3.1. Precise definitions of all features extracted can be found in [32].

3.2.2 Data Presentation and Preparation

In classification tasks the data is assumed to be in matrix form. A data matrix is an $n \times m$-matrix where the rows represent the examples (instances,samples or cases) and the columns represent features (attributes or variables). Examples are the things to be classified and each one of them are characterized

Table 3.1: Features extracted from the images.

Mean	XM	Kurt	BX	Minor	FeretX	Round
StdDev	YM	Area	BY	Angle	FeretY	Solidity
Mode	IntDen	X	Width	Circ.	FeretAngle	
Min	Median	Y	Height	Feret	MinFeret	
Max	Skew	Perim.	Major	%Area	AR	

by the values of a set of predetermined features [87]. In other words, examples represent entities described by one or more features [8]. If examples are described with more than one feature, the data used in classification can be referred to as a multivariate data [8]. Otherwise, the data is univariate. Features are described by a set of corresponding values and examples which have the same features are grouped to form a dataset [8]. More specifically, features can dichotomized to qualitative and quantitative features. Qualitative and quantitative features have different kinds of values and have their own levels of measurements [61]. There are four levels of measurements and the order is based on how much information they carry [61]. These levels are nominal, ordinal, interval and ratio scale. Nominal and ordinal scales belong to qualitative features and features having interval or ratio scale are quantitative features.

A nominal scale signifies that there is no natural order between values [8]. A typical example from nominal scale feature is gender which usually can be assigned as 0 or 1 [61]. Thus it is also a special binary case of nominal features. Frequencies, percentages and modes are commonly used statistical measures with nominal scale features [61]. Ordinal scale implies that there is some kind of order between values [8]. An example of ordinal scale is a 5-point scale (1=very bad, 2=bad, 3=neutral, 4=good, 5=very good) which is typical in questionnaires. Descriptive statistics that rely on rank ordering (e.g. median) can be used together with percentages, frequencies and modes [61].

Interval scale differs from the two aforementioned such that values are ordered and it is rational to talk about the difference between two values [87]. Moreover, the values on an interval scale are measured in fixed and equal units [87]. Temperature in degrees Celsius is a classic example of interval scale. All the common parametric statistics such as means, standard deviations etc. can be applied to the data with an interval scale [71]. The values in a ratio scale are the most permissive, since they are treated as

real numbers [87]. Hence, any mathematical operations are allowed with the values of a ratio scale [87]. A ratio scale has a fixed zero point and ratio scale features can be multiplied by a positive constant without violating the ratio of the values. Weight is a common example of a ratio scale feature. In the thesis all features are in ratio scale in both datasets. Table 3.2 presents an example of a data matrix of an eight-class classification problem. The first column represents the class label (species) of examples and the other columns represent the features.

Missing values are a commonly encountered problem in real world datasets and need special attention. Many algorithms have been developed for the imputation of missing values. If the number of missing values is small and the dataset is large, an easy solution is to exclude examples having missing values from the dataset. However, this is seldom feasible. Another way to handle missing values is to impute them by using some common statistical value such as median, mode or mean. A hot deck imputation [8] is a possible solution to missing values and in it for each example with missing values the most similar example is found and the missing values are imputed from that example. In this study we have an ideal situation, i.e. both datasets were complete and there were no missing values. Hence, a more detailed overview of this subject is beyond the scope of the thesis.

Table 3.2: Data matrix with a class label (species) and five features.

Species	Area	Mean	Width	Height	Kurtosis
Baetis rhodani	30813	108.062	380	503	-1.072
Diura nanseni	239404	93.967	1302	694	-1.552
Heptagenia sulphurea	83741	97.758	958	460	0.026
Hydropsyche pellucidulla	368784	40.928	1175	740	0.542
Hydropsyche siltalai	122618	52.798	531	57	0.317
Isoperla sp.	36800	70.507	269	565	-0.876
Rhyacophila nubila	159612	55.82	957	466	-0.088
Taeniopteryx nebulosa	47759	49.584	373	657	-0.631

In a dataset features may have significantly different ranges where the values lie. Hence it is often useful to make some transformations to the data so that the features are in balance with each other. Moreover, it usually improves the classification. For instance, in Table 3.2 Area feature has considerably greater values than Mean or Kurtosis. Hence, such disproportion can greatly influence the classification and in this case Area feature can largely control the classification results. Thus, a common practice is to use standardization [87] for features. In standardization statistical mean and

standard deviation are evaluated from feature values and from each value the mean is subtracted and the results divided by the standard deviation. This procedure is known as z-score standardization. By means of standardization the columns of a data matrix have a mean of zero and unit variance. Other transformations can also be made such as linear scaling to interval $[a, b]$ where $a < b$. In this thesis standardization was the only transformation used. Transformations should be made carefully since every transformation draws us farther from the original data and understanding the nature and behaviour of original data becomes more difficult.

3.3 Feature Selection

Dimensionality reduction of an example by means of feature extraction and feature selection is one of the most fundamental steps in data processing [8]. Feature selection is an important phase for several reasons. Firstly, high number of features require more computational power. Secondly, with proper feature selection classification results can be improved because irrelevant features are eliminated.

Numerous algorithms have been developed for feature selection and the research continues. In [89], for instance, the number of features was reduced by canonical discriminant analysis and in [8] several feature selection algorithms were presented. However, if the number of features is small, the best possible feature subset can be searched by trial and error method. When the number of features is high, more sophisticated methods can be used, such as PCA [8]. Moreover, an important approach to feature selection is to use knowledge from other studies of the same application. Human expertise can also be a valuable source of information for feature selection, especially in medicine and biology.

In this thesis Scatter method [34, 72, 73] was used for two purposes. Firstly, in the class division problem (described in more detail in Chapter 4) and in feature selection. Scatter method is an algorithm which evaluates a measure for separability. Scatter algorithm computes results from data on the basis of four alternatives [34]:

1. A common scatter value for the whole dataset
2. Scatter values for all classes by means of a whole dataset to express separation between each class vs. rest.
3. Scatter value for all features using the whole dataset.
4. Scatter values for all features within each class vs. rest.

The main idea in Scatter algorithm is to pick a random example and to go through the dataset always going to the closest unvisited example [34]. In the search for the closest example Euclidean distance is used [34]. Moreover, when traversing the examples through the corresponding class labels are saved in a list [34]. The list indicates how well the classes are grouped in the input space. A complete representation with details of Scatter algorithm can be found in [34], but the core of the algorithm goes as follows:

1. Preprocessing. Normalize all variable values into the same interval of $[0, 1]$. For nominal variables first perform their binarization.

2. Create a class label list:

   2.1. Initialize an empty list A. There are pairs (a_i, l_i) where $a_i \in \mathbb{R}^t$, $i = 1, 2, \ldots, n$ and l_i is the corresponding class label of a_i. Moreover, $a_i \in D$ where D is the dataset.

   2.2. Select a random example $x \in D$ so that every example has an equal probability to be chosen.

   2.3. Search for a closest example y of x with respect to Euclidean distance. If y is not unambiguously determined, choose y randomly from the corresponding examples.

   2.4. Insert the class label of x to the end of list A.

   2.5. Update $D := D \setminus \{x\}$ and consider the case y obtained in Step 2.3.. In other words update $x = y$.

   2.6. If $D \neq \varnothing$, return to 2.3..

3. Computing class changes. Hereafter only list A is considered.

   3.1. Let us change counter to be $v = 0$ and index $i = 1$.

   3.2. Take the classes of l_i and l_{i+1}.

   3.3. If $l_i \neq l_{i+1}$, then $v = v + 1$.

   3.4. $i = i + 1$

   3.5. If $i < n$, return to step 3.2..

4. Computing a scatter value.

   4.1. Compute the theoretical maximum number w of changes:

      4.1.1. Find the sizes of classes $c_1, c_2, \ldots, c_k$, i.e., the numbers of their cases.

4.1.2. From $c_1, c_2, \ldots, c_k$, search for the largest class c_i. Let its size be

$$S(c_i) = n - M = n - \sum_{j \neq i} S(c_j),$$

where M is the size of the counterclass (the other classes together) of c_i. If there are several classes c_i of the equal maximal size, go to 4.1.5.

4.1.3. Decide:

4.1.4. If $S(c_i) > M$, then assign $w = 2M$

4.1.5. else assign $w = n - 1$.

4.2. Evaluate the scatter value $s = v/w$ which is in interval $(0, 1]$.

5. Computing a statistical baseline z: Prepare a simulated class label list applying the class distribution of an original dataset given: generate a random list of class labels according to the frequencies of the classes of the dataset. Repeat steps 2-4 at least 30 times for simulated class label lists to give average scatter values with small standard deviations and z as their mean.

6. Computing separation power F:

$$F = z - s.$$

3.4 Evaluation

In classification evaluation is the final stage. It is an important stage since it reveals the success of classification. In order to gain some perspective on classification, certain measures are needed. Different measures yield different perspectives on the classification. Two commonly used measures are accuracy and classification rate (also known as true positive rate or sensitivity) [8]. Accuracy measures how many samples have been classified correctly from all examples. More specifically, accuracy can defined as follows:

$$ACC = 100 \cdot \frac{\sum_{i=1}^{M} tp_i}{\sum_{i=1}^{M} p_i}\%$$

where M is the number of classes, tp_i is the number of correctly classified samples in ith class and p_i is the size of class i. Classification rate measures how many examples have been classified correctly from a specific class. In other words, the classification rate for the ith class is

$$CR_i = 100 \cdot \frac{tp_i}{p_i}\%.$$

Although accuracy is the most common measure, it has a disadvantage, especially when the class distribution is markedly skewed [41]. If a dataset contains one or two classes significantly larger than the other classes, accuracy can be close to 100% although the smaller classes may be misclassified completely. Hence it is advisable also to report other measures than accuracy. Throughout of this thesis accuracy and classification rate have been used as a measures. ROC curves [8], for instance, have been omitted from the thesis.

How can these measures be used in practice? The evaluation of performance measures such as accuracy and classification rate require some modifications to a dataset of n examples. The basic approach is to divide a dataset into two disjoint subsets called training and test set [8] and from these performance measures can be evaluated. This data division method is also called the holdout method [87]. A classifier uses a training set to construct a model from the data and a test set is used to measure the classifier's generalization ability, i.e. the classifier's ability to predict unseen examples.

There are several suggestions for the proportion of training and test set. It was suggested in [8] that $\frac{2}{3}$ of a dataset should be used for training and the rest for testing. The main point is to use the majority of the data for training. A common approach is to use 90% of the data for training and 10% for testing. Generally, the proportion is heavily dependent on the size of the dataset. If a dataset has a skewed class distribution, a stratified holdout should be used in order to ensure that every class is represented in the training and test tests [87]. Moreover, holdout procedure can be performed several times and in each iteration the training and test sets are chosen randomly. This is called repeated (stratified) holdout and the performance measure can be averaged [87]. The disadvantage in the repeated holdout method is that one cannot be sure how many times an example occurs in the training and test sets. Sometimes a classifier has parameters which need to be tuned up to maximize the results in the test set. Thus a dataset needs to be divided into three subsets: training, validation and test sets [26, 87]. Usually a dataset is first divided into training and test sets and then the training set is again divided into two. In the division of a training set the holdout method can be used again in a similar way as before.

In real world situations data are usually limited and independent training, validation and test sets cannot be indefinitely chosen for classification. Therefore some other ways need to be considered. Nowadays a commonly used technique is cross-validation [8, 26, 87]. The main idea in cross-validation is to divide the dataset into a fixed number of folds [87]. Hence, from the cross-validation name k-fold cross-validation is typically used where $k < n$ and n is the number of examples. In k-fold cross-validation the dataset is divided

into k disjoint subsets of equal sizes. To ensure that every subset contains examples from each class, cross-validation can be done using stratification. Thus the cross-validation is called stratified k-fold cross-validation.

When using k-fold cross-validation, the classification process is performed k times. In every round $k-1$ subsets are used for training and one subset for tests [8]. Hence every subset (and every example) is once a test set and $k-1$ times a training set. To ensure that the cross-validation partition does not effect the classification results, k-fold cross-validation can be repeated m times with different partitionings, which can be referred to as m times k-fold cross-validation. When cross-validation is used in classification, the mean of the performance measures is evaluated. The choice of k in cross-validation is entirely data dependent. If a dataset is small, then $k = 3$ or $k = 5$ may be good choices. However, in practice $k = 10$ is a common alternative and often 10-times 10-fold cross-validation is used in classification. An extreme variant in k-fold cross-validation is to use the leave-one-out method [26], where $n-1$ examples are used for training and only one example for testing. This process is repeated n times and each time a different example is left for testing.

If a classification method needs parameter tuning and cross-validation is applied to a dataset, a slightly different approach can be used. In these situations so-called nested cross-validation can be used, where a dataset is first partitioned with k-fold cross-validation and in each round another k-fold cross-validation is applied to a training set ($k-1$ subsets). However, this solution is very time-consuming. Another way is to take a validation set from every training set and use them in parameter tuning. There are also other methods such as bootstrapping, where the same example can be used a second time whenever examples are taken for a training set [8, 87].

3.5 Statistical Testing

Usually a classification process is repeated with several methods using the same test setup and a comparison between results is made. Statistical analysis is necessary to find the significant differences among the results, especially when the methods perform in a similar manner. A typical situation where statistical analysis is done is the comparison of accuracies (see preceding section) of different methods.

The Wilcoxon signed ranks test [61, 71] is a widely used non-parametric test to analyse paired data. When statistical analysis is performed, the data need to satisfy the assumptions of selected methods. The Wilcoxon signed ranks test assumes that [61]:

1. The data are paired observations from a single randomly selected sample,

constructed either through matched pairs or through utilizing subjects as their own control.

2. The data to be analysed must have at least ordinal scale both within and between pairs of observations.

3. There is symmetry of the difference scores about the true median.

The *null hypothesis* (H_0) and *alternative hypothesis* (H_1) of the Wilcoxon signed ranks test are [71]:

H_0**:** Components of the paired data have same medians.

H_1 : Components of the paired data have unequal medians.

The actual process for using the Wilcoxon signed ranks test in practice is as follows [61, 71]:

1. Evaluate the signed difference of the paired data.

2. Calculate the absolute values of the differences.

3. Rank the absolute values of the differences between the two variables from lowest to highest. If a tie situation occur, assign the mean of the tied ranks.

4. Give each rank a positive or negative sign according to the sign of the original difference.

5. From the analysis drop those pairs which are equal. Determine n, the number of nonzero differences.

6. Determine T, the sum of the positive ranks.

To test the hypothesis, test statistic z is formed [61]:

$$z = \frac{x - \mu}{\sigma} = \frac{T - [n(n+1)/4]}{\sqrt{n(n+1)(2n+1)/24}}.$$

If $z > z_{critical}$, reject H_0. The value $z_{critical}$ can be checked, for example, from the tables. The region of rejection must be adjusted appropriately for a two-tailed hypothesis.

Chapter 4

Support Vector Machine

4.1 Binary Classification Problem

Support Vector Machine is a computer algorithm that learns by examples to assign labels to objects [55]. It was originally developed for two-class (binary) classification problems. SVM [4, 5, 7, 9, 24, 70, 80] has gained wide popularity among practitioners and researchers. The ongoing research around SVM is very active and improvements are constantly being developed. SVM has been applied to numerous applications such as image classification [6, 13], and land cover classification [27]. Moreover, SVM has become popular in a wide variety of biological applications [3, 55]. One example of these biological applications is benthic macroinvertebrate classification [33, 38, 76].

The idea of SVM originated in the case where a dataset of two classes was linearly separable. The principle in SVM is to construct a hyperplane (a linear decision function) which separates the classes in dataset. The hyperplane on which the points $\mathbf{x}$ lie, satisfies the equation

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$$

where $\mathbf{w} \in \mathbb{R}^m$ is a weight vector (normal to the hyperplane) and $b \in \mathbb{R}$ is a bias term (also known as a threshold) [5]. Moreover $\langle \cdot, \cdot \rangle$ is an inner product (dot product in $\mathbb{R}^m$). However, a separating hyperplane is not unique and the problem was to find a separating hyperplane that will generalize well [9]. This problem was solved in 1965 with the realization that a separating hyperplane should have maximum margin in order to have good generalization ability [9]. Maximum margin means that the distance between the separating hyperplane and the closest examples of both classes is maximized. These examples are called support vectors since they define the margin, and support vectors usually form a relatively small subset of all training examples. Hence,

SVM is also known as a maximum margin classifier. Figure 4.1 presents an example of a linearly separable data where optimal margin and hyperplane have been found. The black circles and rectangles lying on the dashed lines (canonical hyperplanes) represent the support vectors.

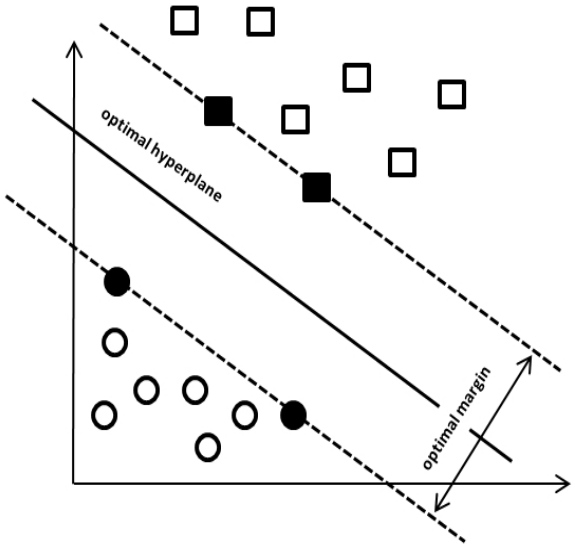


Figure 4.1: Example of an optimal separating hyperplane and optimal margin in a two-dimensional linearly separable data.

How can a separating hyperplane be found exactly if the dataset is linearly separable? Assume that we have a collection of training examples $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^m$, $i = 1, 2 \ldots, n$ are training examples and $y_i \in \{-1, 1\}$, $i = 1, 2, \ldots, n$ are the corresponding class labels of $\mathbf{x}_i$. In a linearly separable case support vectors lie on the canonical hyperplanes $|\langle \mathbf{w}, \mathbf{x}_i \rangle + b| = 1$ for some $i = 1, 2, \ldots, n$ and the distance between the canonical hyperplanes (total margin) is $\frac{2}{\|\mathbf{w}\|}$ [7]. Training examples can be presented with the inequalities

$$y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1, \ i = 1, 2, \ldots, n.$$

Maximizing the margin is equivalent to minimizing the norm $\|\mathbf{w}\|$ [4]. Hence, finding a separating hyperplane with maximum margin returns to solving the following problem:

$$\min_{\mathbf{w}, b} \tfrac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1, \ i = 1, 2, \ldots, n. \tag{4.1}$$

The optimization problem in (4.1) is a convex Quadratic Programming (QP) problem, since the objective function is convex and the points satisfying the constraints form a convex set [5]. To derive the solution for $\mathbf{w}$, a Lagrangian formulation of the problem needs to be introduced [5]. There are two kinds of Lagrangian formulations: primal and dual. The primal Lagrangian formulation is:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \tfrac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1] \tag{4.2}$$

where *Lagrange multipliers* $\alpha_i \geq 0$, $i = 1, 2, \ldots, n$. Primal Lagrangian L is maximized with respect to α_i's and minimized with respect to $\mathbf{w}$ and b. By solving the saddle point, the derivatives of the primal Lagrangian formulation with respect to $\mathbf{w}$ and b must vanish [70] and, hence,

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

and

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

are obtained [70]. According to the Karush-Kühn-Tucker (KKT) conditions [5, 70] Lagrange multipliers, which are non-zero at the saddle point, satisfy the constraints in (4.1) and training examples with $\alpha_i > 0$ are support vectors. Since the optimization problem is a convex QP problem, a dual formulation can likewise be solved. The dual formulation [70] of the optimization problem is

$$\max W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{4.3}$$

subject to

$$\alpha_i \geq 0, \quad i = 1, 2, \ldots, n$$

and

$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

The optimal hyperplane can be found by solving either the primal or the dual Lagrangian formulation. The solution for the weight vector $\mathbf{w}$ is

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i.$$

Now the hyperplane has the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b.$$

A formula for the bias term can be evaluated from the KKT conditions

$$\alpha_i [y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1] = 0, i = 1, 2, \ldots, n$$

by choosing any i for which $\alpha_i \neq 0$ and evaluating b [5]. Thus,

$$b = y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle.$$

Another way to compute b is to take the mean of all values b obtained from the condition $\alpha_i \neq 0$ [5]. The output value of $f(\mathbf{x})$ is a real number. However, the output space is the set $\{-1, 1\}$, so a class label for a new example $\mathbf{x}$ is obtained by evaluating

$$f(\mathbf{x}) = sgn\left(\sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right).$$

Unfortunately linearly separable data is seldom encountered in practice. When the data is linearly non-separable, we need to introduce a modified version from the original optimization problem. Cortes and Vapnik [9] introduced the soft margin SVM and in this modified version the concept of *slack variables* $\xi_i \geq 0$ is needed. The slack variable measures the deviation of an example from the ideal condition of example separability [26]. Now training examples can be presented with inequalities

$$y_i [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1 - \xi_i, i = 1, 2, \ldots, n.$$

The optimization problem can now be presented as follows:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}} \tfrac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \tag{4.4}$$

subject to $y_i [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1 - \xi_i$ and $\xi_i \geq 0$, $i = 1, 2, \ldots, n$. Moreover, the user-defined parameter C (also known as the box constraint) is important in classification. It is a trade-off parameter between maximum margin and minimum classification error. The calculation of an optimal hyperplane goes analogously (see for example [5]) as in the linearly separable case. The dual formulation is now the same as in linearly separable case and the slack variables do not occur in it, but the difference from the linearly separable case is that in the former Lagrange multipliers are bounded above with a constant C.

Kernel function [5, 7, 9, 70] is a key concept in SVM. The use of kernel functions allows us to move from linear support vector classifiers to nonlinear support vector classifiers. The main idea in the use of kernel functions is to map the training examples from the input space to a higher dimensional feature space where the examples are linearly separable. Hence the separating hyperplane can be constructed in a feature space and new examples can be classified there. However, explicit mapping into the feature space can be very difficult and the feature space could be of infinite dimensions. Fortunately, the actual mapping of training examples to the feature space does not need to be done. It has been shown that the inner products between the training examples in the input space can first be evaluated and then a nonlinear transformation of the value of the result can be made [9]. An example of a linearly non-separable data in a two-dimensional space is presented in Figure 4.2, which is mapped with a nonlinear transformation ϕ to another two dimensional feature space where the data is linearly separable. The dashed lines in Figure 4.2 represent the margin and the black rectangles and circles are support vectors. Moreover, a solid line represents the decision boundary. More specifically, kernel functions can be defined as follows [7],

Definition 1. *A kernel is a function K, such that for all $\boldsymbol{x}, \boldsymbol{z} \in X$*

$$K(\boldsymbol{x}, \boldsymbol{z}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle,$$

where ϕ is a mapping from the input space X to an (inner product) feature space F.

An important note is that when kernel functions are used, it is necessary to ensure that a kernel function is valid. If a kernel function satifies the conditions of Mercer's theorem [7, 70], it is a valid kernel function.

Theorem 1. *Mercer's theorem. Let X be a compact subset of $\mathbb{R}^m$. Suppose K is a continuous symmetric function such that the integral operator $T_K : L_2(X) \to L_2(X)$,*

$$(T_K f)(\cdot) = \int_X K(\cdot, \boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x},$$

is positive, that is

$$\int_{X \times X} K(\boldsymbol{x}, \boldsymbol{z}) f(\boldsymbol{x}) f(\boldsymbol{z}) d\boldsymbol{x} d\boldsymbol{z} \geq 0,$$

for all $f \in L_2(X)$. Then we can expand $K(\boldsymbol{x}, \boldsymbol{z})$ in a uniformly convergent series (on $X \times X$) in terms of T_K's eigen-functions $\phi_j \in L_2(X)$, normalized in such a way that $\|\phi_j\|_{L_2} = 1$, and positive associated eigenvalues $\lambda_j \geq 0$,

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\boldsymbol{x}) \phi_j(\boldsymbol{z}).$$

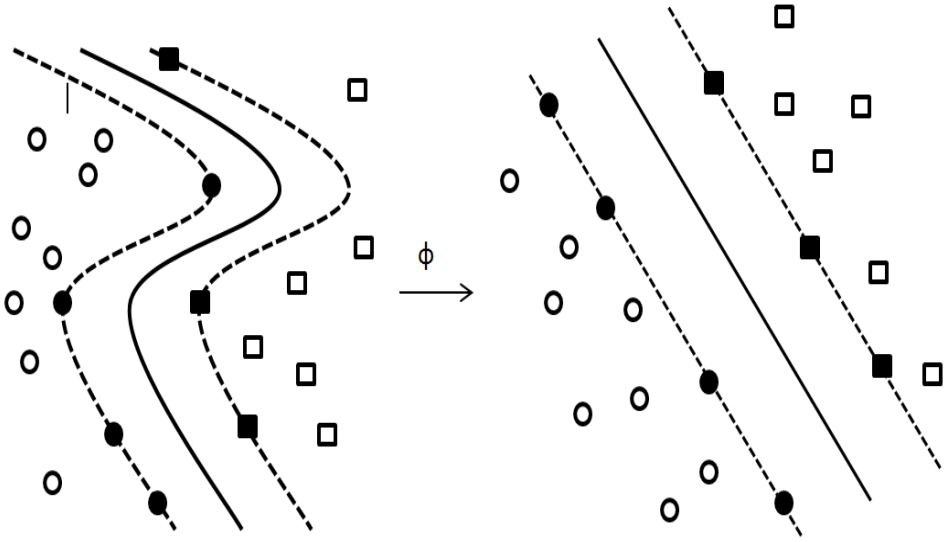


Figure 4.2: An illustration of the use of kernel function in SVM.

There are some commonly used kernel functions in the literature which are also applied to the tests in this thesis. These are:

1. Linear: $\langle \mathbf{x}, \mathbf{z} \rangle$,
2. Polynomial: $(\langle \mathbf{x}, \mathbf{z} \rangle + 1)^d$ where $d \in \mathbb{N}$ is the order of the polynomial kernel function,
3. Radial Basis Function: $e^{-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma^2}}$ where $\sigma > 0$,
4. Sigmoid: $\tanh(\kappa\langle \mathbf{x}, \mathbf{z} \rangle + \delta)$ where $\kappa > 0$ and $\delta < 0$.

When using kernel functions, the dual formulation of the optimization problem is written in the form:

$$\max W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$$

subject to

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \quad \text{and} \quad 0 \le \alpha_i \le C,\ i = 1, 2, \ldots, n.$$

Moreover, the use of kernel functions leads to a decision function in the form

$$f(\mathbf{x}) = sgn\left(\sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b \right)$$

and bias b can be obtained, for instance, by taking the mean

$$b = y_j - \sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}_j, \mathbf{x}_i)$$

over all points with $\alpha_j > 0$ [70].

Training a support vector machine requires solving a very large QP optimization task [64]. Platt [64] introduced a new algorithm in 1998 for training SVMs called Sequential Minimal Optimization (SMO). SMO divides the large quadratic programming optimization problem into a series of the smallest possible QP problems and solves each of them analytically, which speeds up the training process [64]. SMO is a widely used algorithm, but it is beyond the scope of this thesis. Another way in SVM training is to use the Least Squares Support Vector Machines (LS-SVM) [74, 75] which has been used in this thesis. LS-SVM differs from the traditional approach in that now the primal classification problem takes the form

$$\min_{\mathbf{w},b,\mathbf{e}} \tfrac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\sum_{i=1}^{n} e_i^2$$

subject to equality constraints

$$y_i[\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b] = 1 - e_i, i = 1, 2, \ldots, n.$$

The error terms $e_i, i = 1, 2, \ldots, n$ play a role similar to that of slack variables in QP formulation and γ is a user-defined parameter as is a box constraint in QP presentation. The Lagrangian formulation has now the form:

$$L(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha}) = \tfrac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\sum_{i=1}^{n} e_i^2 - \sum_{i=1}^{n} \alpha_i (y_i[\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b] - 1 + e_i)$$

where α_i's can now take positive and negative values [75]. The optimality conditions are [74]:

$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{w}} = 0 \quad &\rightarrow \quad \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x}_i) \\
\frac{\partial L}{\partial b} = 0 \quad &\rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \\
\frac{\partial L}{\partial e_i} = 0 \quad &\rightarrow \quad \alpha_i = \gamma e_i, \quad i = 1, 2, \ldots, n \\
\frac{\partial L}{\partial \alpha_i} = 0 \quad &\rightarrow \quad y_i[\mathbf{w}^T \phi(\mathbf{x}_i) + b] - 1 + e_i = 0, \quad i = 1, 2, \ldots, n.
\end{aligned}$$

Eliminating $\mathbf{w}$ and b yields

$$\left[\begin{array}{c|c} 0 & \mathbf{y}^T \\ \hline \mathbf{y} & \Omega + \gamma^{-1} I \end{array}\right] \left[\begin{array}{c} b \\ \hline \boldsymbol{\alpha} \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline \mathbf{1} \end{array}\right]$$

where $\mathbf{y}^T = [y_1, \ldots, y_n]$, $\mathbf{1}^T = [1, \ldots, 1]$, and I is an $n \times n$ identity matrix [75]. Kernel trick can be applied to the matrix Ω. Thus,

$$\begin{aligned} \Omega_{ij} &= y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \\ &= y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned}$$

Hence, instead of using quadratic programming classifier

$$f(\mathbf{x}) = sgn\left[\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b \right]$$

is obtained by solving a linear set of equations (for more information see [74, 75]).

4.2 Multi-Class Extensions

4.2.1 One-vs-All

Since SVM was originally developed for two-class classification problems, a natural interest awoke to extend SVM to also concern multi-class problems. One-vs-all (also known as one-vs-rest, OVA) [13, 19, 29, 49, 65, 70] is one of the most commonly used multi-class extensions. The basic idea is very simple. In an M class ($M > 2$) classification problem M individual binary SVM classifiers are constructed. Each one of M classifiers is trained to separate one class from the others. OVA method has its pros and cons. The advantage is in the small number of classifiers, but the disadvantage lies in the training phase, which is computationally heavy because every classifier needs to handle full training data. When a new example is to be classified, all M classifiers are run and a classifier giving the positive output for the example assigns the class label for the new example.

However, occasionally situations are encountered where a new example obtains a positive output from more than one classifier, or every classifier gives a negative output. In other words, a tie situation has occurred. A common way to solve these tie situations is to apply a winner-takes-all method [70], where the real outputs of SVM classifiers are compared. More specifically, the winner-takes-all method selects the class having the largest (most positive) output from the tied classes. In other words

$$class\ of\ \mathbf{x} = \arg\max_{i \in T} (\langle \mathbf{w}^i, \phi(\mathbf{x}) \rangle + b^i). \tag{4.5}$$

where T is the set of indices of tied classes. In Equation (4.5) $\mathbf{w}^i$ is the weight vector of the ith classifier and $\phi(\mathbf{x})$ is the mapped new example and b^i is the bias from the ith classifier. If all outputs are negative, the smallest output defines the class label. Hong [29] introduced another way to solve ties based on probabilistic ordering by using Naïve Bayes classifiers and in this thesis a new tie solving strategy based on k-NN is introduced.

4.2.2 One-vs-One

Another typically used multi-class extension of SVM is the one-vs-one method (also known as pairwise classification, OVO) [15, 19, 49, 70]. OVO differs from OVA in several ways. In OVO a classifier for each possible pair of classes is trained [70]. Hence the total number of classifiers is $\frac{M(M-1)}{2}$ in an M class ($M > 2$) classification problem. Whereas in OVA every classifier handles the full training data, in OVO every classifier is trained only with the training examples of ith and jth classes ($i < j$). Although the number of classifiers is decidedly greater than in OVA, the training time needed for an individual classifier is significantly shorter. Therefore it is possible to save time when using OVO method instead of OVA [70].

The information from the individual classifiers needs to be somehow collected together. A commonly used method is the majority voting method (also known as the max-wins rule), where every binary classifier gives a vote for the predicted class [19]. The class having the most votes assigns the class label to a new example. This approach, however, has a drawback. Ties arise from time to time and some tie situation solving strategy needs to be applied. In the thesis tie situations were solved by applying the k-NN method. Several other methods have also been developed for combining the information from individual classifiers in the OVO method (see for example [19]).

4.2.3 Directed Acyclic Graph Support Vector Machine

Platt et al. [63] introduced a learning architecture called Decision Directed Acyclic Graph (DDAG) to combine multiple binary classifiers into a multi-class classifier [63]. Decision DAG is a graph where there are no cycles and the vertices have a direction. Moreover, DDAG consists of $\frac{M(M-1)}{2}$ nodes and each of the nodes corresponds to a binary classifier [49] and in DAGSVM the binary classifier in a node is an SVM classifier.

The training phase of DAGSVM is the same as in OVO, since the number of classifiers is equal [13]. However, the testing phase differs from the previously introduced methods. The classification of a new example begins from the root node and according to the result of a classifier, via left or right edge

will be moved to the next node until a leaf is reached where there is the final class label for the new example. The path from the root node to the leaf is called an evaluation path [63]. Altogether $M-1$ comparisons are needed to solve the predicted class label for a new example.

DDAG is equivalent to operating in a list, where each node eliminates one class from the list [63]. A decision node is formed by taking the first and the last elements on a list. When a test example is evaluated in a decision node, a class which was not chosen is eliminated from the list and DDAG continues to again test the first and the last elements on the list. The advantage of DAGSVM lies in the fast testing phase. It was also developed for the purpose to solve the unclassifiable regions which occur in OVA and OVO methods. However, DAGSVM also has a disadvantage. The order of the list (or DDAG) can be arbitrary and every order can produce different classification results. Moreover, the number of possible list (or DDAG) orders is $M!$ in an M class problem and even with small values of M such as 8 or 10 the number of possible orders is very large. In practice it is impossible or a computationally very demanding task to test every possible order. Figure 4.3 presents an example of a four-class DAGSVM architecture. To conclude, exhaustive search here is reasonable only for small values of M.

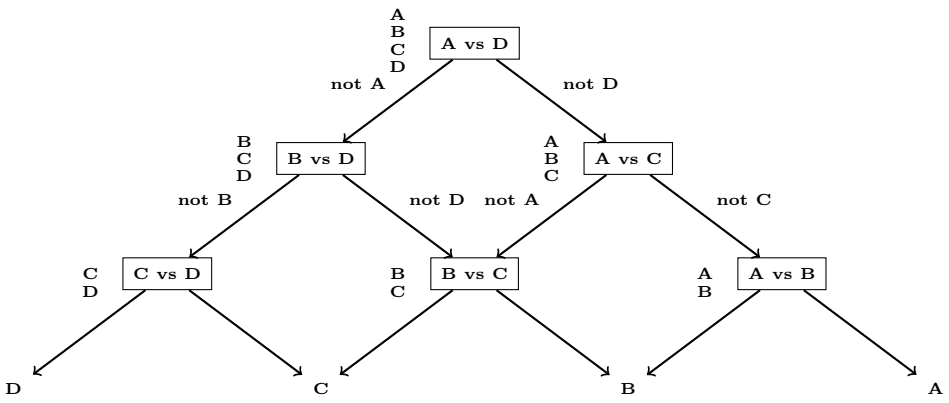


Figure 4.3: An example of a four-class DDAG learning architecture.

4.2.4 Half-Against-Half Support Vector Machines

In [46] an interesting multi-class extension of SVM called Half-Against-Half Support Vector Machines was introduced. It uses a binary tree structure in each node of which there is an SVM classifier. HAH SVM has altogether $M-1$ nodes, which is less than in OVA, OVO and DAGSVM methods. Moreover,

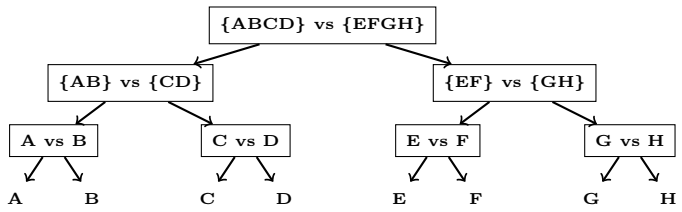


Figure 4.4: An example of an HAH SVM in an eight class classification problem.

the training phase is similar to OVO method [46]. The key point in HAH SVM is to divide classes in an optimal way into two groups. In the root node all classes are handled and they are divided into two groups. In other nodes the root node groups are recursively divided into smaller groups until the predicted class label is found in a leaf. An eight class example of an HAH SVM is shown in Figure 4.4.

The biggest challenge in HAH SVM is to find optimal class divisions in the nodes. Lei and Govindaraju [46] used hierarchical clustering for class division. This approch was used as a suboptimal solution and optimal division was left as an open problem. Moreover, an optimal recursive division of classes into two groups was left as an open research question. In the thesis the division problem was solved by using Scatter method [34, 72, 73], which is a novel approach to this theoretical problem. Scatter method was also used in every node.

Chapter 5

Results

5.1 Publication I - One-vs-One & Tie Situations

The purpose of this publication was to examine how OVO [18, 19] method works in the benthic macroinvertebrate classification and, secondly, to investigate the effect of kernel function choice on the number of tie situations in the majority voting method. The motivation for using OVO was that in [38, 76] it was applied successfully. The motivation for examining tie situations more closely was that they occur in practice relatively often and are typically left with little attention although they may have a great effect on classification results. Tie situations are typically solved with some low level way, such as choosing the smaller index of tied classes or automatically choosing the larger class. However, in this publication a new tie situation solving strategy was presented and was based on k-NN classifier. Furthermore, statistical information about the frequency of tie situations was given in the case of every kernel function used. The final class label for the problematic test example was solved such that k-NN classifier was trained with the training data of tied classes and with the configuration $k = 1$ a final class label was assigned to the test example. The k value was chosen to be 1 because it was the only value which certainly gives a unique solution. With other k values there is a theoretical possibility of encountering another tie situation when multi-class classification is in question.

In the classification seven kernel functions, i.e. linear, polynomial kernel functions (degrees 2-5), Radial Basis Function and Sigmoid kernel function were used. The number of parameter combinations tested was 20 or 400 depending on the kernel function. In Sigmoid case we decided on $\kappa = -\delta$ due for computational reasons. In the classification procedure the dataset

was divided 250 times such that 90% of the dataset (1,215 examples) was left to the training set and the rest of the dataset (135 examples) was left for testing. This procedure was repeated for every parameter combination. More information concerning the species of the smaller dataset and the sizes of the taxonomical groups can be found in the Appendices.

Standardization of training data was applied separately to each binary classifier, but no other transformations such as linear scalings for the dataset were made. The final parameter values were chosen with the method presented in Publication II. The difference from Publication II was that we fixed the box constraint to be the same despite the kernel function selection. Moreover LS method [74, 75] was used to find an optimal hyperplane.

Classification was repeated with two feature sets. These were 15D and 24D feature sets. The 15D feature set was the union of the statistical and geometrical features given in [38, 89]. The dataset contained altogether 25 features and one feature, Area Fraction (%Area), was excluded. The idea of using 15D features came from [38]. The 24D feature set has not been used in any other publication so there is no baseline for the 24D results.

Regarding the results with 15D features, classification errors with linear and RBF kernel functions were less than 6%, which is similar to the SVM results in [38]. However, the quadratic kernel function achieved less than 4% classification error which is slightly better than the best SVM result in [38], but it was at the same level as the best classification error obtained by MLP in [38]. The 24D feature set results were slightly better than the 15D results. The linear, quadratic and RBF kernel functions achieved less than 3% classification error, which was an improvement on the earlier results. The other kernel functions yielded clearly poorer results.

Tie situation analysis followed a pattern similar to the accuracies with the kernel functions. If the accuracy was low, the number of ties was high and, conversely, if the accuracy was high, the number of ties was low. In the statistical information on tie situations means and standard deviations of the number of tie situations were given. With the linear, quadratic and RBF kernel functions the average number of ties was below 2 (less than 1.5%) regardless of the feature set used. Sigmoid kernel function, instead, had the highest mean of ties, being over 43, which was over 30% of the test examples. Kernel selection had a major impact on the number of ties. Moreover, OVO method proved a good alternative for benthic macroinvertebrate classification.

5.2 Publication II - One-vs-One, One-vs-All & Tie Situations

Publication II was a natural continuation of the first publication, since OVA and OVO methods are the most commonly used multi-class extensions. The aim in this publication was to compare OVA and OVO methods against each others. This was an interesting comparison because in [37, 38, 39, 76, 89] it was not used for benthic macroinvertebrate classification. Moreover, in OVA ties are a common problem and therefore tie situations were considered again in this publication. The k-NN method with $k = 1$ was used for solving the tie situations as in Publication I.

Classification was performed by dividing the dataset 250 times in every parameter combination case. The division of the dataset was made such that 90% was left to the training set (1,215 examples) and 10% (135 examples) to the test set. The mean of the results was evaluated as in Publication I. Final parameter values were chosen as in Publication I, but now the method was explained in detail. The main idea was that the mean classification rates (sensitivities, true positive rates) obtained from every parameter combination together with their indices were arranged classwise in a decreasing order in a new table. Now the most frequent parameter index in the first row determined the final parameter values. This method may not yield the best classification result, but it avoids the problems that accuracy may cause when the class distribution is skewed.

In Publication II the same seven kernel functions were used as in Publication I. Compared to Publication I the number of parameter combinations tested was greater. Linear and polynomial kernel functions were tested with 40 parameter values and RBF and Sigmoid (called MLP kernel function in the publication) were tested with 1600 parameter combinations. In Sigmoid kernel function again $\kappa = -\delta$ was set for computational reasons. Otherwise, in Sigmoid the number of parameter combinations tested would have increased to $40^3 = 64,000$. Features were standardized to have zero mean and unit variance and this was done separately for each binary classifier, as in Publication I. No other transformations such as linear scalings to intervals $[-1, 1]$ or $[0, 1]$ were made in order to keep the classification procedure as natural as possible. The baseline for feature selection was a 15D feature set. However, the 15D feature set itself was not used since it had already been tested in Publication I. In this publication the statistical feature set (7D) and geometrical feature set (8D) and a random eight-feature subset (R8D) from the 15D feature set were used. In [37, 38, 39, 89] the division into statistical and geometrical feature sets was made but no random feature subset was used in

other related articles.

Publication I showed the importance of the kernel function selection when considering tie situations and the same tendency also continued in Publication II. When OVO method was applied, RBF kernel function had the lowest number of ties (mean frequency below 2.0) despite the choice of feature set. Moreover, the linear, quadratic and cubic kernel functions had a mean less than 7.0 in the number of tie situations. By contrast Sigmoid had a mean over 36 (over 26% of test examples) in ties notwithstanding the feature set. With a higher degree of polynomial kernel functions there was a greater difference between the number of ties when statistical and other features were compared. When the statistical feature set was used together with the 4th degree polynomial kernel function, the mean of tie situations was below 3 when other feature sets had a mean around 12. Furthermore, in the case of the 5th degree polynomial kernel function the mean of tie situations with the statistical feature was 8 when in other feature set choices the mean was over 20.

The tie situation analysis with OVA method showed a totally different kind of mean numbers, but some similarities with the OVO results were apparent. Firstly, in all feature set selections RBF again gained the lowest mean in ties (range 12%-28% of test examples) and, moreover, Sigmoid had the highest means (over 90% of test examples) despite the feature set selection. However, in OVA the higher degree of polynomial kernel functions gained smaller means than the linear or quadratic kernel functions. Among the polynomial kernel functions (the linear kernel function included) the means varied widely. With the geometrical feature set the means ranged from 81 to around 111, while with the statistical feature set the corresponding range was from around 23 to almost 110. Finally, with the random feature set range was from around 35 to almost 95. Overall, with OVA method tie situations were more frequent than with OVO and the choice of kernel function had a major influence on the number of ties.

Tie situations can have a significant effect on classification results if these situations are incorrectly solved. Furthermore, when OVA and OVO methods were compared against each other, with OVO ties seemed to be more infrequent than with OVA. This result is quite natural, because in OVA we have only M classifiers (in M class problem) and each one of them is trained to separate one class from the rest. Furthermore, in OVA we use all the available training data in every classifier and, hence, the training data is likely to be more difficult to separate. When using OVO method, each binary classifier handles only the training data of ith and jth class instead of all the available training data. Thus the training data of an individual binary classifier in OVO may be more readily separable than in OVA.

When comparing the accuracies of OVA and OVO methods, we made some interesting discoveries. The accuracies gained with the geometrical feature set together with OVA and OVO methods were quite similar except with the Sigmoid kernel function, which achieved around 11% better accuracy with OVA than OVO. Otherwise, RBF kernel function was the best alternative with accuracies around 77% (OVA) and 78% (OVO).

The use of the random feature set increased the level of accuracies with RBF, linear and polynomial kernel functions. In OVA the cubic and 4th degree polynomial kernel function and RBF kernel function obtained the highest accuracies of 87.4%-89.5%. The quadratic, cubic and RBF kernel functions were instead the best choices with OVO method and the random feature set. These accuracies ranged from 88.9% to 90.4%.

The statistical features proved to be the best feature set among the feature sets tested. In OVA, out of seven possible kernel functions only one, RBF, achieved over 90% accuracy, being 91.1%. However, the 4th and 5th degree polynomial kernel functions also reached accuracies of nearly 90%. In OVO method only two kernel functions, Sigmoid and the 5th degree polynomial kernel function, were left below 90% accuracies and from these two the polynomial kernel function reached around 89% accuracy. The quadratic, cubic and RBF kernel functions achieved 93.7% accuracy. In Publication I the linear, quadratic and RBF kernel functions were the best ones, and, compared to the results of Publication II, the use of the quadratic and RBF kernel functions with OVO method got more support. Moreover, the use of RBF kernel function also gained confirmation from OVA method.

5.3 Publication III - Half-Against-Half Support Vector Machines

Publication III examined the suitability of HAH SVM for benthic macroinvertebrate classification. In this publication as in Publications I and II the smaller dataset was used. The publication had two novelties. Firstly, HAH SVM was used in this application for the first time. Secondly, a Scatter method [34] was applied to a class division problem, which is the most challenging problem in this multi-class extension. Lei and Govindaraju [46] left the determination of an optimal division unresolved problem and were satisfied with the suboptimal solution given by hierarchical clustering. Furthermore, in [46] optimal recursive division of the classes was left as an open question.

In Publication III the structure of a directed binary tree was created

beforehand from the full dataset. Scatter method was applied to each of the nodes separately and class divisions were decided according to the concept of the separation power given by Scatter method for the classes. That half of the classes which had the highest separation powers formed their own half and the rest formed the other half of the classes in a node. This procedure was repeated in every node until there was only one class in the leaf. As a baseline for the Scatter method another binary tree construction was made by making a random choice of class divisions in every node.

The dataset was standardized to have zero mean and unit variance in the preprocessing stage and no other transformation was made for the data. In Publications I and II the term accuracy was used as a synonym for classification rate (sensitivity,true positive rate), but now accuracy is referred to as defined in Section 3.4. For the classification, the dataset was divided 100 times into training, validation and test sets such that 80% (1,080 examples) of the dataset was left for training, 10% (135 examples) to the validation set and the rest of the data was left for the test set (135 examples). In Publication III four feature sets were used. These were 7D (familiar from Publication II), 15D (used in Publication I), a randomly chosen 17D feature set and finally 25D, which contained all the available features. The same seven kernel functions were used as in Publications I and II. The number of parameter combinations tested was significantly greater than in Publications I and II. Polynomial kernel functions, including the linear kernel function, were tested with 100 parameter values. The RBF and Sigmoid kernel functions were tested with 10,000 parameter combinations and in Sigmoid case the same $\kappa = -\delta$ was agreed on as in the other publications. The accuracies obtained by Scatter method and random choice were also examined using two-tailed Wilcoxon signed ranks test with $p < 0.05$.

The use of 7D feature set produced good results in Publication II. The results with HAH SVM and 7D feature set were again relatively good. From the 3rd to 5th degrees the polynomial kernel functions obtained above 90% accuracies with Scatter and random choices. The differences between these results were less than 1%. The best results were achieved with RBF. With Scatter method 93% accuracy was obtained and with random choice 93.2% accuracy. In the 15D feature set case only with three kernel functions (quadratic, cubic and RBF) accuracies were above 90% achieved in both division alternatives. Of these three kernel functions RBF was the best one and the corresponding accuracies were 95.2% with Scatter method and 95.9% with random division. These two accuracies were also the highest among all results in Publication III.

The 17D feature set, which was not used in the preceding publications, was the poorest alternative. With the cubic kernel function the best accu-

racies were obtained in both divisions. Using Scatter method yielded 90.3% accuracy and random division 91.2% accuracy. When RBF was the best alternative in 7D and 15D cases, it reached the 90% limit and gained the second place with both methods. The last feature set was 25D, where all the available features were used. The best accuracy was now achieved with the quadratic kernel function with both division methods. The results were similar to the 7D results. With Scatter method 93% accuracy was achieved and 93.4% accuracy with random choice. The RBF kernel function again showed its power reaching above 90% accuracy with both methods and being in second place.

An interesting detail was that regardless of the division method the best accuracies within a specific feature set was achieved by the same kernel function. For instance, in the 15D feature set case RBF achieved the highest accuracy with Scatter and random methods. When the best accuracies of Scatter and random methods were compared against each other, random division yielded better accuracy in all cases and in three feature set cases the difference between accuracies was statistically significant (100 cases). A noteworthy detail was that the difference between the best accuracies of Scatter and random methods was less than 1% in all feature set cases. Hence, from the practical point of view, both division methods were equally good.

5.4 Publication IV - A Comparison of Classification Methods

The purpose of this publication was to investigate the suitability of several classification methods in classification of benthic macroinvertebrates. This publication gave a good baseline for the SVM results in Publications I-III since the smaller dataset was also used in this publication. Altogether 11 classification methods were tested. These included: k-Nearest Neighbour [59, 60, 62, 79] , Linear Discriminant Analysis [48, 56, 60, 77], Quadratic Discriminant Analysis [8, 89] Minimum Mahalanobis Distance Classifier (MMDC) [88], Classification Tree [16, 48], Multinomial Logistic Regression (MNLR) [1], Naïve Bayes [3, 31, 47], K-Means [8, 25], Self-Organizing Map [26, 67, 68], Multi-Layer Perceptron [10, 26, 82] and Radial Basis Function network [26]. Of these methods, MMDC and MNLR, for instance, were used for first time for benthic macroinvertebrate classification.

Classification was performed with the 15D feature set also used in Publications I and III. Accuracy was chosen as the main performance measure in the comparison of the classification methods. In the preprocessing stage

the features of the dataset were standardized to have zero mean and unit variance. No other transformations were made for the dataset. Stratified 10 times 10-fold cross-validation was applied to the dataset. Hence 100 training and test sets were obtained. In the case of Radial Basis Function network and Multi-Layer Perceptrons training sets were divided into a smaller training set and validation set such that 80% of the dataset was the training set, 10% was the validation set and the last 10% was left as the test set. Radial Basis Function network was tested with 40 different values of σ (the width of Gaussian basis function) and MLP was tested altogether with 240 different configurations. When the best configurations for the RBFN and MLP were found by comparing the mean accuracies of the validation sets, RBFN and MLP were trained again with full training data and evaluated with the test sets.

For K-Means and SOM class tags of each cluster (or neurons in SOM) had to be defined for the classification task. In these methods the majority principle was applied. If a tie occurred in class tag determination with K-Means, the closest sample (from the tied classes) with respect to the centroid of a cluster determined the final class tag for the cluster. K-Means was tested with 93 (8-100) different K values. Moreover, SOM was tested with 43 configurations (the number of neurons in a topology altered from 8 to 50). In the case of k-NN odd values from 1 to 51 were used, since the choice of odd values decreases the probability of a tie situation in classification. Moreover, classification with k-NN was repeated with four measures: Euclidean, cityblock, cosine and correlation measures.

The classification methods produced several interesting results. The k-NN method worked relatively well with all measures. The best mean accuracies were obtained, despite the k value, with Euclidean and cityblock measures. The cosine measure was the third best and the lowest accuracies were achieved with the correlation measure regardless of the k value. A general trend with k-NN was that the increment of k usually diminished the accuracy. This happened with all measures. With Euclidean and cityblock metrics the highest mean accuracies were obtained with $k = 1$ or $k = 3$ and the mean accuracy was above 92%. The fact that $k = 1$ gave such a good results strengthened the value of the tie situation strategy presented in Publications I and II.

Classification trees (CT) are a widely used method and in this publication CART algorithm [16] was applied to the classification. With CT around 85% accuracy was achieved and the result was similar to C4.5 performance in [89], where the mean error rate was 0.1731. Naïve Bayes classifier produced poor results. It reached only a 77.8% mean accuracy, which was the lowest one when taking all classification methods into account except the correlation

measure in k-NN. MNLR gave 90.8% accuracy, which is a relatively good result. MMDC gained a slightly higher accuracy being 92.6%, so it proved to be a good choice for benthic macroinvertebrate classification. Linear (LDA) and Quadratic Discriminant Analysis (QDA) are widely used methods and in this publication LDA reached 90.1% accuracy and QDA 93.7% accuracy. In [89] QDA achieved a mean test error of 0.0736 so the levels in the results were similar to each other.

Unsupervised methods did not work very well in the classification. K-Means gave 84.4% mean accuracy, which was 1% lower than CT. The result was achieved using 100 clusters. SOM gave accuracy quite similar to those of K-Means. More specifically, it reached 82.6%, this accuracy being the next lowest result. The accuracy was achieved using 50 neurons in a lattice.

The last aggregate was to use artificial neural networks. RBFN proved a good choice for this classification task since it achieved a mean accuracy of 93.7%. It was the next highest mean accuracy of all classification methods used and it achieved the same accuracy as that of QDA. The best accuracy was obtained when $\sigma = 3.0$. The other artificial neural network method was MLP. The best accuracy, 94.1%, was achieved with configuration 15×15×7×8 and this accuracy was the highest of all methods used in Publication III.

5.5 Publication V - Directed Acyclic Graph Support Vector Machines vs Directed Acyclic Graph k-Nearest Neighbour

The aim of Publication V was to examine the appropriateness of DAGSVM and DAGKNN for benthic macroinvertebrate classification. The first difference from Publications I-IV was that in Publication V a larger 50 species dataset was used. DAGSVM [63] was used in [33] for benthic macroinvertebrate classification with great success and this encouraged us to apply this multi-class extension to a larger dataset. DAGKNN classification method is introduced in Publication V. The basic idea in DAGKNN is the same as in DAGSVM, but now each node contains a k-NN classifier instead of an SVM binary classifier. Species names and the sizes of the species subsets can be found in the Appendices. Another novelty in Publication V was that feature selection was made using Scatter method [34, 72, 73].

Before making the actual classifications, the 50 species dataset was divided into groups of 10 species. The division was made according to the numbers of examples of the species. The first group was formed from the ten largest species etc. and the fifth group contained the ten smallest species.

Thus enough examples for the training and tests were obtained. The benthic macroinvertebrate specimens had been scanned three times, but in Publication V the data from the first scanning was used. Because the sizes of groups 1-5 varied so much, the same cross-validation technique could not be used for the classification. Thus, 10 times 10-fold cross-validation to the datasets of groups 1 and 2 was applied. To group 3 the 10 times 5-fold cross-validation and to the datasets of groups 4 and 5 10 times 3-fold cross-validation was applied. The dataset from every group was standardized separately to have zero mean and unit variance. No other transformations were made for the datasets. Feature selection was made separately for each group using Scatter method and the threshold value for the selection of feature was 0.1.

In the classification seven kernel functions, the same as in Publications I-III, were used. In the case of DAGKNN four distance measures were used and these were the same as those used in Publication IV with k-NN. With DAGKNN only the odd k values which were less or equal to the smallest species size in the group were tested. The polynomial kernel functions (linear kernel function included) were tested with 40 parameter values. The RBF and Sigmoid kernel functions were tested with 1,600 parameter combinations and in Sigmoid the same agreement of $\kappa = -\delta$ was made as in Publications I–III.

In this study a new approach to parameter choice in SVM was used. Generally, it can be presented as follows: After cross-validating the group's data we obtained $10 \times \mu$ disjoint training and test sets. Firstly, each binary SVM was trained with a suitable subset from the full training set. Secondly, the accuracy of a training set was determined by giving the full training set as a test set for the trained SVMs. Thirdly, the accuracy of a real test set was evaluated. The final accuracy for the respective parameter combinations was the average of $10 \times \mu$ accuracies. Thus, every parameter combination yielded a pair of values, the first of which was the mean accuracy from the training set and the second element was the mean accuracy of the test sets. The final parameters were chosen by evaluating

$$\arg\min_i [(1 - ACC_{TRAIN,i}) + 2 \cdot (1 - ACC_{TEST,i})]$$

where i is the index for the parameter combination. Weighting was done to prevent possible tie situations and to separate those parameter values causing overfitting.

In Publication V total classification times were measured. Time was measured with every kernel function and with every distance measure used in DAGKNN. The time analysis showed that DAGKNN is faster than DAGSVM, but in the times measured it should be noted that DAGSVM was usually

tested with more parameter combinations than DAGKNN. An advantage of DAGKNN compared to DAGSVM was the simplicity of the k-NN classifier with respect to the more complex SVM classifier, where parameter tuning and finding the right kernel function can be time-consuming. Moreover, in DAGKNN the k values tested are bounded above while in SVM the parameter values are not so bounded. Differences were found among the measures tested. When using DAGKNN, Euclidean and cityblock metrics were faster to evaluate than correlation or cosine measures. Moreover, in DAGSVM the linear kernel function was the fastest to evaluate and the other polynomial kernel functions were also relatively fast. The most time-consuming kernel functions were the RBF and Sigmoid kernel functions.

A general observation in the DAGKNN results was that the small odd k values from 1 to 9 were the best alternatives in all classification problems of groups 1-5. In Publication V only the results obtained from the three best k values were presented in the case of all measures. It seems that small k values are generally good choices for benthic macroinvertebrate classification, since in Publication IV the choice of $k = 1$ resulted in the best accuracy when k-NN was used. Moreover, the choice of $k = 1$ was a good choice for solving tie situations in OVO and OVA methods.

In the classification of group 1 DAGKNN achieved 78.7% mean accuracy with cityblock metric, but the Euclidean metric resulted in only 1% lower mean accuracy. Overall, with all measures used, the range of accuracies presented was only 4%. The use of DAGSVM in group 1 classification yielded a wider range of accuracies. Now the quadratic, cubic and RBF kernel functions achieved the best accuracies and from these the quadratic kernel function reached to 86.7% accuracy. The cubic kernel function obtained 0.3% lower accuracy so they can be considered of equally good from the practical point of view. Moreover, RBF reached an accuracy of nearly 85%. Hence the difference between the best accuracies of DAGKNN and DAGSVM in group 1 classification was 8%.

The classification results in group 2 showed similar trend to that of the group 1 classification. Euclidean measure with $k = 5$ achieved the best accuracy, namely 76.5%. Again, all distance alternatives achieved accuracies quite close to each other since the interval in which the best accuracies located was within 5%. Compared to DAGSVM the best accuracies of DAGKNN lost to the corresponding accuracies of DAGSVM. The quadratic and RBF kernel functions were the best alternatives and these achieved 81.4% and 82.2% accuracies. A noteworthy detail was that now the difference between the best accuracies of DAGSVM and DAGKNN was below 6% while in the group 1 classification it was 8%.

The results for the group 3 classification were interesting. DAGKNN

yielded the best accuracies of all DAGKNN results in this group. Euclidean and cityblock metrics had above 83% accuracies and more specifically cityblock metric with $k = 3$ achieved a mean accuracy of 83.6%. Again the reported DAGKNN accuracies were very close to each other (within 5% range). With DAGSVM the linear, quadratic and RBF kernel functions worked best. RBF had the best accuracy being 85.9%, and thus the difference between the best accuracies of DAGSVM and DAGKNN was below 3%.

In the group 4 classification the results were poorer in DAGKNN than in the group 3 classification. The cityblock and Euclidean metrics were again the best alternatives. Cityblock metric achieved an accuracy of 80.4% when $k = 3$. Otherwise, accuracies below 80% were achieved. The interval in which the accuracies were located was less than 4% (76.6%-80.4%). A similarity between groups 3 and 4 classification results was found. In both cases the linear, quadratic and RBF kernel functions yielded the best accuracies. Now the RBF kernel function had the highest accuracy, 86.3%, and the accuracies of the linear and the quadratic kernel functions were remainded at 84.2% and 83.2%. The difference between the best accuracies of DAGKNN and DAGSVM was therefore around 6%.

Group 5 was the smallest group and the results presented from DAGKNN were spread over a more wider interval. Cityblock with $k = 3$ achieved the best accuracy, but now the accuracies were spread over an interval of 71.2%-79.0%. The next best measure choice was Euclidean, as frequently before. In DAGSVM the typical kernel functions were the best ones. The linear kernel function (82.6%) and RBF kernel function (82.9%) were the best alternatives while the other kernel functions achieved below accuracies 80%.

Chapter 6

Conclusions

This thesis focused on the classification of benthic macroinvertebrates [33, 37, 38, 39, 76, 89], which has generally attracted little attention [76]. General tools for automated benthic macroinvertebrate identification have proved to be difficult to develop, but BugID [50, 69] is proof that the automated processing and identification of benthic invertebrate samples can indeed be implemented in practice. The need for automated classification of benthic macroinvertebrates is great, because benthic macroinvertebrates are commonly used in biomonitoring to assess water quality. However, the actual classification of benthic macroinvertebrates is traditionally done manually by taxonomists, but this is a highly cost-intensive and time-consuming process. With the help of machine learning methods the classification task could be accomplished more effectively. Biologists and taxonomists are in a key role, also their willingness to use alternative methods in taxa identification [76].

Two datasets were examined in the thesis. The one of them contained 1,350 images from eight taxonomic groups of benthic macroinvertebrates. The other dataset contained 4,868 images from 50 species of benthic macroinvertebrates. All the image processing stages, such as feature extraction from the images was done beforehand by researchers in the Finnish Environment Institute, Jyväskylä, Finland. The classifications performed in the thesis used preprocessed data. In the preprocessing stage ImageJ software [32] was used and more details on the preprocessing are presented in [89]. All tests were performed in Matlab environment.

Altogether 16 different classification methods were used in the thesis, but the main emphasis was given to multi-class extensions of SVM. The thesis was composed of five publications and in Publications I-III and V SVM was in a key role. Publication IV included a comparison where classification methods other than SVM were used. The smaller dataset was used in Publications I-IV and in Publication V the larger dataset was examined. The

smaller dataset has also been used in articles [33, 37, 38, 39, 76, 89]. More specifically in [37, 38, 39] special attention was paid to the application of artificial neural networks such as MLP and RBF network. SVM together with OVO method were used in [38, 76] and DAGSVM in [33]. Moreover, Bayes classifier, decision tree, random forest and random Bayes forest were applied in [89].

Since automated benthic macroinvertebrate classification has been so little researched, the thesis presents a lot of new experimental information from this application. The first research question concerned the juxtaposition of SVM versus other classification methods. In Publications I-III and V SVM yielded excellent results with both datasets. The best results in Publications I (maximum accuracy with 15D feature set 96.2% and 24D feature set 97.7%) and III (maximum accuracy with 15D feature set 95.9%) outperformed other classification methods in Publication IV (maximum accuracy 94.1%). Moreover, the best accuracies in Publication II with OVO (93.7%) and OVA (91.1%) methods were better than most of the accuracies obtained in Publication IV, although in Publication II smaller feature sets were used than in Publication IV. Furthermore, the results in Publications I and III were comparable and in some cases also better than the results in [38, 37, 39, 76, 89]. However, it should be remembered that in these articles there were different test arrangements and this may affect the results. Furthermore, in [38, 37, 39, 89] only a 15D feature set was used. There are so far no other studies on the larger dataset than the publication included in this thesis. There is no point of comparison for the results in Publication V.

The second research question concerned what multi-class extension would be the best alternative for benthic macroinvertebrate classification. Publication I focused on applying OVO method and Publication II was a comparison of OVA and OVO methods. From these two alternatives OVO was the better one. From the practical point of view OVA would be better since the number of classifiers is only $O(M)$ while in OVO the corresponding number of classifiers is $O(M^2)$. However, the tests yielded a better performance for OVO. Publication III applied HAH SVM. DAGSVM was applied to a smaller dataset in [33]. This publication was not included in the thesis, but it still achieved good results. For the classification of the smaller dataset HAH SVM and OVO methods showed a great capability to classify benthic macroinvertebrate examples with good accuracy.

DAGSVM and DAGKNN were used in Publication V. Of these two classification methods DAGSVM yielded better results. The best groupwise accuracies fell into an interval of 82.2%-86.7% for DAGSVM, whereas the correponding DAGKNN accuracies were between 76.5% and 83.6%. The results with DAGKNN and DAGSVM were good when taking into account that

classification methods were applied to 10 class classification problems. Usually the general trend in classification problems is that the performance declines when the number of classes increases. At this stage DAGSVM is more suitable than DAGKNN for the classification of benthic macroinvertebrates in the larger dataset.

The third research problem was to identify the best kernel function and feature set for the classification problem. Throughout the thesis seven kernel functions were tested. Five of the kernel functions were polynomial kernel functions (the linear kernel function included) and the other were RBF and Sigmoid (called MLP in Publications I and II). In Publication I linear, quadratic and RBF kernel functions proved to be the best ones. Publication II suggested RBF and the 4th and 5th degree polynomial kernel functions for use with OVA and the performances of OVO method favoured for the quadratic, cubic and RBF kernel functions. When HAH SVM was tested in Publication III, the results had similar tendency to that in Publication II. Now the best alternatives were the quadratic, cubic and RBF kernel functions. In Publication V the quadratic, cubic and RBF kernel functions were, generally speaking, the best choices for benthic macroinvertebrate classification. Overall, a noticeable trend is discernible in the publications. The quadratic and RBF kernel functions nearly always yielded the best results. As a final choice the RBF kernel function would be the best one for benthic macroinvertebrate classification.

Feature selection can greatly affect classification. Feature reduction is an important part since making the number of features as low as possible reduces the computational demands in classification. Moreover, by eliminating the "'bad features"', classification results may improve. In the thesis feature selection was done in Publications I-IV by using the knowledge from the articles [37, 38, 39, 89] or by a random choice as in Publication II (R8D feature set) or Publication III (17D feature set). Publication V differs from the Publications I-IV since in it the feature selection was done by the Scatter method [34]. The 15D feature set (the combination of statistical and geometrical features) and statistical feature set (7D) alone proved to be good choices for benthic macroinvertebrate classification. The 15D feature set also obtained good results in [37, 38, 39, 89]. Scatter method proved to be a viable alternative to feature selection and its results were promising.

The thesis showed that benthic macroinvertebrate classification is possible with high accuracy when the classifier is properly chosen and possible parameters have been appropriately adjusted. In the thesis four multi-class extensions of SVM were investigated and all these methods managed to classify benthic macroinvertebrate examples well. Publication IV focused on other classification methods than SVM, and from these classification meth-

ods MLP, RBF network, QDA, MMDC and k-NN with Euclidean or cityblock metrics also succeeded well in classification, but they did not beat the results of SVM. Overall, the thesis has produced valuable information on automated benthic macroinvertebrate classification. Many of the classification methods used in the thesis were applied for the first time to this application.

Although this thesis had a strong emphasis on the practical side, new methodological results were obtained. New k-NN variant, DAGKNN, was presented and achieved relatively good results although it did not beat the best DAGSVM results. Tie situations are a common problem in OVA and OVO methods, and their resolution often has been left for little attention. A new tie situation resolving strategy for the majority voting method was presented based on 1-NN classifier. This strategy was used in Publications I and II. Accuracy is a commonly used performance measure in the parameter selection, but there is a drawback in its use, especially when the class distribution is skewed. In Publications I and II a different approach was usd for parameter selection. It was based on ordering the mean classwise classification rates from all parameter combinations. The final parameter combination was that including the topmost classification rates. HAH SVM is a relatively new variant in the multi-class extensions of SVM. The greatest theoretical problem is to find the optimal class divisions in nodes. In [46] a suboptimal solution based on hierarchical clustering was presented. However, in this thesis Scatter method [34] was applied to the class division problem and this was a new approach to this problem. Moreover, the search for an optimal class division was conducted in every node separately.

There remains much research to be done in the benthic macroinvertebrate classification. Firstly, there are other multi-class extensions, not used in this thesis, such as Adaptive Directed Acyclic Graph (ADAG) [36], Error Correcting Output Codes (ECOC) [14, 40] and all-together method [70, 78] to be tested. Moreover, many of the multi-class extensions were originally developed for SVM, but it would be interesting to apply OVA, OVO, HAH, DAG, ADAG and ECOC methods, for instance, to other classification methods such as k-NN, LDA or QDA. For example, in [81] OVA and OVO methods were used with k-NN and SVM classifiers and in the thesis DAG structure was used with k-NN classifier.

The excellent results from various applications speak in favour of using SVM. In benthic macroinvertebrate classification SVMs also proved to be a suitable classification method. However, there are some challenges with SVM which need to be overcome if it is to be applied in practice. The results of SVM depend heavily on the right choice of parameter values and kernel functions. The linear and, more generally, the polynomial kernel functions are better choices from the practical point of view because there is only

one parameter to be tuned up. When using RBF, the number of optimized parameters is two and in the case of Sigmoid there are three parameters to be optimized. How to choose the right kernel function is still an open question and often the only option is to test different kernel functions and choose the best one.

The search for optimal parameter values is usually made in practice by using grid-search [30] (as in this thesis) since this is the easiest way to proceed. The problem occurs if the number of parameter value combinations is high and the datasets are large. Thus the search for optimal parameter values requires much computational power because training a binary SVM classifier needs to be done several times and this is a most time-consuming process. The testing phase is minor compared to training. If an extensive parameter value search is made using an average level PC, the training phase can easily take days or weeks. Nowadays, PCs have multi-core processors, making it possible to use parallelization in the search for optimal parameter values. This speeds up the search but the computational requirements can still be high. Parallelization is feasible, since the parameter value combinations are independent of each other. In future research an adaptive approach could provide an interesting starting point for parameter value optimization.

To sum up, the thesis was concerned with applying machine learning methods to the benthic macroinvertebrate classification. Special attention was paid to SVM and its multi-class extensions. The results in Chapter 5 proved that automated benthic macroinvertebrate classification is possible with good accuracy. This thesis gave a wide perspective on the existing classification methods and introduced a new variant of k-NN. Altogether 16 classification methods were applied in the thesis. Furthermore, the thesis proposed new strategies for interesting theoretical problems such as tie situations in OVA and OVO methods and the class division problem in HAH SVM. Two datasets were used and in future the focus will be on research with the larger dataset.

Chapter 7

Personal Contributions

In the following the contribution of the present author is described. The author of the thesis worked together with Martti Juhola (hereafter referred to as MJ).

I. The present author designed the test setup and performed the tests. The publication was written by the author. MJ supervised the research and proposed the idea of examining tie situations.

II. The present author designed the test setup and performed the tests. The publication was written by the author. MJ proposed the idea of examining tie situations and supervised the research.

III. Publication III is the author's own work. Jorma Laurikkala conducted the statistical tests. MJ commented the manuscript. Scatter method was based on articles [34, 72, 73].

IV. The present author designed the test setup and performed the tests. The publication was written by the author. MJ supervised the research.

V. The present author designed the test setup and performed the tests. The publication was written by the author. MJ supervised the research. Scatter method was based on articles [34, 72, 73].

Bibliography

[1] A. Agresti. *Categorical Data Analysis*. John Wiley & Sons, New York, USA, 1990.

[2] L.A. Bartsch, W.B. Richardson, and T.J. Naimo. Sampling benthic macroinvertebrates in a large flood-plain river: considerations of study design, sample size, and cost. *Environmental Monitoring and Assessment*, 52, 425–439, 1998.

[3] M.B. Blaschko, G. Holness, M.A. Mattar, D. Lisin, P.E. Utgoff, A.R. Hanson, H. Schultz, E.M. Riseman, M.E. Sieracki, W.M. Balch, and B. Tupper. Automatic in situ identification of plankton. In *Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)*, Vol. 1, pp. 79–86, 2005.

[4] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT '92)*, pp. 144–152, 1992.

[5] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 121–167, 1998.

[6] O. Chapelle, P. Haffner, and V.N. Vapnik. Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5), 1055–1064, 1999.

[7] N. Christiani and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge, UK, 2000.

[8] K.J. Cios, W. Pedrycz, R.W. Swiniarski, and L.A. Kurgan. *Data Mining: A Knowledge Discovery Approach*. Springer-Verlag, New York, USA, 2007.

[9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3), 273–297, 1995.

[10] N. Coskun and T. Yildirim. The effects of training algorithms in MLP network on image classification. In *Proceedings of the International Joint Conference on Neural Networks*, IEEE, Vol. 2, pp. 1223–1226, 2003.

[11] L.A. Courtney and W.H. Clements. Assessing the influence of water and substratum quality on benthic macroinvertebrate communities in a metal-polluted stream: an experimental approach. *Freshwater Biology*, 47(9), 1766–1778, 2002.

[12] J. Dahl, R.K. Johnson, and L. Sandin. Detection of organic pollution of streams in southern Sweden using benthic macroinvertebrates. *Hydrobiologia*, 516, 161–172, 2004.

[13] C. Demirkesen and H. Cherifi. A comparison of multiclass SVM methods for real world natural scences. In *Proceedings of the 10th International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS 2008)*. Springer *Lecture Notes in Computer Science*, 5259, pp. 752–763, 2008.

[14] T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2, 263–286, 1995.

[15] K.-B. Duan and S.S. Keerthi. Which is the best multiclass SVM Method? An empirical study. In *Proceedings of Multiple Classifier Systems (MCS)*. Springer *Lecture Notes in Computer Science*, 3541, pp. 278–285, 2005.

[16] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*, John Wiley & Sons, New York, USA, 2nd ed., 2001.

[17] Biological Indicators of Watershed Health. U.S. Environmental Protection Agency. http://www.epa.gov/bioindicators/index.html, Accessed 25.6.2012.

[18] J. Fürnkranz. Round Robin Classification. *Journal of Machine Learning Research*, 2, 721–747, 2002.

[19] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44, 1761–1776, 2011.

[20] K.J. Gaston and M.A. O'Neill. Automated species identification: why not? *Philosophical Transactions of the Royal Society B*, 359, 655–667, 2004.

[21] H.C.J. Godfray. Challenges for taxonomy. *Nature*, 417, 17–19, 2002.

[22] L. Gray. Changes in water quality and macroinvertebrate communities resulting from urban stormflows in the Provo River, Utah, U.S.A. *Hydrobiologia*, 518, 33–46, 2004.

[23] R.W. Griffiths. Environmental quality assessment of the St. Clair River as reflected by the distribution of benthic macroinvertebrates in 1985. *Hydrobiologia*, 219, 143–164, 1991.

[24] S.R. Gunn. Support vector machines for classification and regression. Technical report, University of Southampton, 1998.

[25] Y. Hara, R.G. Atkins, S.H. Yueh, R.T. Shin, and J.A. Kong. Application of neural networks to radar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 32(1), 100–109, 1994.

[26] S. Haykin. *Neural Networks: A Compherensive Foundation*. Prentice-Hall, London, UK, 2nd ed, 1999

[27] L.-M. He, F.-S. Kong, and Z.-Q. Shen. Multiclass SVM based land cover classification with multisource data. In *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, IEEE, pp. 3541–3545, 2005.

[28] C.W. Hickey and W.H. Clements. Effects on heavy metals on benthic macroinvertebrate communities in New Zealand streams. *Environmental Toxicology and Chemistry*, 17(11), 2338–2346, 1998.

[29] J.-H. Hong and S-B. Cho. A probabilistic multi-class strategy of one-vs.-rest support vector machines for cancer classification. *Neurocomputing*, 71, 3275-3281, 2008.

[30] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification. Technical report. Available at: http://www.csie.ntu.edu.tw/cjlin/ papers/guide/guide.pdf.

[31] J. Huang, J. Lu, and C.X. Ling. Comparing naive Bayes, decision trees, and SVM with AUC and accuracy. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM'03)*, IEEE, pp. 553–556, 2003.

[32] ImageJ: public domain Java-based image processing program. Available: http://rsbweb.nih.gov/ij/, Accessed 25.6.2012.

[33] H. Joutsijoki and M. Juhola. Automated benthic macroinvertebrate identification with decision acyclic graph support vector machines. In *Proceedings of 2nd IASTED International Conference on Computational Bioscience (CompBio 2011)*. IASTED, pp. 323–328, 2011.

[34] M. Juhola and M. Siermala. A scatter method for data and variable importance evaluation. *Integrated Computer-Aided Engineering*, 19, 137–149, 2012.

[35] L. Kantola, E. Koskenniemi, R. Paavola, and M. Heikkinen. *Ohjeita järvien ja jokien pohjaeläimistöseurannan näytteenottoon ja raportointiin*, Ympäristöopas 87, Pohjois-Pohjanmaan ympäristökeskus, Oulu, 2001 (in Finnish).

[36] B. Kijsirikul and N. Ussivakul. Multiclass support vector machines using adaptive directed acyclic graph. In *Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN 2002)*, IEEE, Vol. 1, pp. 980–985.

[37] S. Kiranyaz, M. Gabbouj, J. Pulkkinen, T. Ince, and K. Meissner. Classification and retrieval on macroinvertabrate image databases using evolutionary RBF neural networks. In *Proceedings of the International Workshop on Advanced Image Technology (IWAIT)*, 2010.

[38] S. Kiranyaz, T. Ince, J. Pulkkinen, M. Gabbouj, J. Ärje, S. Kärkkäinen, V. Tirronen, M. Juhola, T. Turpeinen, and K. Meissner. Classification and retrieval on macroinvertebrate image databases. *Computers in Biology and Medicine*, 41(7), 463–472, 2011.

[39] S. Kiranyaz, M. Gabbouj, J. Pulkkinen, T. Ince, and K. Meissner. Network of evolutionary binary classifiers for classification and retrieval in macroinvertebrate databases. In *Proceedings of 2010 IEEE 17th International Conference on Image Proceedings*, pp. 2257–2260, 2010.

[40] A. Klautau, N. Jevtić, and A. Orlitsky, On nearest-neighbor error-correcting output codes with application to all-pairs multiclass support vector machines. *Journal of Machine Learning Research*, 4, 1–15, 2003.

[41] M. Kubat, R.C. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30, 195–215, 1998.

[42] N. Larios, H. Deng, W. Zhang, M. Sarpola, J. Yuen, R. Paasch, A. Moldenke, D.A. Lytle, S.R. Correa, E.N. Mortensen, L.G. Shapiro, and T.G. Dietterich. Automated insect identification through concatenated histograms of local appearance features: feature vector generation and region detection for deformable objects. *Machine Vision and Applications*, 19, 105–123, 2008.

[43] N. Larios, B. Soran, L.G. Shapiro, G. Martínez-Muñoz, J. Lin, and T.G. Dietterich. Haar random forest features and SVM spatial matching kernel for stonefly species identification. In *Proceedings of 20th International Conference on Pattern Recognition (ICPR)*, IEEE, pp. 2624–2627, 2010.

[44] N. Larios, J. Lin, M. Zhang, D. Lytle, A. Moldenke, L. Shapiro, and T. Dietterich. Stacked spatial-pyramid kernel: An object-class recognition method to combine scores from random trees. In *2011 IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 329–335, 2011.

[45] H.-G. Lax, E. Koskenniemi, P. Sevola, and P. Bagge. *Tenojoen pohjaeläimistö ympäristön laadun kuvaajana.* Vesi- ja ympäristöhallinnon julkaisuja - sarja A131, Helsinki, 1993 (in Finnish).

[46] H. Lei and V. Govindaraju. Half-Against-Half multi-class support vector machines. In *Proceedings of the 6th International Workshop on Multiple Classifier Systems (MCS 2005)*. Springer *Lecture Notes in Computer Science*, 3541, 156–164, 2005.

[47] D.D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*. Springer *Lecture Notes in Computer Science*, 1398, pp. 4–15, 1998.

[48] T.-S. Lim, W-Y. Loh, and Y-S. Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40, 203–228, 2000.

[49] A.C. Lorena, A.C.P.L.F. de Carvalho, and J.M.P. Gama. A review on the combination of binary classifiers in multiclass problems. *Artificial Intelligence Review*, 30, 19–37, 2008.

[50] D.A. Lytle, G. Martínez-Muñoz, W. Zhang, N. Larios, L. Shapiro, R. Paasch, A. Moldenke, E.N. Mortensen, S. Todorovic, and T.G. Dietterich. Automated processing and identification of benthic invertebrate

samples. *Journal of the North American Benthological Society*, 29(3), 867–874, 2010.

[51] N. MacLeod, M. Benfield, and P. Culverhouse. Time to automated identification. *Nature*, 467, 154–155, 2010.

[52] M.M. Marques and F. Barbosa. Biological quality of waters from an impacted tropical watershed (middle Rio Doce basin, southeast Brazil), using benthic macroinvertebrate communities as an indicator. *Hydrobiologia*, 457, 69–76, 2001.

[53] H. Martínez-Muñoz, W. Zhang, N. Payet, S. Todorovic, N. Larios, A. Yamamuro, D. Lytle, A. Moldenke, E. Mortensen, R. Paasch, L. Shapiro, and T.G. Dietterich. Dictionary-free categorization of very similar objects via stacked evidence trees. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, IEEE, pp. 549–556, 2009.

[54] T.M. Mitchell. *Machine Learning*. McGraw-Hill, New York, USA, 1997.

[55] W.S. Noble. What is a support vector machine? *Nature Biotechnology*, 24(12), 1565–1567, 2006.

[56] D. O'Brien, M. Gupta, R.M. Gray, and J.K. Hagene. Analysis and classification of internal pipeline images. In *Proceedings of 2003 IEEE International Conference on Image Processing (ICIP 2003)*, IEEE, Vol. 3, pp. 577–580, 2003.

[57] A.E. Ogbeibu and B.J. Oribhabor. Ecological impact of river impoundment using benthic macro-invertebrates as indicators. *Water Research*, 36, 2427–2436, 2002.

[58] T. Oki and S. Kanae. Global hydrological cycles and world water resources. *Science*, 313, 1068–1072, 2006.

[59] A. Oliver, J. Freixenet, A. Bosch, D. Raba, and R. Zwiggelaar. Automatic classification of breast tissue. In *Proceedings of the Second Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'05)*. Springer *Lecture Notes in Computer Science*, 3523, pp. 431–438, 2005.

[60] P. Parveen and B. Thuraisingham. Face recognition using multiple classifiers. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, IEEE, pp. 179–186, 2006.

[61] M.A. Pett. *Nonparametric Statistics for Health Care Research: Statistics for Small Samples and Unusual Distributions*. SAGE Publications, Thousands Oaks, California, USA, 1997.

[62] M. Peura, A. Visa, and P. Kostamo. A new approach to land-based cloud classification. In *Proceedings of the 13th International Conference on Pattern Recognition (ICPR 1996)*, IEEE, Vol. 4, pp. 143–147, 1996.

[63] J.C. Platt, N. Christiani, and J. Shawe-Taylor. Large margin dags for multiclass classification. In *Advances in Neural Information Processing Systems (NIPS 1999)* 12, pp. 547–553, 2000.

[64] J.C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, 1998.

[65] R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5, 101–141, 2004.

[66] RiverLife project. Available in Finnish (partly English) at: http://www.ymparisto.fi/riverlife. Accessed 25.6.2012.

[67] J. Saarikoski, J. Laurikkala, K. Järvelin, and M. Juhola. A study of the use of self-organising maps in information retrieval. *Journal of Documentation*, 65(2), 304–322, 2009.

[68] J. Saarikoski, J. Laurikkala, K. Järvelin, and M. Juhola. Self-organising maps in document classification: A comparison with six machine learning methods. In *Proceedings of the 10th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA 2011)*. Springer *Lecture Notes in Computer Science*, 6593, pp. 260–269, 2011.

[69] M.J. Sarpola, R.K. Paasch, E.N. Mortensen, T.G. Dietterich, D.A. Lytle, A.R. Moldenke, and L.G. Shapiro. An aquatic insect imaging system to automate insect classification. *Transactions of the ASABE*, 51(6), 2217-2225, 2008.

[70] B. Schölkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, and Beyond*. MIT Press, MA, USA, 2002.

[71] S. Siegel and N.J. Castellan Jr.. *Nonparametric statistics for the Behavioral Sciences*. Mcgraw-Hill, New York, USA, 2nd ed., 1988.

[72] M. Siermala, M. Juhola, J. Laurikkala, K. Iltanen, E. Kentala, and I. Pyykkö. Evaluation and classification of otoneurological data with

new data analysis methods based on machine learning. *Information Sciences*, 177, 1963–1976, 2007.

[73] M. Siermala and M. Juhola. Techniques for biased data distributions and variable classification with neural networks applied to otoneurological data. *Computer Methods and Programs in Biomedicine*, 81, 128–136, 2006.

[74] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, New Jersey, USA, 2002.

[75] J.A.K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9, 293–300, 1999.

[76] V. Tirronen, A. Caponio, T. Haanpää, and K. Meissner. Multiple order gradient feature for macro-invertebrate identification using support vector machines. In *Proceedings of International Conference on Adaptive and Natural Computing Algorithms (ICANNGA 2009)*. Springer *Lecture Notes in Computer Science*, 5495, pp. 489–497, 2009.

[77] K. Torkkola. Linear discriminant analysis in document classification. In *IEEE ICDM Workshop on Text Mining*, IEEE, 2001.

[78] D. Tsujinishi, Y. Koshiba, and S. Abe. Why pairwise is better than one-against-all or all-at-once. In *Proceedings of 2004 IEEE International Joint Conference on Neural Networks*, IEEE, Vol. 1, pp. 693–698, 2004.

[79] A. Vailaya, A. Jain, and H.J. Zhang. On image classification: City images vs. landscapes. *Pattern Recognition*, 31(12), 1921–1935, 1998.

[80] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, USA, 2nd ed., 2000.

[81] K. Varpa, H. Joutsijoki, K. Iltanen, and M. Juhola. Applying one-vs-one and one-vs-all classifiers in k-nearest neighbour method and support vector machines to an otoneurological multi-class problem. In *Proceedings of MIE 2011*. Studies in Health Technology and Informatics, Vol. 169, IOS Press, pp. 579–583, 2011.

[82] Y.V. Venkatesh and S.K. Raja. On the classification of multispectral satellite images using the multilayer perceptron. *Pattern Recognition*, 36(9), 2161–2175, 2003.

[83] N.J. Voelz, R.E. Zuellig, S.-H. Shieh, and J.V. Ward. The effects of urban areas on benthic macroinvertebrates in two Colorado plains rivers. *Environmental Monitoring and Assessment*, 101, 175–202, 2005.

[84] K.-M. Vuori et al.. *Suomen pintavesien tyypittelyn ja ekologisen luokittelujärjestelmän perusteet*. Suomen Ympäristökeskus, SY807, Helsinki, 2006 (in Finnish).

[85] K.-M. Vuori. Hydropsychidae-heimon vesiperhostoukat ympäristökuormituksen mittareina virtaavissa vesissä. Vesi- ja ympäristöhallinnon julkaisuja, sarja A 170, Helsinki, 1993 (in Finnish).

[86] Watershedss: A decision support system for nonpoint source pollution control. http://www.water.ncsu.edu/watershedss/, Accessed 25.6.2012.

[87] I.H. Witten and E. Frank. *Data Mining - Practical Machine Learning Tools and Techniques*. Morgan-Kaufmann, San Francisco, USA, 2nd ed., 2005.

[88] Y. Zhang and J. Zhou. A study on content-based music classification. In *Proceedings of Seventh IEEE International Symposium on Signal Processing and Its Applications*, IEEE, Vol. 2, pp. 113–116, 2003.

[89] J. Ärje, S. Kärkkäinen, K. Meissner, and T. Turpeinen. Statistical classification and proportion estimation – an application to macroinvertebrate image database. In *Proceedings of the 2010 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010)*, IEEE, pp. 373–378, 2010.

Appendices

Table 1: Species in the smaller dataset and their corresponding class sizes.

Species	Size	Species	Size
Baetis rhodani	116	*Hydropsyche siltalai*	271
Diura nanseni	129	*Isoperla* sp.	311
Heptagenia sulphurea	172	*Rhyacophila nubila*	83
Hydropsyche pellucidulla	102	*Taeniopteryx nebulosa*	166

Table 2: Species in the larger dataset and their corresponding class sizes.

Species	Size	Species	Size
Agapetus	24	*Gammarus lacustris*	38
Ameletus inopinatus	113	*Gyraulus*	20
Arctopsyche ladogensis	65	*Habrophlebia*	81
Asellus aquaticus	328	*Heptagenia fuscogrissea*	19
Atherix ibis	31	*Heptagenia sulphurea*	54
Athripsodes	13	*Hydraena*	113
Baetis digitatus	20	*Hydropsyche pellucidulla*	117
Baetis muticus	290	*Lepidostoma hirtum*	37
Baetis niger	181	*Leptophlebia*	40
Baetis rhodani	136	*Leuctra*	176
Bithynia tentaculata	292	*Limnius volckmari*	167
Caenis luctuosa	66	*Micrasema gedium*	70
Caenis rivulorum	70	*Micrasema setiferum*	291
Callicorixa wollastoni	84	*Myxas glutinosa*	228
Capnia	29	*Nemoura*	246
Ceratopsyche nevae	36	*Ophiogomphus cecilia*	11
Ceratopsyche silfvenii	222	*Oulimnius tuberculatus*	31
Ceratopogodinae	61	*Oulimnius tuberculatus larvae*	13
Cheumatopsyche lepida	126	*Paraleptophlebia*	12
Chimarra marginata	52	*Pisidium*	50
Dicranota	27	*Radix balthica*	24
Elmis aenea	185	*Sericostoma personatum*	60
Ephemera aurivillii	236	*Sigara semistriata*	39
Ephemera ignita	116	*Tanypodinae*	62
Ephemera mucronata	55	*Wormaldia subnigra*	11

Publication I

Kernel selection in multi-class support vector machines and its consequence to the number of ties in majority voting method

Henry Joutsijoki and Martti Juhola

Available at:
`http://dx.doi.org/10.1007/s10462-011-9281-3`

Artif Intell Rev
DOI 10.1007/s10462-011-9281-3

Kernel selection in multi-class support vector machines and its consequence to the number of ties in majority voting method

Henry Joutsijoki · Martti Juhola

Abstract Support vector machines are a relatively new classification method which has nowadays established a firm foothold in the area of machine learning. It has been applied to numerous targets of applications. Automated taxa identification of benthic macroinvertebrates has got generally very little attention and especially using a support vector machine in it. In this paper we investigate how the changing of a kernel function in an SVM classifier effects classification results. A novel question is how the changing of a kernel function effects the number of ties in a majority voting method when we are dealing with a multi-class case. We repeated the classification tests with two different feature sets. Using SVM, we present accurate classification results proposing that SVM suits well to the automated taxa identification of benthic macroinvertebrates. We also present that the selection of a kernel has a great effect on the number of ties.

Keywords Classification · Support vector machines · Kernel function · Majority voting method · Benthic macroinvertebrates

1 Introduction

Every day our environment encounters all kinds of challenges. Global warming and other natural catastrophes together with ecological disasters like oil catastrophes are a constant threat for changing the balance in our surrounding nature. This is why we need to improve our knowledge about the state of our every day environment. A most interesting research subjects is aquatic ecosystems due to their diversity. Aquatic ecosystems contain numerous smaller ecosystems and one of these is benthic ecosystem. Benthic ecosystems are very sensitive to any changes and the first signs where the consequences of the catastrophes or

H. Joutsijoki ( ) · M. Juhola
School of Information Sciences, University of Tampere, Kanslerinrinne 1, 33014 Tampere, Finland
e-mail: henry.joutsijoki@uta.fi

M. Juhola
e-mail: martti.juhola@cs.uta.fi

Published online: 21 August 2011

extensive changes are seen is in the number of benthic macroinvertebrates. Thus, the constant biomonitoring of aquatic ecosystems is an important issue.

Continuously growing interest towards biomonitoring has increased the amount of biologically oriented data largely. Despite this fact, the automated benthic macroinvertebrate recognition has gained little attention (Tirronen et al. 2009). The automatic recognition and classification of benthic macroinvertebrates (Larios et al. 2008; Lytle et al. 2010; Sarpola et al. 2008) are difficult tasks and so we need fast and reliable classification tools to solve this problem. Nowadays, Support Vector Machine (SVM) is a widely used classification tool in many applications. It is a relatively new method within machine learning theory while it was invented in the mid 1990s by Cortes and Vapnik (1995). SVM differs from the other classification methods significantly. Its purpose is to generate a separating hyperplane between two classes, which minimizes the generalization error, but at the same time, maximizes the margin, i.e. the distance between separating a hyperplane and the data.

SVM was originally generated for the binary (two-class) classification tasks, but later on it was generalized also for the multi-class cases for which the classification may sometimes be caught in a tie situation between classes. SVM became quickly very popular, because it gave very good results in many real-world problems. From the computational point of view, SVM is very fast and the only problem is in finding an optimal hyperplane. As a matter of fact, finding an optimal hyperplane is an NP-complete problem in general but making suitable constraints we can avoid the NP-completeness (Cortes and Vapnik 1995).

This is a preliminary paper in automated taxa identification of benthic macroinvertebrates. We have three objectives in this papers. Firstly, we investigate how one-vs-one method suits for benthic macroinvertebrate identification. Secondly, we examine what kernel function is the best alternative for automated benthic macroinvertebrate classification problem in our dataset. Thirdly, we research what happens to the number of tie situations with different kernel functions when the majority voting method is used.

In Sect. 2 we introduce basic information about macroinvertebrates. Section 3 briefly handles the basic theory of SVM in linearly separable, linearly non-separable and multi-class extension. Section 4 explains data description and result analysis. In Sect. 5 discussion and further research topics are considered.

2 Macroinvertebrate data

The following text is mainly based on (Riverlife project 2011). Before we can use SVM or any other classification method to classify benthic macroinvertebrates, there has been background work for reaching this stage. Firstly, biologists have collected numerous samples from the freshwater areas. There are two main approaches for this stage: qualitative and quantitative. The difference between them is that when using a quantitative method, we know the amount of benthic organisms per known area. If the qualitative method is used, then sample area cannot be accurately defined. Secondly, samples will be preprocessed for actual identification. Thirdly, benthic macroinvertebrates are separated from the samples and, lastly, they are recognized and classified into their own taxa.

Recognizing and classifying individual samples itself is a very challenging task. Specialists do not always have consensus about the determination of species (or even genus) of benthic macroinvertebrates in a sample. In these cases a common way is to leave the recognition to a genus level (or some upper level instead of guessing the right species). We notice these situations when in the name of the taxonomic group we have an abbreviation of sp. which tells us that the identification has been left to the genus level. For instance, in

our dataset we have seven taxonomic groups identified to a species level and one, *Isoperla* sp., has been left to a genus level. Alone in one river, there might be hundreds of species of macroinvertebrates, so the common way is to interpret taxon number instead of listing all possible species. Because the differences between the species and more generally between the taxa can be very small, taxa identification is very careful and time consuming work.

The term benthic organisms is a top level concept for all those invertebrates (animals without backbone) that are dependent on aquatic environment sometimes during their life cycle. The class of invertebrates is very large and has been diversed widely. There are thousands of different of invertebrates making their automated recognizing very challenging. Benthic macroinvertebrates have several ecological importances. Firstly, they are an important part of the food chain, because they are, among others, an essential foodsource for fishes. Secondly, benthic invertebrates are also indicators of the current situation of aquatic environment. A sudden decrease in the taxa richness of benthic macroinvertebrates can tell us about the radical changes of benthic environments or, on the contrary, a sudden increase can tell us about the improvement of the quality of environment.

Benthic macroinvertebrates suit well for monitoring the water quality, because they have a long life cycle. For example, river pearl mussel can have a life cycle of decades and crabs can have life cycle of several years. Typically benthic macroinvertebrates have a life cycle from 1 to 2 years Tirronen et al. (2009). Benthic macroinvertebrates can be used as an indicator of the toxic substances in a water system with several different ways. We present two usages. Firstly, from benthic macroinvertebrates the residues of the toxic substances can be measured. Secondly, the health of the population can be analyzed with the help of biomarkers, such as the morphological deformations of the benthic macroinvertebrates.

At our disposal, we have a wide collection of benthic macroinvertebrates in different taxonomic groups and sorting them demands high level of expertise. Benthic macroinvertebrates can be divided into four groups according to their properties. These properties are:

1. Morphological properties,
2. Functional feeding group,
3. Demands of the habitat,
4. Biome.

Morphological properties are the oldest and the most common way to make the taxon of benthic macroinvertebrates. Mostly this division reaches up to the species. Also, the size of an animal is an important morphological property. Hence, usually we see the dichotomy to benthic macroinvertebrates and benthic meioinvertebrates. Benthic meioinvertebrates are small animals that can be seen only with a microscope, and they contain their own species and groups or the youth stages of benthic macroinvertebrates. Benthic macroinvertebrates are normally that part of benthic animals which can be seen with the naked eyes.

In our dataset members from each taxonomical group were imaged by a flatbed scanner (HP Deskjet 4850) and the used software was Vuescan 8.4.57 Ärje et al. (2010). Scanning was performed with 1,600 dpi resolution and the images were saved in a JPG format. Each sample in the scanned images was saved as an individual image. Before this could have been done, the thresholding and segmentation of the images were made with ImageJ program. Moreover, images were normalized such that using dilation and erosion the benthic macroinvertebrates were deleted from the images and then the brightness of the background was equalized consistent for the whole image. Feature extraction was made also for the each benthic macroinvertebrate image by using ImageJ program.

3 Methods

3.1 Linearly separable case

In classification methods which are based on machine learning there are two starting points. These are choosing training and test sets. A training set is a collection of training examples. Let the input space be X and output space Y. Usually we have $X \subseteq \mathbb{R}^n$ and $Y = \{-1, 1\}$ for SVM. Let the training set be

$$S = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in X, y_i \in Y, i = 1, 2, \ldots, l\}$$

where the $\mathbf{x}_i$'s are the training examples and values y_i, $i = 1, 2, \ldots, l$, are the corresponding class labels from $\{-1, 1\}$.

Definition 1 Let $X \subseteq \mathbb{R}^n$ and $\mathbf{x} \in X$. A hyperplane is of the form

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \sum_{i=1}^{n} x_i w_i + b = 0 \tag{1}$$

where $\mathbf{w}$ is the weight vector and b is a bias term.

We can present a training set in the form

$$\begin{aligned} \langle \mathbf{x}_i, \mathbf{w} \rangle + b \geq 1, \quad \text{for} \quad y_i = 1 \\ \langle \mathbf{x}_i, \mathbf{w} \rangle + b \leq -1, \quad \text{for} \quad y_i = -1. \end{aligned}$$

Combining these inequalities we obtain

$$y_i[\langle \mathbf{x}_i, \mathbf{w} \rangle + b] \geq 1 \quad \text{for} \quad i = 1, 2, \ldots, l. \tag{2}$$

When we have a linearly separable case, we can rescale the weight vector and bias term such that the nearest points from the both classes (support vectors) are at the distance of $1/\|\mathbf{w}\|$ from the hyperplane. These planes can be represented as $|\langle \mathbf{w}, \mathbf{x} \rangle + b| = 1$, and hence we get that the distance between the planes is $\frac{2}{\|\mathbf{w}\|}$. This distance is called the margin. More details about the margin concept can be found from Schölkopf and Smola (2002).

For finding the formulas for the $\mathbf{w}$ and b, we need to minimize $\|\mathbf{w}\|$ in order to maximize the margin. Because minimizing $\|\mathbf{w}\|$ is equivalent with minimizing $\frac{1}{2}\|\mathbf{w}\|^2$, our goal is to minimize $\frac{1}{2}\|\mathbf{w}\|^2$ subject to the constraints in (2) (Fletcher 2009). Thus we can perform the optimization problem as a Quadratic Programming (QP) problem. Term $\frac{1}{2}\|\mathbf{w}\|^2$ need to satisfy the Karush-Kühn-Tucker conditions (Burges 1998; Schölkopf and Smola 2002; Theodoridis and Koutroumbas 2006). Hence

$$\frac{\partial}{\partial \mathbf{w}} L_P = \mathbf{0}, \tag{3}$$

$$\frac{\partial}{\partial b} L_P = 0, \tag{4}$$

$$\alpha_i \geq 0, \quad i = 1, 2, \ldots, l \tag{5}$$

$$\alpha_i[y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1] = 0, \quad i = 1, 2, \ldots, l \tag{6}$$

where α_i's are *Lagrange multipliers* and L_P is the *primal* formulation of the Lagrange function defined as

$$L_P = \tfrac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{x}_i, \mathbf{w}\rangle + b) - 1]. \tag{7}$$

We need to minimize L_P with respect to $\mathbf{w}$ and b.

Now from the Eqs. (3), (4) and (7) it follows that

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0. \tag{8}$$

Because of the constraint in (5), an optimal solution of $\mathbf{w}$ is a linear combination where there are no zero multipliers, that is,

$$\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i$$

where SV is the set of indices of the support vectors. A bias term can be calculated easily when we choose an arbitrary $i \in SV$ and use equation (6) to solve b. Thus

$$b = y_i - \langle \mathbf{x}_i, \mathbf{w}\rangle.$$

A common habit is to take an average over all of the support vectors in SV (Fletcher 2009). Putting Eqs. in (8) to (7) we obtain the *dual presentation* L_D of the primal form. Hence

$$L_D = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{9}$$

is subject to $\alpha_i \geq 0 \; \forall i$ and $\sum_{i=1}^{l} \alpha_i y_i = 0$. A new example $\mathbf{x}$ can be classified according to the sign of decision function $f(\mathbf{x}) = \langle \mathbf{x}, \mathbf{w}\rangle + b$.

3.2 Linearly non-separable case

When extending SVM to handle also non-separable data, we need a new concept called non-negative *slack variables*, $\xi_i, i = 1, 2, \ldots, l$ (Cortes and Vapnik 1995). We have to relax the constraints introduced to linearly separable case. This is done by

$$y_i[\langle \mathbf{x}_i, \mathbf{w}\rangle + b] \geq 1 - \xi_i.$$

Now the primal formulation has the form

$$L_P = \tfrac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{x}_i, \mathbf{w}\rangle + b) - 1 + \xi_i] - \sum_{i=1}^{l} \mu_i \xi_i \tag{10}$$

where $\alpha_i, \mu_i, \xi_i \geq 0 \, \forall \, i$. The error term has chosen to be $C \sum_{i=1}^{l} \xi_i$ instead of $C(\sum_{i=1}^{l} \xi_i)^k$, where C is a parameter (also called as a box constraint) to be chosen by the user (Burges 1998), because then the optimization problem has QP form and it also vanishes in dual formulation. Also, the box constraint bounds the influence of outliers in the data that would otherwise have greater α_i (Christiani and Shawe-Taylor 2003; Howley and Madden 2004).

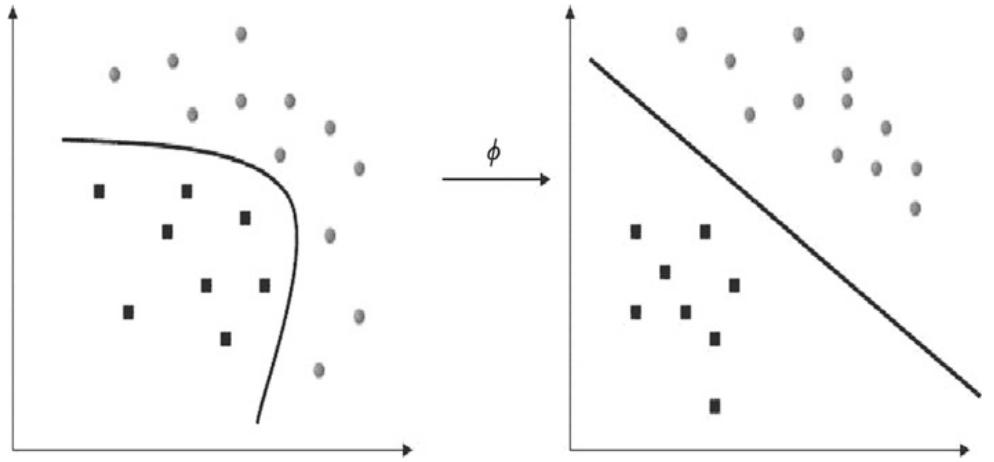


Fig. 1 Linear separation in a two-dimensional feature space

Analogously as in the linearly separable case we get

$$\frac{\partial L_P}{\partial \mathbf{w}} = \mathbf{0} \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i, \tag{11}$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \quad \sum_{i=1}^{l} \alpha_i y_i = 0, \tag{12}$$

$$\frac{\partial L_P}{\partial \xi_i} = 0 \Rightarrow \quad C = \alpha_i + \mu_i, \tag{13}$$

$$\alpha_i [y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 + \xi_i] = 0 \quad \forall i, \tag{14}$$

and

$$\mu_i \xi_i = 0. \tag{15}$$

Substituting the equality constraints in Eqs. (11)–(13) into Eq. (10) we derive the dual formulation

$$L_D = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{16}$$

subject to $0 \leq \alpha_i \leq C \ \forall i$ and $\sum_{i=1}^{l} \alpha_i y_i = 0$. The dual problem is a maximizing problem and as we see, the only difference between dual formulation in linearly separable and linearly non-separable cases is that in the latter one the Lagrange multipliers are bounded above with a box constraint C.

When the data is not linearly separable, we can use so called "kernel trick" to map the feature vector from the input space to a higher dimensional feature space by using a non-linear transformation. Illustration of the kernel trick is in the Fig. 1. A feature space is usually referred to as a Hilbert space (Schölkopf and Smola 2002) which can be thought as a generalization of the Eucledian space. Kernel trick can be justified by Cover's theorem (Cover 1965).

Definition 2 Kernel K is a symmetric function such that $K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$, $\forall \mathbf{x}, \mathbf{w} \in X$, where ϕ is a nonlinear mapping from an input space to a feature space.

When we apply mapping ϕ to Eq. (1), this obtains a form

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b = 0 \tag{17}$$

Moguerza and Muñoz (2006). Now, the Lagrangian to be minimized is

$$L_P = \tfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i[y_i(\langle \phi(\mathbf{x}_i), \mathbf{w}\rangle + b) - 1 + \xi_i] - \sum_{i=1}^{l}\mu_i\xi_i. \tag{18}$$

Thus, we get

$$\mathbf{w} = \sum_{i=1}^{l} y_i \alpha_i \phi(\mathbf{x}_i)$$

by performing the similar calculations to those in Eqs. (11)–(15). Furthermore, we can easily evaluate the threshold b in the similar manner as in the previous cases. Moreover, the dual, maximizing problem with respect to $\boldsymbol{\alpha}$ has the form

$$L_D = \sum_{i=1}^{l}\alpha_i - \tfrac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

such that $\sum_{i=1}^{l} y_i\alpha_i = 0$ and $C \geq \alpha_i \geq 0$. A new example $\mathbf{x}$ can now be classified according to the sign of (17).

A major benefit in using kernel functions is that we do not need to know the actual mapping and so all what we need to know is the inner products between the test example and the training examples. This is, from the computational point of view, a very important aspect which makes SVM so useful in many real-world problems.

Not all kernel functions are valid and the following theorem, called Mercer's theorem (Christiani and Shawe-Taylor 2003; Vapnik 2000), gives us necessary and sufficient conditions for proper kernel functions. In Mercer's theorem Hilbert space $L_2(X)$ is the set of functions f for which

$$\|f\|_{L_2} = \int_X |f(\mathbf{x})|^2 d\mathbf{x} < \infty,$$

where X is a subset of $\mathbb{R}^n$.

Theorem 1 *Let $X \subseteq \mathbb{R}^n$ be compact (closed and bounded subset). Suppose that K is a continuous symmetric function $K(\mathbf{x}, \mathbf{z}) = K(\mathbf{z}, \mathbf{x})$ such that the integral operator T_K: $L_2(X) \to L_2(X)$,*

$$(T_K f)(\cdot) = \int_X K(\cdot, \mathbf{x}) f(\mathbf{x}) d\mathbf{x}$$

is positive, that is

$$\int_{X\times X} K(\mathbf{x}, \mathbf{z}) f(\mathbf{x}) f(\mathbf{z}) d\mathbf{x}\, d\mathbf{z} \geq 0,$$

for all $f \in L_2(X)$. *Then we can expand* $K(\mathbf{x}, \mathbf{z})$ *in a uniformly convergent series (on* $X \times X$*) in terms of* T_k*'s eigen-functions* $\phi_j \in L_2(X)$*, normalized in such a way that* $\|\phi_j\|_{L_2} = 1$*, and positive associated eigenvalues* $\lambda_j \geq 0$*,*

$$K(\mathbf{x}, \mathbf{z}) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\mathbf{x}) \phi_j(\mathbf{z}).$$

We theoretically consider an infinite number of valid kernels. By choosing, for example, two Mercer's condition satifying kernel functions, their every linear combination with positive coefficients is also a valid kernel function. However, there are few commonly used kernels. These are:

1. Linear kernel $\langle \mathbf{x}, \mathbf{z} \rangle$,
2. Polynomial $(1 + \langle \mathbf{x}, \mathbf{z} \rangle)^d$ where $d \in \mathbb{N}$ is the order of the kernel,
3. Multi-Layer-Perceptron (MLP) $\tanh(\kappa \langle \mathbf{x}, \mathbf{z} \rangle + \delta)$ for parameters $\kappa > 0$ and $\delta < 0$,
4. Radial Basis Function (RBF) $e^{-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma^2}}$ where $\sigma > 0$.

We have found an optimal hyperplane by using QP optimization, but there are also some other methods for this task. Maximal margin optimization can end up to very large and time consuming and a lot of memory demanding QP problems. We can also use the Sequential Minimal Optimization (SMO) introduced by Platt (1998). SMO divides large QP problems into smaller ones to be solved separately and analytically. This can speed up the training process significantly. Another way to optimise the margin is to use the Least Square (LS) method. LS difference is in the penalty term $\frac{C}{2} \sum_{i=1}^{l} \xi_i^2$. In LS the Lagrangian has the form

$$L_P = \tfrac{1}{2}\|\mathbf{w}\|^2 + \tfrac{C}{2} \sum_{i=1}^{l} \xi_i^2 - \sum_{i=1}^{l} \alpha_i [y_i (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b) - 1 + \xi_i].$$

Detailed information about LS can be found from Suykens and Vandewalle (1999).

3.3 Multi-class extension

Suppose that we have a training data of M classes where $M > 2$. We have two alternatives for the multi-class extension. These are one versus all (OVA) and one versus one (OVO) methods. In OVA we construct M binary classifiers such that each one of them is trained to separate one class from the other. A test sample is input to each classifier and a final class for the test sample is given according to the winner-takes-all method. In OVO we first train $M(M-1)/2$ individual binary classifiers between all pairs of classes. When a new sample is classified, it is tested with every binary classifier, and then the final class is chosen for the test sample with some voting method. In this work we use the OVO method.

There are several voting methods to solve the final class for a test sample. In this work we use *the majority voting method*. We obtain, from each classifier, a vote to which class a sample should belong. Then the method counts which class gains the highest number of votes and choose it as the final result. The method is very easy and it brings good results, but problems arise when ties occur. It cannot solve these cases directly and we need some additional tools to solve the problem. Other voting methods like the winner-takes-all method and error-correcting codes are also typically used (Hong et al. 2008).

However, the majority voting system in OVO consist of a problem when the voting goes to a tie situation with two or more classes. These cases need special attention. Hong and Cho (2008) introduced a tie solving method by using Naïve Bayes classifiers. In this work we

used the traditional k-nearest-neighbor (k-NN) method to solve the tie situations between the classes. Always when a tie appeared, we trained k-NN with the same training data of the tied classes as used with the SVMs. After that we classified the problematic test sample.

In order to prevent the situation that k-NN gives a new tie situation, which is always possible, when we are dealing with more than two classes, we need to consider a carefully used k value. In a binary problem we can choose the k value to be odd. If k is even, then a tie can occur again. In multi-class cases for every k value, whether it is even or odd and greater than one, there is a chance that a tie can appear again. In this work, we have an eight class classification problem. Thus, when using OVO, we have 28 binary SVM classifiers which means 28 votes for each test example. Then, theoretically, our worst case scenario is that we have a seven class tie, each class having four votes for each. In order to prevent this situation we have chosen $k = 1$ which gives certainly a unique solution for each tie situation.

Tie analysis is an interesting issue and it has not been investigated as much as it should have been done. Theoretically these examples which are not classifed into any class or are not classified unambiguously need special tools. Here we also investigate the number of ties in the cases of kernel functions used. How to choose a right kernel is still an open question. A novel question is how the selection of a kernel function will effect the number of ties. If the amount of ties is small, then these examples do not have a major impact on the final classification distribution, but if the number of ties is greater, then the tie situations will effect, maybe crucially, the classification itself.

4 Experimental tests

4.1 Data description

The image database consists of altogether 1,350 images from eight different taxonomical groups of benthic macroinvertebrate: *Baetis rhodani, Diura nanseni, Heptagenia sulphurea, Hydropsyche pellucidulla, Hydropsyche siltalai, Isoperla* sp., *Rhyacophila nubila* and *Taeniopteryx nebulosa*. In the following we will refer to the classes with capital letters A-H and in Fig. 2 there is an example image from every class.

Fig. 2 An example image on every taxonomical group. Taxa order from *upper left* to *bottom right* is A-H.

In the preprocessing stage (see details Ärje et al. 2010) features have been calculated such that, first, the whole image has been thresholded and then all individual particles have been searched for what in this case means benthic macroinvertebrates. Outliers have been removed according to their area with predicted conditions in an image. Features have been calculated from the images by using a free public Java-based image processing program called ImageJ (ImageJ).

The data includes 26 attributes (ImageJ). Excluding the identification number and area fraction we have used all the given features. In (Kiranyaz et al. 2010, 2011) divided features into two categories: geometrical and statistical. They used 15 features from all 26 possible. The statistical features used are: {Mean, Standard deviation Mode, Median, Integrated Density, Kurtosis, Skewness}, where the Integrated Density means the sum of the values of the pixels in the image or selection. Respectively the geometric features were: {Area, Perimeter, Width, Height, Feret's Diameter, Major Minor, Circularity}, where Feret's Diameter is the longest distance between any two points along the selection boundary. Other features which we used but not in (Kiranyaz et al. 2010, 2011) are: {Min, Max, X, Y, XM, YM, BX, BY, Angle} where {X, Y, XM, YM, BX, BY} are the features concerning coordinates of a smallest rectangle enclosing the selection in the image or coordinates of the center of the fitted ellipse. A reader can find the accurate description about all features from (ImageJ). We repeated the classification procedure for the feature sets, both smaller and extended one. In the result tables these sets are referred to with the abbreviations 15D and 24D, which means the dimensions of each sample.

The classification process was repeated 250 times for each of 28 classifier, for which means of the results were calculated as a final result. Each column vector in the data matrix was normalized to have zero mean and unit variance. In the classification procedure a training set was selected to be 90% of the whole data set and the rest 10% was the test set. With these assumptions sizes of the training and test sets were 1,215 and 135. Training and test sets were selected by weightening to obtain results more truthfully when the sizes of the groups alternated from 83 to 311. Weightening was made by the sizes of the taxonomic groups. In each round the elements of the training and test sets were chosen randomly. In training and testing the Bioinformatics Toolbox of Matlab was used.

From Table 1 we see the used parameter values. The second column represents the box constraint (bc) value, which corresponds to the regularization value in a soft-margin case. Parameters κ, δ and σ are the parameters in MLP and RBF kernels. The last column depicts the k value for which each classifier was tested in the case of a tie situation in voting.

Table 1 Parameter values of kernels

Kernel	Box constraint	Kernel parameters			
	bc	κ	δ	σ	k
Linear	$0.5, \ldots, 10$	–	–	–	1
MLP	$0.5, \ldots, 10$	$0.5, 1.0, \ldots, 10$	$-10, -9.5, \ldots, -0.5$	–	1
Polynomial ($d = 2$)	$0.5, \ldots, 10$	–	–	–	1
Polynomial ($d = 3$)	$0.5, \ldots, 10$	–	–	–	1
Polynomial ($d = 4$)	$0.5, \ldots, 10$	–	–	–	1
Polynomial ($d = 5$)	$0.5, \ldots, 10$	–	–	–	1
RBF	$0.5, \ldots, 10$	–	–	$0.5, 1.0, \ldots, 10$	1

In classifying the data, each of the 28 SVM classifiers was trained and tested with linear kernel together with 20 parameter values. We also tested the polynomial kernels, degree $d \in \{2, 3, 4, 5\}$, with 20 parameter values. In the MLP kernel case we made the agreement that $\kappa = -\delta$. Also, the values σ were chosen to be equal with κ and $-\delta$ in order to get the comparison of the classification results more consistent. Hence the parametric space is equal to the parameters κ, δ and σ. So both MLP and RBF were tested with $20 \cdot 20 = 400$ combinations of the parameters. We performed the tests for the both feature sets, 15D and 24D, with the same combinations of the parameters.

4.2 Results

We used seven different kernel functions altogether. Tables 2 and 3 show the results of the linear kernel. As seen, we obtained excellent performances with the 24D feature set. Classes D, F, G and H were classified perfectly and the other classes with over 90% accuracy. For the classes B, C and E expanded feature set did not bring any crucial information which would have improved the recognition accuracy. The recognition of other classes was improved at least 4% with the 24D feature set. In Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 each row indicates the distribution, how the test samples from specific class have been classified within all the classes. Tables are not symmetric, because the group sizes alternate within the classes.

Table 2 Results by linear kernel with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**94.5**	0.0	0.0	0.0	5.5	0.0	0.0	0.0
Class B	0.0	**96.5**	2.9	0.0	0.6	0.0	0.0	0.0
Class C	1.5	0.0	**92.0**	0.0	6.5	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**93.2**	0.0	0.0	6.8	0.0
Class E	1.8	0.8	1.6	0.0	**95.8**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**95.1**	0.0	4.9
Class G	0.0	0.0	0.0	5.3	0.0	0.0	**94.7**	0.0
Class H	0.0	0.0	0.1	0.0	0.0	4.7	0.0	**95.2**

Table 3 Results by linear kernel with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**98.9**	0.0	0.0	0.0	1.1	0.0	0.0	0.0
Class B	0.0	**96.2**	2.9	0.0	0.9	0.0	0.0	0.0
Class C	1.7	0.0	**91.3**	0.0	7.0	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0
Class E	1.1	0.4	2.3	0.0	**96.2**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0
Class G	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0
Class H	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**

Table 4 Results by quadratic kernel with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**93.5**	0.0	0.1	0.0	6.4	0.0	0.0	0.0
Class B	0.0	**98.0**	0.6	0.1	1.3	0.0	0.0	0.0
Class C	0.0	0.2	**96.0**	0.0	3.8	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**98.2**	0.0	0.0	1.8	0.0
Class E	0.6	1.1	1.5	0.0	**96.8**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**96.0**	0.0	4.0
Class G	0.0	0.0	0.0	3.1	0.0	0.0	**96.9**	0.0
Class H	0.0	0.0	0.0	0.0	0.0	5.0	0.0	**95.0**

Table 5 Results by quadratic kernel with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**95.8**	0.0	0.0	0.0	4.2	0.0	0.0	0.0
Class B	0.0	**97.7**	0.9	0.0	1.4	0.0	0.0	0.0
Class C	0.6	0.6	**91.8**	0.0	7.0	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**99.9**	0.0	0.0	0.1	0.0
Class E	1.3	1.0	1.7	0.0	**96.0**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0
Class G	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0
Class H	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**

Table 6 Results by cubic kernel with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**86.9**	0.0	1.2	0.0	11.7	0.0	0.0	0.2
Class B	0.0	**94.6**	1.5	0.1	1.7	0.5	1.5	0.1
Class C	0.1	1.7	**84.0**	0.0	13.8	0.3	0.0	0.1
Class D	0.1	0.0	0.4	**94.7**	0.0	1.0	3.1	0.7
Class E	2.0	1.4	3.8	0.2	**92.3**	0.2	0.0	0.1
Class F	0.1	0.0	0.0	0.0	0.0	**89.9**	0.0	10.0
Class G	0.0	0.1	1.1	3.0	0.0	0.1	**94.5**	1.2
Class H	0.4	0.0	0.0	0.0	0.0	21.7	0.0	**77.9**

Changing the kernel into quadratic did not make any major differences comparing to the linear kernel. In the extended feature set class B was recognized slightly better than in the case of the linear kernel, but the difference is very small. The most interesting aspect in this kernel was that class C was classified over 4% better in the 15D feature set than in the extended one. More details can be found from the Tables 4 and 5.

Table 7 Results by cubic kernel with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**87.9**	0.1	2.3	0.0	9.7	0.0	0.0	0.0
Class B	0.4	**94.8**	2.9	0.0	1.1	0.2	0.0	0.6
Class C	0.4	4.9	**76.1**	0.0	18.6	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**97.1**	0.0	2.0	0.8	0.1
Class E	2.9	2.9	9.6	0.0	**84.6**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**99.9**	0.0	0.1
Class G	0.0	0.0	0.0	0.0	0.0	1.2	**98.5**	0.3
Class H	0.0	0.0	0.0	0.0	0.0	0.8	0.0	**99.2**

Table 8 Results by polynomial kernel ($d = 4$) with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**87.4**	0.3	0.5	0.0	10.0	1.0	0.0	0.8
Class B	1.8	**90.8**	0.8	3.5	2.4	0.0	0.7	0.0
Class C	0.8	5.9	**70.7**	0.0	21.4	0.1	0.0	1.1
Class D	1.6	0.0	0.3	**92.7**	0.2	0.2	3.4	1.6
Class E	2.6	2.7	6.2	0.3	**86.7**	0.5	0.1	0.9
Class F	0.3	0.0	0.2	0.1	0.5	**83.3**	0.0	15.6
Class G	0.6	1.0	0.8	3.5	0.5	0.3	**88.9**	4.4
Class H	0.3	0.0	1.8	0.0	1.9	28.7	0.0	**67.3**

Table 9 Results by polynomial kernel ($d = 4$) with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**85.4**	0.1	1.7	0.0	12.5	0.0	0.0	0.3
Class B	1.4	**91.6**	4.0	0.6	2.3	0.1	0.0	0.0
Class C	1.5	4.5	**67.2**	0.0	25.3	0.0	0.0	1.5
Class D	2.1	0.3	0.0	**93.2**	0.2	0.3	3.2	0.7
Class E	3.4	2.9	11.9	0.0	**80.8**	0.3	0.1	0.6
Class F	0.1	0.0	0.0	0.0	0.0	**97.8**	0.0	2.1
Class G	0.2	0.0	0.7	1.5	1.0	0.3	**91.2**	5.1
Class H	0.0	0.0	0.5	0.0	1.2	7.2	0.0	**91.1**

Altering the degree of the polynomial kernel into 3 made the results worse than in the two aforementioned ones. Classes A, C and E yielded clearly worse results than before, which is a sign of possible overfitting. Otherwise, the rest of the classes was quite well classified. Table 6 shows that classes C and E have been classified better with the 15D feature set than with the extended feature set. The rest of the classes has poorer performances than those of the corresponding classes in Table 7. The difference is considerable in class H. The same kind of tendency, as in the 3rd degree of the polynomial kernel, was also present for the 4th degree of the polynomial kernel. With the smaller feature set we achieved better results in

Table 10 Results by polynomial kernel ($d = 5$) with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**79.3**	0.6	0.2	0.0	12.1	3.5	0.0	4.3
Class B	0.2	**83.2**	1.3	4.8	1.9	2.4	5.4	0.8
Class C	0.1	9.1	**60.9**	0.6	24.2	2.9	0.2	2.0
Class D	1.5	0.1	0.5	**87.1**	1.8	3.2	3.7	2.1
Class E	2.8	3.1	8.1	0.2	**82.4**	1.5	0.1	1.8
Class F	0.4	0.0	0.2	0.0	1.1	**85.5**	0.0	12.8
Class G	0.5	1.8	0.8	3.7	6.2	4.8	**79.2**	3.0
Class H	0.3	0.0	3.0	0.0	5.8	29.0	0.0	**61.9**

Table 11 Results by polynomial kernel ($d = 5$) with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**73.7**	0.3	6.3	0.0	18.7	0.4	0.0	0.6
Class B	3.5	**83.4**	3.6	0.0	7.0	0.5	0.0	2.0
Class C	2.1	7.1	**56.1**	0.0	33.9	0.4	0.0	0.4
Class D	1.4	0.0	0.0	**84.1**	0.0	7.8	4.5	2.2
Class E	4.4	3.5	13.1	0.0	**78.5**	0.3	0.0	0.2
Class F	0.0	0.0	0.0	0.0	0.1	**95.6**	0.0	4.3
Class G	0.0	0.0	0.3	1.8	1.1	16.4	**77.1**	3.3
Class H	0.5	0.0	1.1	0.0	2.4	16.8	0.0	**79.2**

Table 12 Results by RBF kernel ($\sigma = 7.0$) with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**93.9**	0.0	0.0	0.0	6.1	0.0	0.0	0.0
Class B	0.0	**97.1**	2.3	0.0	0.6	0.0	0.0	0.0
Class C	1.9	0.0	**92.7**	0.0	5.4	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**94.1**	0.0	0.3	5.6	0.0
Class E	3.2	2.1	1.2	0.0	**93.5**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**93.9**	0.0	6.1
Class G	0.0	0.0	0.0	3.8	0.0	0.0	**96.2**	0.0
Class H	0.0	0.0	0.0	0.0	0.0	4.9	0.0	**95.1**

classes A, C and E from which class C has been difficult to recognize for every kernel. When $d = 5$, classes A, C, D, E and G were better classified with the smaller number of features. This assures the possible overfitting problem in the higher degrees of polynomial kernels. Tables 8, 9, 10 and 11 show the exact results when $d = 4$ and $d = 5$.

RBF and MLP kernels are very interesting cases because their results are total opposites. With RBF when $\sigma = 7.0$ and $bc = 8.0$ classification results were excellent and as good as

Table 13 Results by RBF kernel ($\sigma = 7.0$) with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**98.7**	0.0	0.0	0.0	1, 3	0.0	0.0	0.0
Class B	0.0	**96.8**	2.4	0.0	0.8	0.0	0.0	0.0
Class C	1.9	0.0	**91.7**	0.0	6.4	0.0	0.0	0.0
Class D	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0
Class E	3.0	1.7	1.4	0.0	**93.9**	0.0	0.0	0.0
Class F	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0
Class G	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0
Class H	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**

Table 14 Results by MLP kernel ($\kappa = -\delta = 0.5$) with 15D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**67.8**	0.1	3.9	0.6	9.9	9.2	0.9	7.6
Class B	0.2	**85.4**	4.4	2.3	3.8	0.4	2.9	0.6
Class C	4.3	11.6	**45.3**	4.8	15.0	4.4	6.6	8.0
Class D	1.1	7.2	2.6	**74.8**	2.4	2.3	8.0	1.6
Class E	13.4	3.8	18.0	1.8	**43.9**	7.2	2.3	9.6
Class F	10.3	0.2	3.2	1.7	5.6	**48.8**	5.5	24.7
Class G	1.5	7.5	4.7	12.9	4.1	5.8	**55.3**	8.2
Class H	8.5	0.5	6.7	1.3	6.9	21.3	6.5	**48.3**

Table 15 Results by MLP kernel ($\kappa = -\delta = 0.5$) with 24D feature set and $bc = 8.0$ (Boldfaced numbers indicate correctly classified samples in specific class in percentage)

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Class A	**72.3**	0.1	3.2	0.2	9.5	5.0	1.1	8.6
Class B	0.2	**71.7**	8.8	4.8	6.8	0.6	5.6	1.5
Class C	4.4	10.9	**32.6**	5.6	19.0	3.9	8.4	15.2
Class D	0.8	13.6	3.8	**55.4**	3.0	4.6	16.4	2.4
Class E	13.6	3.9	20.0	2.5	**28.6**	6.5	4.8	20.1
Class F	9.4	0.4	3.9	3.1	7.1	**42.0**	9.5	24.6
Class G	2.5	6.6	6.9	14.4	4.9	9.1	**46.7**	8.9
Class H	9.5	0.8	8.1	2.3	8.9	19.0	9.3	**42.1**

with the linear kernel. One half of the classes was perfectly recognized and the other half had over 90% accuracy. With the smaller feature set every class was recognized with over 90% accuracy, but none of them had perfect 100% result. MLP was clearly the worst alternative for classifying the benthic macroinvertebrate images. Only two classes reached over 70% accuracy in both feature set options, and the major part of the rest classes had less than 50% performance. In RBF, when we fixed σ and changed the box constraint value, final results did

Table 16 Number of the ties

	Mean	Standard deviation
Linear	0.4 (0.3)	0.6 (0.5)
Polynomial ($d = 2$)	1.4 (1.0)	1.1 (1.0)
Polynomial ($d = 3$)	7.8 (6.8)	2.7 (2.5)
Polynomial ($d = 4$)	10.2 (13.0)	2.9 (3.4)
Polynomial ($d = 5$)	16.0 (21.1)	3.7 (4.2)
MLP	47.8 (43.2)	5.6 (5.7)
RBF	0.1 (0.4)	0.4 (0.6)

Numbers in the parenthesis are the results when used the smaller feature set and correspondingly other numbers are from the extended feature set

Table 17 Weighted means of the correctly classified samples in percentage

	15D	24D
Linear	94.8	97.7
Polynomial ($d = 2$)	96.2	97.6
Polynomial ($d = 3$)	88.9	91.9
Polynomial ($d = 4$)	82.5	87.3
Polynomial ($d = 5$)	77.8	80.1
RBF	94.3	97.3
MLP	54.8	45.0

not alternate significantly. The same situation was when we fixed κ in MLP. Then changing *bc* did not change the final results barely all. Moreover, the same situation was also in the case of the linear kernel and for all polynomial kernels used. Accurate information is presented in Tables 12, 13, 14 and 15.

How did changing the kernel function effect the number of ties? The answer is that changing the kernel did have a major impact on the amount of ties. The mean altered from less than 1 to a over 40. The interval is very large and it creates several other questions about the importance of tie analysis. Table 16 show the means and standard deviations of the numbers of ties. As expected, a number of ties is directly connected to the classification results. When a kernel fits well to the data, a number of ties is low and, on the contrary, if a kernel does not fit to the data, ties are more frequent. We have calculated the statistical parameters in both feature set cases. An interesting aspect is that the amount of ties did not differ largely, although we increased the number of features from 15 to 24. This can be seen from Table 16 by comparing the numbers, which are in the parenthesis and which are not in the parenthesis. Table 16 shows that the ties need to be concerned when dealing with SVM. To Table 17 we gathered the weighted means of correctly classified samples from the result Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 for easyning the analyzation of the results.

5 Discussion

The automated recognition of benthic macroinvertebrates is a very demanding task because of the huge variety of species. Applying SVM in the classification of the benthic macroinvertebrates has been under research only for a short time and so there remains a lot of research

to be done in this area. Macroinvertebrates are an important part of the aquatic ecosystems and they are an important indicator of the current state of the aquatic ecosystem.

Our results tell that SVM is a very promising tool in classifying the benthic macroinvertebrates. There has been an open question from the mid 1990s since the invention of SVM what kernel would be the right choice for a given application. General rules have not been discovered so far, and here we have researched the choice of a right kernel function for the macroinvertebrate image data set. The results showed that the basic and simplest kernel, linear kernel, gave very good results. Furthermore, RBF gave also very good performance.

Choosing a right kernel for an application is an open question (Moguerza and Muñoz 2006) and finding right parameter values is also an interesting question. In practice this is solved by forking the best interval of parameter values or fixing the interval in the beginning and finding the best combination of values in it. Theoretically, the most interesting question is the ties occurring in voting. These cases are difficult to handle and are often solved by selecting randomly the class within the tied classes or to choose the class which has the most training examples.

However, these methods are not 100% accurate and this is why we need to develop more theoretical tools for solving such situations. As seen from Table 16 that ties are not so rare events that they should be dismissed. On the contrary, tie situation can exist in over 30% of all test examples which has a major effect on classification. Kernel choice and its effect on the number of ties is a question that needs more research.

Moreover, feature selection is very important question. We made the classification tests with two different feature sets, but further research is needed. The separation of features into two categories brings up the question how classification would succeed if we used either statistical or geometrical features. Moreover, would the feature selection effect the number of ties? Furthermore, in the future research it is important to investigate how classification succeeds with other multi-class methods such as one-versus-all and LIBLINEAR (Fan et al. 2008). Also, it is interesting to test other classification methods than support vector machines to this application area.

Acknowledgments We would like to thank the Finnish Environment Institute for the data. The first author is thankful to Tampere Graduate Program in Information Science and Engineering for the support.

References

Ärje J, Kärkkäinen S, Meissner K, Turpeinen T (2010) Statistical classification and proportion estimation—an application to a macroinvertebrate image database. In: Proceedings 20th international IEEE workshop on machine learning for signal processing (MLSP 2010), Kittilä, Finland, pp 373–378

Burges CJC (1998) A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov 2:121–167

Christiani N, Shawe-Taylor J (2003) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge

Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20:273–297

Cover TM (1965) Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE Trans on Electr Comp EC-14

Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ (2008) LIBLINEAR: a library for large linear classification. J Mach Learn Res 9:1871–1874

Fletcher T (2009) Support vector machines explained. Available: http://www.cs.ucl.ac.uk/staff/T.Fletcher/. Accessed 17 June 2011

Hong JH, Cho SB (2008) A probabilistic multi-class strategy of one-vs.-rest support vector machines for cancer classification. Neurocomputing 71:3275–3281

Hong JH, Min JK, Cho UK, Cho SB (2008) Fingerprint classification using one-vs-all support vector machines dynamically ordered with naïve bayes classifiers. Pattern Recognit 41:662–671
Howley T, Madden MG (2004) The genetic evolution of kernels for support vector machine classifiers. In: Proceedings of AICS-2004 15th Irish Conference on Artificial Intelligence & Cognitive Science
ImageJ: public domain Java-based image processing program. Available: http://rsbweb.nih.gov/ij/docs/index.html
Kiranyaz S, Gabbouj M, Pulkkinen J, Ince T, Meissner K (2010) Classification and retrieval on macroinvertebrate image databases using evolutionary RBF neural networks. In: Proceedings of the International Workshop on Advanced Image Technology (IWAIT), 11–12 January 2010 Kuala Lumpur, Malaysia
Kiranyaz S, Ince T, Pulkkinen J, Gabbouj M, Ärje J, Kärkkäinen S, Tirronen V, Juhola M, Turpeinen T, Meissner K (2011) Classification and retrieval on macroinvertebrate image databases. J Comput Biol Med 41(7):463–472
Larios N, Deng H, Zhang W, Sarpola M, Yuen J, Paasch R, Moldenke A, Lytle DA, Correa SR, Mortensen EN, Shapiro LG, Dietrich TG (2008) Automated insect identification through concatenated histograms of local appearance features: features vector generation and region detection for deformable objects. Mach Vision Appl 19(2):105–123
Lytle DA, Martínez-Muñoz G, Zhang W, Larios N, Shapiro L, Paasch R, Moldenke A, Mortensen EN, Todorovic S, Dietrich TG (2010) Automated processing and identification of benthic invertebrate samples. J N Am Benthol Soc 29(3):867–874
Moguerza JM, Muñoz A (2006) Support vector machines with applications. Stat Sci 21(3):332–336
Platt JC (1998) Sequential minimal optimization: a fast algorithm for training support vector machines. In: Technical Report MSR-TR-98-14 Microsoft Research
Riverlife project. Available in Finnish: http://www.ymparisto.fi/riverlife, Accessed 20 June 2011
Sarpola MJ, Paasch RK, Dietrich TG, Lytle DA, Mortensen EN, Moldenke AR, Shapiro L (2008) An aquatic insect imaging device to automate insect classification. Trans ASABE 51(6):2217–2225
Schölkopf B, Smola AJ (2002) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge
Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9:293–300
Theodoridis S, Koutroumbas K (2006) Pattern recognition, 3rd edn. Academic Press, Edinburgh
Tirronen V, Caponio A, Haanpää T, Meissner K (2009) Multiple order gradient feature for macroinvertebrate identification using support vector machines. In: Proceedings of the adaptive and natural computing algorithms. Lecture notes in computer Science, vol 5495, pp 489–497
Vapnik VN (2000) The nature of statistical learning theory, 2nd edn. Springer, Berlin

Publication II

Comparing the One-vs-One and One-vs-All Methods in Benthic Macroinvertebrate Image Classification

Henry Joutsijoki and Martti Juhola

Available at:
`http://dx.doi.org/10.1007/978-3-642-23199-5_30`

Comparing the One-vs-One and One-vs-All Methods in Benthic Macroinvertebrate Image Classification

Henry Joutsijoki and Martti Juhola

School of Information Sciences
University of Tampere, Kanslerinrinne 1, FI-33014 Tampere, Finland
{henry.joutsijoki,martti.juhola}@uta.fi

Abstract. This paper investigates automated benthic macroinvertebrate identification and classification with multi-class support vector machines. Moreover, we examine, how the feature selection effects results, when one-vs-one and one-vs-all methods are used. Lastly, we explore what happens for the number of tie situations with different kernel function selections. Our wide experimental tests with three feature sets and seven kernel functions indicated that one-vs-one method suits best for the automated benthic macroinvertebrate identification. In addition, we obtained clear differences to the number of tie situations with different kernel funtions. Furthermore, the feature selection had a clear influence on the results.

Keywords: Support vector machines, machine learning, benthic macroinvertebrates, water quality, kernel function.

1 Introduction

Water is a part of our every day life. We drink it daily and we use it in our daily routine activities. Thus, we often can take it for a granted thing. Nevertheless, when we encounter problems, like problems in the water systems and the occurrences of natural catastrophes or chemical disasters, the actual need of water and its quality are quickly in people's minds. Macroinvertebrates are used in biomonitoring to study human induced changes on aquatic ecosystems [20]. Benthic macroinvertebrates are excellent indicators of the state of ecosystems and water quality, because they react quickly to any changes in water quality. If the number of benthic macroinvertebrate species is high and the population of each species is large, ecosystem is in good condition. Otherwise, we can assume that something unnatural has happened.

Support vector machines (SVM), compared to other classification methods, have been under research only for a short time of period, since they were developed by Vapnik et al. [4,19] no later than in the mid 1990s. SVM has gained wide popularity and the ongoing research is very active. Especially, the selection of a right kernel function [2] and the optimal parameter search have been a relevant research topic from the beginning. SVM has been applied to numerous applications such as visual object recognition [8], text categorization [17], automated musical instrument classification [13] and automated video content analysis [12], but automated benthic macroinvertebrate identification [9,10,11,14,18,20] has got very little attention. Traditionally classification of macroinvertebrate samples has been made by the specialists, but this approach

P. Perner (Ed.): MLDM 2011, LNAI 6871, pp. 399–413, 2011.

is time-consuming and expensive [20]. Furthermore, when hundreds or even thousands of samples are classified manually one by one, the probability of human mistakes increases. By automatization of the identification process as accurate as possible, we can reduce the costs of human-made taxa identification and the use of time to it.

The identification of benthic macroinvertebrates and classification of their images are difficult problems from the pattern recognition point of view, because differences between the species of benthic macroinvertebrates can be very small. Moreover, the sizes, positions and shapes differ in each image raising the level of the problem even higher. Generally speaking, the purpose of the articles and researches concerning classification methods is to find and develop thoretical tools such that they work in real-world applications as well as possible. Applications, especially, with biologically stressed datas, reveal us how complex and diversed the nature surrounding us is really, and usually the nature proves to be more complex than we think, and the results give often suprising aspects. This paper clarifies the dissimilarities and the similarities between the taxonomical groups from the pattern recognition aspect. We can analyze with the help of the results, which species can be separated and cannot be separated by means of existent classification methods.

The research of automated benthic macroinvertebrate has also influence on other areas. This is a relatively new application area in pattern recognition and, therefore, there is a lot of research in this area, while the Globe consists of thousands of species of benthic macroinvertebrates and we have examined only a small portion of them by the methods of machine learning. The investigation and development of benthic macroinvertebrate identification help us to improve the water quality control. Hence, we can notice and react to the changes of the aquatic ecosystems and examine the reasons behind the changes. On the basis of pattern recognition we can automate the mechanical identification process and act better for the improment of freshwater areas.

We have three objectives in this paper. Firstly, we investigate, whether one-vs-all is a better method than one-vs-one in classifying the benthic macroinvertebrate images. Secondly, we examine how the feature selection effects the classification in one-vs-one and one-vs-all methods, and thirdly, we examine the number of tie situations, when using the one-vs-all method and the majority voting method in the one-vs-one method.

In Sect. 2 we describe the general theory of binary SVMs and we discuss the multiclass extensions and their pros and cons. In Sect. 3 we describe the test arrangements and give the data description as well as we present the experimental results and analyze them. Section 4 is left for conclusions and further research topics.

2 Method

2.1 Linearly Separable Case

Support Vector Machines [1,3,4,19] were originally developed for the binary classification problems. Let the input space be $X \subseteq \mathbb{R}^n$ and the output space $Y = \{-1, 1\}$. Let the training set be $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\} \in \mathbb{R}^n \times Y$, where $\boldsymbol{x}_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, l$, are the training examples and values $y_i \in Y$ are their corresponding class labels. Our task is to find a hyperplane $f(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{w} \rangle + b = 0$ separating the two classes from each other.

Separating hyperplane is not an unique one, but adding the constraint that the margin, i.e., the distance between the data and the hyperplane, is maximized, then we find a unique hyperplane. We can represent our training data in the form $y_i[\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle + b] \geq 1$,$i = 1, 2, \ldots, l$. When we have a linearly separable case, we can rescale the weight vector $\boldsymbol{w}$ and the bias term b such that the hyperplane is at the distance of $\frac{1}{\|\boldsymbol{w}\|}$ from the closest points (support vectors) of both classes. In other words, canonical hyperplanes, where the support vector lies on, can be represented as $|\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle + b| = 1$ for some i. Hence, the margin (the distance between the canonical hyperplanes) is the equal to $\frac{2}{\|\boldsymbol{w}\|}$. Maximizing margin is an equivalent problem to minimizing $\frac{1}{2}\|\boldsymbol{w}\|^2$ subject to $y_i[\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle + b] \geq 1$, $i = 1, 2, \ldots, l$.

Thus, we can use the Quadratic Programming (QP) method to find the formulas for the $\boldsymbol{w}$ and b. Primal form of the Lagrangian is

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{l} \alpha_i [y_i(\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle + b) - 1],$$

where $\alpha_i \geq 0 \ \forall i$. We need to minimize the Lagrangian with respect to $\boldsymbol{w}$ and b and maximize it with subject to Lagrange multipliers α_i. By computing the Karush-Kühn-Tucker (KKT) conditions (see [1,3]) we obtain

$$\boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x}_i \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0.$$

Sometimes the primal formulation can be hard to solve, so it is better to represent the problem in dual formulation. Moreover, from the dual presentation we can solve the α_i's. Hence

$$L_D = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \tag{1}$$

subject to $\alpha_i \geq 0 \ \forall i$ and $\sum_{i=1}^{l} \alpha_i y_i = 0$. The Lagrangian of L_D is maximized with respect to α_i's. Because the support vectors have positive α_i's and the rest are equal to 0, the optimal solution of $\boldsymbol{w}$ can now be presented as

$$\boldsymbol{w} = \sum_{i \in SV} \alpha_i y_i \boldsymbol{x}_i, \tag{2}$$

where SV is the set of indices of the support vectors. The bias term can be easily evaluated by choosing an arbitrary $i \in SV$ and from the KKT condition $\alpha_i[y_i(\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle + b) - 1] = 0$ we obtain a formula for the b. That is, $b = y_i - \langle \boldsymbol{x}_i, \boldsymbol{w}\rangle$. A numerically safer option is to take the mean of all possible values of b [1]. A new example $\boldsymbol{x}$ can be now classified according to the sign of the decision function

$$f(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{w}\rangle + b. \tag{3}$$

If the sign is positive, an example gets the class label of 1 and otherwise it obtains the label -1.

2.2 Linearly Non-separable Case

In the real world the datasets of applications are rarely linearly separable, when the finding of hyperplane is easy. For this reason, we need the tools to handle also the linearly non-separable datasets. The first extension was the invention of *slack variables* [4] $\xi_i \geq 0$, $i = 1, 2, \ldots, l$, where each ξ_i measures the penalty of misclassification of the training point $\boldsymbol{x}_i$. Hence, we can represent our data by the inequalities

$$y_i[\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle + b] \geq 1 - \xi_i,$$

$i = 1, 2, \ldots, l$. Now, the objective function to be minimized has the form:

$$\tfrac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l} \xi_i.$$

Thus,

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i[y_i(\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle + b) - 1 + \xi_i] - \sum_{i=1}^{l} \mu_i \xi_i,$$

where $\alpha_i, \mu_i, \xi_i \geq 0$, $i = 1, 2, \ldots, l$. Again, L_P is minimized with respect to variables $\boldsymbol{w}, b$ and maximized subject to α_i, μ_i, $i = 1, 2, \ldots, l$. Parameter C, also called box constraint, is a trade-off parameter, which controls the influence of outliers in the data and controls the trade-off between maximum margin and minimum classification error [5]. Furthermore, the dual presentation is otherwise same than in (1), but the constraints are now $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x}_i = 0$. So, the difference between the presentations of L_D in the linearly separable and linearly non-separable cases is that in the latter one α_i's are bounded above with constant C. As in the linearly separable case, the solution for $\boldsymbol{w}$ equals (2) in the linearly non-separable case. Bias can be calculated by choosing a support vector satisfying $0 < \alpha_i < C$ and noticing the KKT condition as before [1]. A new example $\boldsymbol{x}$ is classified according to the equation (3).

2.3 Nonlinear Support Vector Machines

The use of kernel functions [1,3,4] was a groundbreaking innovation that extended the opportunities of SVMs widely. The basic idea is to map the feature vector from an input space to a higher dimensional feature space by a nonlinear transformation ϕ. Hence, the kernel function evaluates the inner product between training points and classifying a new example, it counts inner products between training points and test example.

Definition 1. *Kernel function is a symmetric function such that*

$$K(\boldsymbol{x}, \boldsymbol{z}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle \; \forall \boldsymbol{x}, \boldsymbol{z} \in X,$$

where ϕ is a nonlinear mapping from an input space to a feature space.

Analogously, as in the previous cases, we get

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i[y_i(\langle\phi(\boldsymbol{x}_i),\boldsymbol{w}\rangle + b) - 1 + \xi_i] - \sum_{i=1}^{l}\mu_i\xi_i,$$

and

$$L_D = \sum_{i=1}^{l}\alpha_i - \tfrac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i,\boldsymbol{x}_j).$$

Now, equation (2) has the form

$$\boldsymbol{w} = \sum_{i\in SV} y_i\alpha_i\phi(\boldsymbol{x}_i).$$

and $b = y_i - \langle\phi(\boldsymbol{x}_i),\boldsymbol{w}\rangle$ or we can take the mean of all possible values of b. A new example $\boldsymbol{x}$ can be classified according to the sign of the decision function

$$f(\boldsymbol{x}) = \langle\phi(\boldsymbol{x}),\boldsymbol{w}\rangle + b = \sum_{i\in SV}\alpha_i y_i K(\boldsymbol{x},\boldsymbol{x}_i) + b.$$

We have some restrictions for the use of kernel functions. Necessary and sufficient conditions for the valid kernel functions are presented in Mercer's theorem [1,3,19]. There are a few commonly used kernels also used in this work. These are:

(i) Linear (or dot product) $\langle\boldsymbol{x},\boldsymbol{z}\rangle$,
(ii) Polynomial $(1+\langle\boldsymbol{x},\boldsymbol{z}\rangle)^d$ where $d\in\{2,3,4,5\}$ is the order of the kernel,
(iii) Multi-Layer Perceptron (MLP) $\tanh(\kappa\langle\boldsymbol{x},\boldsymbol{z}\rangle + \delta)$ for parameters $\kappa > 0$ and $\delta < 0$,
(iv) Radial Basis Function (RBF) $e^{-\frac{\|\boldsymbol{x}-\boldsymbol{z}\|^2}{2\sigma^2}}$ where $\sigma > 0$.

We have a few commonly used multi-class extensions for the SVMs. Two of these are one-vs-all and one-vs-one methods [8]. In the following we will refer to them with the abbreviations OVA and OVO.

2.4 One-vs-All

Suppose that we have a problem of M classes and $M > 2$. In OVA we train M individual binary classifiers, where each one of them separates one class from the rest. Every classifier is trained with a full training set, meaning that the training set from each class is needed to separate an individual class from the rest of the classes. A test sample is classified into a specific class according to the results of the binary classifier. In an SVM classifier the output of a binary classifier is either 1 or -1. Number 1 indicates that the test sample belongs to the class k and correspondingly -1 indicates that the test sample belongs to the rest of $M-1$ classes, i.e to the class set $\{1,2,\ldots,k-1,k+1,\ldots M\}$.

Now, the desired situation in classifying the test sample with the OVA method is that one binary classifier gives output 1 and the other classifiers give -1. Hence, the

class with output 1 is chosen. Unfortunately, we encounter sometimes tie situations. These situations can be divided into two categories. Firstly, we have two or more binary classifiers with output 1 and secondly, every classifier gives -1. Tie situations are difficult to handle and they need special attention. In this work we have used the k-nearest-neighbor method, with $k = 1$.

The OVA method is usually more time-consuming than OVO, although the number of classifiers is significantly smaller, but the size of the training set for every classifier is larger than in OVO. Rifkin and Klautau [15] argued that OVA is as accurate as any other multi-class method. They assumed that the used binary classifiers are well-tuned regularized classifiers such as SVMs. A comparison between OVA and OVO will be considered in Section 3.

2.5 One-vs-One

In OVO we first train $\frac{M(M-1)}{2}$ binary classifiers, where every pair (k, m) is such that $k < m$ and $k, m \in \{1, 2, \ldots, M\}$. Although the number of binary classifiers is greater than in the OVA method, training process is faster, because the sizes of the training sets are much smaller for every binary classifier than in OVA. For a new test example, each classifier gives a vote to which class the test sample should belong. The final class will be evaluated with the majority voting method, where the class gained the most votes will be chosen. The majority voting method is simple to construct and it gives very good results in many cases [7]. Tie situations can happen also in OVO and they can be handled with different ways, but in this work we have used the k-nearest-neighbor method, with $k = 1$. Any other alternative than $k = 1$ could produce a new tie situation, when we are dealing with multi-class problems.

3 Experimental Tests

3.1 Data Description and Test Arrangements

Image dataset has altogether 1350 images from eight different taxonomical groups of benthic macroinvertebrates. The Latin-based names of the classes are: *Baetis rhodani, Diura nanseni, Heptagenia sulphurea, Hydropsyche pellucidulla, Hydropsyche siltalai, Isoperla* sp., *Rhyacophila nubila* and *Taeniopteryx nebulosa*. In the Figure 1 there is a sample image from all groups. Group sizes differ from each other and they are 116, 129, 172, 102, 271, 311, 83 and 166. In Tables 2-7 and in the following text we will refer to the groups with the capital letters A-H.

Data has been collected and classified first by the taxonomic experts [20]. After that collected benthic macroinvertebrate samples were scanned by a flatbed scanner (HP Deskjet 4850) and saved in the JPG format. In the preprocessing stage the images were segmented and normalized [20] and outliers were removed. Lastly, the features were extracted and calculated with a public java-based ImageJ program [6]. The data has 26 attributes altogether, where 25 of them were features and the last attribute was identification number. Kiranyaz et al. [9,10,11] and Ärje [20] selected 15 features from all possible and they divided the chosen features into two categories: geometrical and

Fig. 1. Image from each taxonomic group

statistical. The statistical features were {Mean, Standard deviation, Mode, Median, Integrated Density, Kurtosis, Skewness}, where Integrated Density means the sum of the values of the pixels in the image or selection. Moreover, the geometrical features were {Area, Perimeter, Width, Height, Feret's Diameter, Major, Minor, Circularity}, where Feret's Diameter is the longest distance between any two points along the selection boundary. Width and height are the measures of the smallest rectangle enclosing the selection in the image. Major and minor are the axes of the best fitting ellipse. Other 10 features not used in this work are {Min, Max, X, Y, XM, YM, BX, BY, Angle, Area Fraction}. Features {X, Y, XM, YM, BX, BY} concern the coordinates of the smallest enclosing rectangle and the center point of the selection as well as the first order spatial moments [6]. Features Min and Max are the minimum and maximum gray values within the selection [6]. Angle is the angle between the primary axis and a line parallel to the x-axis in the image [6].

We used three different feature sets and every feature set was tested with seven kernel functions, presented in the previous section. Firstly, we used only the aforementioned statistical features. Secondly, the geometrical features were used. Thirdly, we chose a random eight feature subset from the union of geometrical and statistical features. In this way we got the following feature set: {Area, Mean, Width, Height, Minor, Integrated Density, Kurtosis, Skewness}. Detailed information about each feature can be found from [6]. Experimental tests showed in [7] that the union of geometrical and statistical features is a good choice for classifying the macroinvertebrate images. Thus, in this paper we wanted to use alone statistical, geometrical and a randomly chosen eight feature subset.

We performed the classification process 250 times for each parameter combination of the kernel functions. Because in every round the training and test sets were chosen randomly, the classification results differ for every round. That is why we calculated the mean of the 250 round's classification results as a final result and the final result table was presented in percentages. From Table 1 we see the parameter values of the kernel functions, and the last column indicates the k-value used in tie situations. The common parameter for all kernel functions is the trade-off parameter bc, also known as the

box constraint value. Other more specific kernel parameters are σ, κ and δ. Parameters bc, σ, κ have the same parameter space. An exception from the previous parameters is $\delta < 0$ in MLP. We made an agreement of $\kappa = -\delta$ due to the computational reasons, because, otherwise, the tested parameter combinations would have increased significantly. With these assumption, we tested linear and polynomial kernel functions (degrees of $2, 3, 4$ and 5) with 40 parameter combinations and RBF and MLP with 1600 parameter combinations. We have used the binary SVM implementation of Bioinformatics Toolbox of Matlab as a basis for our tests. All the implementations and tests were made with Matlab. Moreover, we have used the Least Square method [16] in finding the optimal hyperplane. The data was divided into training and test sets such that 90% of the data was selected to be a training set and the rest of the data was left for its test set. In each round the members of the training and test sets were chosen randomly. Because the class sizes vary from 83 to 311, the random selection of a training set within the classes was made according to the sizes of the classes.

Table 1. Parameter values of kernel functions

Kernel	Box constraint	Kernel parameters			
	bc	κ	δ	σ	k
Linear	$0.5, 1.0, ..., 20$	–	–	–	1
MLP	$0.5, 1.0, ..., 20$	$0.5, 1.0, ..., 20$	$-20, -19.5, ..., -0.5$	–	1
Pol. ($d = 2$)	$0.5, 1.0, ..., 20$	–	–	–	1
Pol. ($d = 3$)	$0.5, 1.0, ..., 20$	–	–	–	1
Pol. ($d = 4$)	$0.5, 1.0, ..., 20$	–	–	–	1
Pol. ($d = 5$)	$0.5, 1.0, ..., 20$	–	–	–	1
RBF	$0.5, 1.0, ..., 20$	–	–	$0.5, 1.0, ..., 20$	1

How is the best parameter combination chosen? Usually, we do not find one parameter combination, which would produce the best classification result for every class. Hence, we applied the following procedure. We present it in general form. Suppose that we have a M class problem. Moreover, we have made the tests with N different parameter combinations. Now, we have N result tables of size $M \times M$. Each row of the result table indicates the total distribution in percentages, how the members of the specific class has been divided within all classes. Hence, the diagonal members are those, which tell us, how many members in percentage from a specific class has been classified correctly. Let $D_{ii,j}$ be the ith diagonal member from the jth result table, when $i \in \{1, 2, \ldots, M\}$ and $j \in \{1, 2, \ldots, N\}$. Furthermore, let us index the parameter combinations such that I_j, $j \in \{1, 2, \ldots, N\}$, is the jth used parameter combination. Now, we form a table T of a size $N \times 2M$, where we sort, alongwith each column, all the members $D_{ii,j}$, $i \in \{1, 2, \ldots, M\}$ and $j \in \{1, 2, \ldots, N\}$, together with corresponding I_j into decreasing order. Thus, we choose such index that, which has the most occurrences in the first row of T. That index corresponds to the best parameter combination and it defines the best result table which will be chosen. If we have a tie situation, we take the occurrences of the tied indices and their corresponding classification rates that are not in the first row of T. Thus, we choose such index (among the tied indices) that has more topmost classification rates than the other indices, when the occurrences of the first row of T is excluded.

Example 1. Let us consider a three class classification problem, where we have tested RBF kernel function with three parameter combinations. Let $I_1 = (1, 1.5), I_2 = (2, 2.5)$ and $I_3 = (3, 3.5)$, where the first parameter is the box constraint value and the second one is a value of σ. Let $D_{11,1} = 50.0, D_{22,1} = 70.0, D_{33,1} = 80.0, D_{11,2} = 30.0, D_{22,2} = 40.0, D_{22,3} = 20.0, D_{11,3} = 40.0, D_{22,3} = 90.0$ and $D_{33,3} = 90.0$. Now, the table T is

Index	Class 1	Index	Class 2	Index	Class 3
1	50.0	3	90.0	3	90.0
3	40.0	1	70.0	1	80.0
2	30.0	2	40.0	2	20.0.

Hence, from the first row of T we can choose index 3, because it has the most occurrences. Thus, the best parameter combination would be $(3, 3.5)$.

In Tables 2-7 we have compressed the results such that we have presented only the diagonal elements of the result tables, i.e., the percentage of the correctly classified test samples from each class and the results have been taken from the best parameter combination of the kernel function (optimal parameter values are in the parenthesis). In Tables and in the following text the term accuracy is used to mean classification rate. Moreover, in Table 8 we have collected the general weighted mean classification scores from each kernel function with every used feature set and multi-class method case for facilitating the analyzation of the results.

3.2 Results

Let us first consider the results of the geometrical features. From Table 2 we see that class A includes the lowest identification percentage of all classes. Classes B and D are quite well classified in most kernel function cases. Other classes have been identified with less than 80% accuracy (one exception in RBF case) and the classes F, G and H have, with one exception, under 70% results. Comparing to the corresponding results in Table 3, in OVO method, we have significant changes. Class A, which had the worst accuracy in OVA, has now improved accuracy in each kernel function case. Improvement is nearly 40% at its best. At the same time class F has a lower precision than in OVA, when the fourth degree polynomial kernel excluded. Otherwise, any clear general trends between OVA and OVO cannot be said, since both methods obtained either better or worse results. Complete results of OVA and OVO with the geometrical features can bee seen from Tables 2 and 3.

Tables 4 and 5 show the results when randomly selected eight features were used. The increment in a general accuracy can clearly be seen from the tables, since there are noticeably over 90% accuracies in many classes and kernel function cases. In OVA classes B and E consist of in most kernel cases at least 90% accuracy and the rest of classes have often over 80% results. This is a clear improvement in comparison with the geometrical feature results. In Table 5 we find some classes, which had below 90% in OVA, having now obtained over 90% accuracy. In addition, we do not find as many below 70% results from the classes as in OVA. The 5th degree polynomial kernel function and MLP are exceptions.

Table 2. Average accuracies in percentages: One-vs-all method with different kernel functions and geometrical features

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear ($bc = 1.0$)	35.8	94.2	67.1	82.6	79.0	62.5	56.6	63.3
Pol. ($d = 2$, $bc = 20.0$)	33.5	93.1	74.8	90.6	82.5	60.0	59.6	67.0
Pol. ($d = 3$, $bc = 5.0$)	34.5	95.3	78.9	89.2	86.0	59.0	65.0	67.6
Pol. ($d = 4$, $bc = 1.0$)	33.6	92.1	78.0	86.1	87.5	60.0	66.3	68.4
Pol. ($d = 5$, $bc = 0.5$)	32.9	76.4	74.4	74.6	86.6	61.4	65.7	66.6
RBF ($\sigma = 0.5$, $bc = 20.0$)	42.9	97.4	85.7	92.6	88.7	68.2	74.4	65.7
MLP ($\kappa = 0.5$, $bc = 0.5$)	32.1	82.6	50.3	71.4	76.7	58.8	41.2	50.0

Table 3. Average accuracies in percentages: One-vs-one method with kernel functions and geometrical features

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear ($bc = 13.5$)	73.5	95.1	70.9	76.8	80.8	37.5	63.2	76.5
Pol. ($d = 2$, $bc = 19.5$)	67.9	95.7	80.1	90.1	79.2	43.9	68.1	79.8
Pol. ($d = 3$, $bc = 2.0$)	66.7	96.9	85.2	91.6	83.7	52.9	64.4	72.2
Pol. ($d = 4$, $bc = 0.5$)	62.5	97.0	72.5	89.4	85.6	61.1	64.2	65.9
Pol. ($d = 5$, $bc = 0.5$)	53.6	93.2	65.5	78.0	82.8	60.4	58.8	59.5
RBF ($\sigma = 1.0$, $bc = 15.0$)	65.7	97.9	88.2	92.2	88.2	64.3	76.5	64.3
MLP ($\kappa = 0.5$, $bc = 0.5$)	38.0	82.2	41.3	57.4	51.2	45.3	29.4	47.5

Table 4. Average accuracies in percentages: One-vs-all method with kernel functions and randomly selected features

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear ($bc = 11.5$)	88.3	96.7	70.5	81.3	88.7	76.8	72.8	62.1
Pol. ($d = 2$, $bc = 11.5$)	89.1	97.2	82.6	84.6	90.5	83.8	77.6	70.6
Pol. ($d = 3$, $bc = 20.0$)	92.1	97.1	82.6	93.0	91.8	88.3	84.8	74.1
Pol. ($d = 4$, $bc = 0.5$)	90.4	94.6	81.6	91.5	92.0	88.0	86.3	75.3
Pol. ($d = 5$, $bc = 0.5$)	84.1	74.6	65.5	78.9	90.8	85.5	83.3	76.1
RBF ($\sigma = 1.0$,$bc = 19.0$)	91.5	96.3	89.3	96.8	95.1	85.6	87.0	78.2
MLP ($\kappa = 0.5$, $bc = 1.0$)	53.2	60.6	39.8	57.9	52.9	52.2	33.5	38.0

Table 5. Average accuracies in percentages: One-vs-one method with kernel functions and randomly selected features

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear ($bc = 1.0$)	94.3	97.6	74.8	93.1	92.2	82.3	87.7	84.8
Pol. ($d = 2$, $bc = 20.0$)	95.4	98.5	90.1	96.8	90.8	81.7	89.6	88.1
Pol. ($d = 3$, $bc = 1.0$)	92.1	97.5	86.5	92.9	93.7	83.5	90.7	81.2
Pol. ($d = 4$, $bc = 0.5$)	86.6	95.7	75.1	90.2	90.6	85.5	85.0	72.5
Pol. ($d = 5$, $bc = 0.5$)	79.1	77.9	58.4	79.8	84.6	75.7	60.1	65.2
RBF ($\sigma = 1.50$, $bc = 17.0$)	90.9	97.3	90.9	94.8	95.3	85.9	93.4	80.3
MLP ($\kappa = 0.5$, $bc = 2.0$)	64.8	71.7	40.9	66.8	43.7	44.3	41.5	45.7

In the third instance, for the statistical feature set, we obtained interesting results. Although the number of the features is one smaller than in the previous sets, the results are still good. Class D had, for example, in Table 6 almost perfect 100% accuracy, for the RBF kernel function. Moreover, RBF achieved also over 90% accuracies in other four classes. A large part of the kernel functions gained over 80% results, which is a good result in such a difficult classification problem. The linear kernel function succeeded remarkably better in the statistical features in the one-vs-one method than in one-vs-all. The difference between these two cases was over 30% being a clear improvement. Besides the linear kernel, a large part of the other kernel functions in OVO obtained better results than in OVA. With this in mind we can say that statistical features together with the one-vs-one method is a good alternative for classifying the benthic macroinvertebate images.

Table 6. Average accuracies in percentages: One-vs-all method with kernel functions and statistical features

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear ($bc = 12.5$)	75.5	85.2	57.2	58.7	74.9	85.1	60.0	69.6
Pol. ($d = 2$, $bc = 8.0$)	83.5	87.9	73.9	87.8	76.5	87.1	72.8	72.1
Pol. ($d = 3$, $bc = 12.0$)	87.1	91.4	81.3	93.9	80.5	90.2	89.5	77.0
Pol. ($d = 4$, $bc = 0.5$)	86.9	89.9	82.3	95.6	88.5	92.7	90.6	85.7
Pol. ($d = 5$, $bc = 0.5$)	85.7	90.5	82.5	87.6	88.5	94.3	90.0	89.9
RBF ($\sigma = 1.0$, $bc = 17.5$)	91.4	95.6	83.6	99.1	87.2	94.7	94.6	88.3
MLP ($\kappa = 0.5$, $bc = 0.5$)	42.2	52.2	43.0	53.4	46.0	56.8	31.9	41.8

Table 7. Average accuracies in percentages: One-vs-one method with kernel functions and statistical features

	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear ($bc = 1.0$)	95.2	97.0	85.3	92.4	83.7	95.6	89.4	93.5
Pol. ($d = 2$, $bc = 13.5$)	94.6	98.4	89.1	95.8	88.1	96.0	93.4	97.8
Pol. ($d = 3$, $bc = 0.5$)	94.2	98.9	88.9	97.3	89.6	96.0	95.3	93.4
Pol. ($d = 4$, $bc = 1.0$)	93.0	94.3	85.7	96.9	87.6	93.9	96.1	91.4
Pol. ($d = 5$, $bc = 0.5$)	88.9	89.6	79.4	94.8	86.7	94.6	91.0	84.0
RBF ($\sigma = 2.5$, $bc = 8.0$)	94.3	99.3	89.3	96.4	87.5	95.9	97.6	96.2
MLP ($\kappa = 0.5$, $bc = 12.5$)	75.7	73.6	56.0	81.6	48.8	73.1	76.0	60.5

Table 8, where the accuracies have been calculated according to the sizes of the classes, affirms the aforementioned observations of the preceding result tables. The geometric features are clearly the worst alternative in OVA and OVO methods, but the evenness of average accuracies is interesting between the kernel function results in OVA and OVO. All kernels, except MLP, are very close to each other. Alone in OVA or OVO, the differences between the kernels are quite great. When the random features were used, the linear and the 5th degree polynomial kernel functions had a considerable gap in the average results for the OVA and OVO methods. These cases had a nearly 8% difference, when other kernel functions had quite even results. Alone in OVA the best result was achieved with RBF as well as in OVO. The greatest general improvent

in results was in the statistical features. Here the linear kernel function and MLP had almost an 18% increasement and the other kernels, except the 5th degree of the polynomial kernel function, made also betterments, but smaller than with the two mentioned kernel functions. Otherwise, RBF gained the best results in OVA, and quadratic, cubic and RBF obtained the same 93.7% percentage result.

Table 8. Weighted means of the accuracies

	One-vs-all			One-vs-one		
Kernel	Geom.	Stat.	Random	Geom.	Stat.	Random
Linear	68.3	73.3	79.6	68.5	91.1	87.3
Pol. $d = 2$	70.3	80.4	84.8	72.0	93.7	89.8
Pol. $d = 3$	71.9	85.5	87.8	74.6	93.7	88.9
Pol. $d = 4$	71.9	89.0	87.4	73.9	91.6	85.0
Pol. $d = 5$	68.9	89.2	81.1	69.2	88.6	73.9
RBF	76.9	91.1	89.5	78.2	93.7	90.4
MLP	60.0	47.5	49.2	49.1	65.6	49.9

Tie situation analysis is an interesting topic. We gathered statistical information about the numbers of tie situations, when OVA and OVO were used, and we also see the effect of a kernel function on tie situations. All information is found from the Tables 9 and 10. Table 9 shows that the average number of ties is very low in many kernel function cases, but the selection of features has a great influence on the amount of ties. The linear, quadratic, cubic and RBF kernels had only few tie situations in each feature set case. The polynomial kernels (degrees of 4 and 5) had a clear difference between the statistical features and the rest feature sets. MLP had the topmost number of ties with every feature set. When comparing the number of ties and the average classification results from Table 9, we see that these two have a common factor. If the number of ties is high, the results are usually worse than otherwise.

In OVA the number of ties has spread to a wider interval than in OVO. For instance, in the column of the statistical features, the minimum is 17.2 and the maximum is 123.8. The higher degrees of the polynomial kernel functions have, in every feature set case, smaller numbers of ties than the linear and quadratic kernels. This is an opposite situation compared to OVO. RBF still had the lowest frequency in ties, but, when the

Table 9. The numbers of ties when using a one-vs-one method and different kernel functions

	Geometrical		Statistical		Random	
Kernel	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
Linear	2.9	1.6	0.5	0.7	1.2	1.1
Pol. $d = 2$	2.6	1.7	0.4	0.6	1.2	1.1
Pol. $d = 3$	6.3	2.5	0.6	0.8	3.8	1.8
Pol. $d = 4$	11.9	3.3	2.9	1.6	12.2	3.3
Pol. $d = 5$	20.4	4.3	8.0	2.9	23.6	4.5
RBF	1.8	1.3	0.1	0.3	0.8	0.9
MLP	44.9	5.7	36.5	6.1	47.0	5.9

Table 10. The numbers of ties when using a one-vs-all method and different kernel functions

	Geometrical		Statistical		Random	
Kernel	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
Linear	110.6	3.9	109.7	4.0	94.9	4.5
Pol. $d = 2$	94.7	4.3	78.3	5.2	57.3	4.6
Pol. $d = 3$	83.5	5.0	42.4	5.0	38.0	4.9
Pol. $d = 4$	81.0	5.3	27.0	4.2	35.4	4.7
Pol. $d = 5$	82.0	5.2	23.3	4.1	48.1	4.7
RBF	38.4	5.1	17.2	3.7	18.8	3.6
MLP	123.4	4.7	123.8	4.2	125.9	3.6

differences are so great, tie analysis can have a really enormous influence on the results. Furthermore, the method also has a great effect, and from this point of view OVO is preferable to use instead of OVA. The difference between numbers of tie situations in OVA and OVO can be explained with nature of the methods. Because OVO has, in this application, 28 classifiers giving the votes, the ties are not so frequent. In OVA we have only 8 classifiers and it gives a greater probability to induce ties more frequent.

4 Discussion

In this paper, we examined the SVM multi-class extensions of one-vs-one and one-vs-all, and their applicability to classify the benthic macroinvertebrate images. We performed wide tests with three feature sets, and each one of them was tested with seven different kernel functions. Altogether we made 42 test set-ups together with wide parameter spaces of the kernel functions. Our experiments showed that, although the classification of benthic macroinvertebrate is a difficult problem, one-vs-one method with statistical features obtained highly promising results. More specifically, in this feature set the quadratic, cubic and RBF kernel functions obtained over 93% accuracies for weighted mean results. This is a very good result, when the number of the features is below one third of all possible features.

In the one-vs-all method, the statistical and randomly selected feature sets were quite even, but with the geometrical feature set, the average identification percentage was lower. Nevertheless, the best results with the statistical features were still behind the one-vs-one results. From the kernel functions Gaussian RBF had the best mean accuracies in every feature set. The same detail also occurred for the one-vs-one method. Overall, SVM is a very promising technique in classifying macroinvertebrate images and these results give a good basis for further research and development. The next stage is to solve how other classification methods, such as k-nearest-neighbor, decision trees, naïve Bayes, linear discriminant analysis, will succeed in this classification problem.

Acknowledgments. We thank Finnish Environment Institute, Jyväskylä, Finland for the data. The first author is also thankful to the Tampere Graduate Program in Information Science and Engineering for the support.

References

1. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 121–167 (1998)
2. Chauduri, A., De, K., Chatterjee, D.: A Comparative Study of Kernels for the Multi-class Support Vector Machine. In: Guo, M., Zhao, L., Wang, L. (eds.) Fourth International Conference of Natural Computation (ICNC 2008), vol. 2, pp. 3–7 (2008)
3. Christiani, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press, Cambridge (2000)
4. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20, 273–297 (1995)
5. Debnath, R., Takahide, N., Takahashi, H.: A decision based one-against-one method for multi-class support vector machine. Pattern Analysis Applications 7, 164–175 (2004)
6. ImageJ: public domain Java-based image processing program, `http://rsbweb.nih.gov/ij/docs/index.html`
7. Joutsijoki, H., Juhola, M.: Kernel selection and its consequense to the number of ties in majority voting method (2011) (submitted)
8. Kahsay, L., Schwenker, F., Palm, G.: Comparison of multiclass svm decomposition schemes for visual object recognition. In: Kropatsch, W., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, pp. 334–341. Springer, Heidelberg (2005)
9. Kiranyaz, S., Gabbouj, M., Pulkkinen, J., Ince, T., Meissner, K.: Classification and Retrieval on Macroinvertebrate Image Databases using Evolutionary RBF Neural Networks. In: Proceedings of the International Workshop on Advanced Image Technology, Kuala Lumpur, Malaysia (2010)
10. Kiranyaz, S., Ince, T., Pulkkinen, J., Gabbouj, M., Ärje, J., Kärkkäinen, S., Tirronen, V., Juhola, M., Turpeinen, T., Meissner, K.: Classification and retrieval on macroinvertabrate image databases. Computers in Biology and Medicine 41, 463–472 (2011)
11. Kiranyaz, S., Gabbouj, M., Pulkkinen, J., Ince, T., Meissner, K.: Network of evolutionary binary classifiers for classification and retrieval in macroinvertebrate databases. In: Proceedings of 2010 IEEE 17th International Conference on Image Processing (ICIP), pp. 2257–2260 (2010)
12. Li, Y., Dorai, C.: Instructional Video Content Analysis Using Audio Information. IEEE Transactions on Audio, Speech, and Language Processing 14(6), 2264–2274 (2006)
13. Liu, J., Xie, L.: SVM-Based Automatic Classification of Musical Instruments. In: International Conference on Intelligent Computation Technology and Automation (ICICTA 2010), vol. 3, pp. 669–673 (2010)
14. Lytle, D.A., Martinez-Muñoz, G., Zhang, W., Larios, N., Shapiro, L., Paasch, R., Moldenke, A., Mortensen, E.N., Todorovic, S., Dietterich, T.G.: Automated processing and identification of benthic invertebrate samples. Journal of North American Benthological Society 29(3), 867–874 (2010)
15. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. Journal of Machine Learning 5, 101–141 (2004)
16. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. Neural Processing Letters 9, 293–300 (1999)
17. Thorsten, J.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)

18. Tirronen, V., Caponio, A., Haanpää, T., Meissner, K.: Multiple order gradient feature for macroinvertebrate identification using support vector machines. In: Kolehmainen, M., Toivanen, P., Beliczynski, B. (eds.) ICANNGA 2009. LNCS, vol. 5495, pp. 489–497. Springer, Heidelberg (2009)
19. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (2000)
20. Ärje, J., Kärkkäinen, S., Meissner, K., Turpeinen, T.: Statistical classification and proportion estimation - an application to macroinvertebrate image database. In: IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010), Kittilä, Finland, pp. 373–378 (2010)

Publication III

Half-Against-Half Multi-Class Support Vector Machines in Classification of Benthic Macroinvertebrate Images

Henry Joutsijoki

Available at:
`http://dx.doi.org/10.1109/ICCISci.2012.6297281`

Half-Against-Half Multi-Class Support Vector Machines in Classification of Benthic Macroinvertebrate Images

Henry Joutsijoki
School of Information Sciences
University of Tampere
Kanslerinrinne 1, FI-33014 Tampere, Finland
Email: henry.joutsijoki@uta.fi

***Abstract*—In this paper we investigated how Half-Against-Half Support Vector Machine (HAH-SVM) succeed in the classification of the benthic macroinvertebrate images. Automated taxa identification of benthic macroinvertebrates is a slightly researched area and in this paper HAH-SVM was for the first time applied to this application area. The main problem in HAH-SVM is to find the right way to divide the classes in a node. We solved the problem by using two different approaches. Firstly, we applied the Scatter method which is a novel approach for the class division problem. Secondly, we formed the class divisions in an HAH-SVM by a random choice. We performed extensive experimental tests with four different feature sets and tested every feature set with seven different kernel functions. The tests showed that by the Scatter method and random choice formed HAH-SVMs performed from classification problem very well obtaining over 95% accuracy. Moreover, the 7D and 15D feature sets together with the RBF kernel function are good choices for this classification task. Generally speaking, HAH-SVM is a promising strategy for automated benthic macroinvertebrate identification.**

I. Introduction

The growing interest towards biological issues in the past decades has increased our knowledge about the nature and the environment surrounding us. The interest is still expanding and the exponentially growing amount of information brings us continuosly more challenges to develop computationally better and more reliable tools for analyzing the data. Due to the nature's diversity and, hence, its complexity general tools are difficult to invent.

An important part of the nature is freshwater areas. These are in a minority position when taking account into all aquatic environments in the Globe and, that is why, the constant biomonitoring is needed. Due to the environmental legislation, the need of biomonitoring has increased in the past decades [19]. Benthic macroinvertebrates are an essential part of the freshwater areas. They live on the bottom of the waterbodies and their life cycle is usually from 1-2 years [19]. Benthic macroinvertebrates react quickly to changes in water quality and the observed changes in them are good indicators for the situation of a freshwater area [20]. Chemical samples give a short snap-shot of the situation for the researchers, but for long-lasting biomonitoring benthic macroinvertebrates or more generally biological organisms are better indicators [19].

If the benthic macroinvertebrates are used in an extensive biomonitoring, this requires that the identification methods of the benthic macroinvertebrates are good and reliable. Generally, the automated taxa identification of benthic macroinvertebrates [4]–[6], [8]–[12], [14]–[16], [20] has received minor attention in the areas of pattern recognition and machine learning. Traditionally, the identification of benthic macroinvertebrate samples is made by the taxonomic experts. The biggest problem for using automated taxa identification techniques in real life is the the reluctance of taxonomic experts [19]. By automated taxa identification the costs of identification process could be decreased greatly and it would make the identification more effective. By this means more extensive biomonitoring could be made and the problems of the freshwater areas could be prevailed.

Support Vector Machines (SVMs) [1], [21] has been applied for the automated taxa identification of benthic macroinvertebrates with great success. One-vs-all and one-vs-one methods were examined in [4] and in [4], [5] special attention for the problem of the tie situations was given. Tie situation problem can be reflected in the benthic macroinvertebrate identification to a situation where the taxon of a new sample cannot be uniquely determined. In real world the tie situations are not rare at all, because often taxonomists encounter samples which cannot be unambiguously determined, when different taxonomists can have different opinions what species is in question. Directed Acyclic Graph Support Vector Machine (DAGSVM) was applied in [6] and it showed to be a good choice for the classification of benthic macroinvertebrates.

There are three main points in this paper. Firstly, we want to investigate how Half-Against-Half Support Vector Machines introduced by Lei and Govindaraju [13] contrive to classify the benthic macroinvertebrate samples at our disposal. Secondly, we examine which kernel function is the best one for this multi-class extension, whereas in [4]–[6] the one-vs-all, one-vs-one and DAGSVM multi-class extensions were investigated. Thirdly, we contemplate how the HAH-SVM built by the

978-1-4673-1938-6/12/$31.00 ©

Scatter method (algorithm described exactly in [7]) manages compared to a randomly builded HAH-SVM.

In Section II we describe shortly the theory of SVM [1], [2], [21] and we introduce the Half-Against-Half Support Vector Machines [13] method. Section III is devoted to the results and we also present the technical details how the experimental tests were arranged. The last section is left to discussion, conclusion and further research questions.

II. Methods

A. Support Vector Machine

Let us have a given training set $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_l, y_l)\}$ where $\mathbf{x}_i \in \mathbb{R}^n$ are the training examples and $y_i \in \{-1, 1\}$ are the corresponding class labels of $\mathbf{x}_i$, $i = 1, 2, \ldots, l$. The goal is to find a hyperplane $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ classes separating with maximum margin where $\mathbf{w}$ is a weight vector and $b \in \mathbb{R}$ is a threshold. The closest points to the hyperplane are called as support vectors and the margin has the value of $\frac{2}{\|\mathbf{w}\|}$ (see details [1], [4]). An optimal hyperplane can be found by solving the quadratic programming problem:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}} \tfrac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{l} \xi_i \tag{1}$$

subject to $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$ and $\xi_i \geq 0, i = 1, 2, \ldots, l$, where ξ_i's are the slack variables. The optimatization problem in (1) can be solved more easily in the dual form:

$$\max L(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{2}$$

subject to $\sum_{i=1}^{l} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$ where C is a user-defined parameter, which controls the trade-off between maximum margin and minimum classification error. New example $\mathbf{x}$ can be classified according to the sign of the decision function

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \tag{3}$$

where b can be evaluated, for instance, as a mean of all possible values of b (see details [4], [5]).

The use of kernel trick [1] enlarged the capabilities of SVM. The main point is to map the training data into higher dimensional space by a nonlinear transformation ϕ. In the feature space the data, which is linearly non-separable in the input space, can be separated with a hyperplane. Fortunately, with the help of the kernel function, $K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$, we can evaluate the separating hyperplane without doing the actual mapping to the higher dimensional space. Commonly used (also in this paper) kernel functions are: linear kernel function $\langle \mathbf{x}, \mathbf{z} \rangle$, polynomial kernel functions $(1 + \langle \mathbf{x}, \mathbf{z} \rangle)^d$ where $d \in \mathbb{N}$ is the order of the kernel function. Furthermore, there are Radial Basis Function (RBF) $\exp(-\|\mathbf{x} - \mathbf{z}\|^2 / 2\sigma^2)$ with $\sigma > 0$ and Sigmoid kernel function $\tanh(\kappa \langle \mathbf{x}, \mathbf{z} \rangle + \delta)$ with $\kappa > 0$ and $\delta < 0$. All valid kernel functions need to satisfy the conditions of Mercer's theorem [21]. Calculations for the optimal hyperplane with the kernel functions go analogously as in the linearly separable and linearly non-separable cases. Now, the decision function in (3) has the form

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i \langle \phi(\mathbf{x}), \phi(\mathbf{x}_i) \rangle + b \tag{4}$$

and a new example is classified according to the sign of $f(\mathbf{x})$.

B. Half-Against-Half Support Vector Machines

HAH-SVM is an interesting variation for extending SVM to also concern multi-class cases. It was introduced by Lei and Govindaraju [13]. HAH-SVM is a mixture of one-vs-one, one-vs-all and DAGSVM methods, since the complexity of the training phase is similar to one-vs-one [13] and the number of classifiers is $\mathrm{O}(K)$ as in the one-vs-all method and the structure has the same kind of exterior features as DAGSVM has. Every node in an HAH-SVM contains of a binary SVM classifier. The classification of a new example begins at the root node and according to the results of an SVM classifier, we move via right or left edge until we are in a leaf where the final class label of the test example can be found.

The greatest problem in HAH-SVM is to find the optimal way to divide the classes in each node. When the number of classes is small, we can use a prior knowledge about the classes or simply choose randomly the divisions. If the number of classes is high, we need to use some more sophisticated methods to solve optimal way to divide the classes. In this paper we have used the Scatter method [7] for finding the optimal class divisions in each node. We divided the classes into two groups according to the results of the Scatter method. To the first group was chosen to include such a half of the classes which had the best separability in the sense of the Scatter method. This procedure was repeated in every node. This approach is novel and the exact description of the Scatter method can be found from [7]. In Figure 1 there are the HAH-SVMs (Scatter and random division) which were used in this paper. More theoretical and experimental information about HAH-SVM can be found from [13].

III. Experimental Results

A. Data Description and Test Arrangements

Data is compounded of 1350 images from eight taxonomical groups of benthic macroinvertebrates: {*Baetis rhodani*, *Diura nanseni*, *Heptagenia sulphurea*, *Hydropsyche pellucidulla*, *Hydropsyche siltalai*, *Isoperla* sp., *Rhyacophila nubila* and *Taeniopteryx nebulosa*}. Group sizes alternate and are 116, 129, 172, 102, 271, 311, 83 and 166. We will refer to the groups with the capital letters A-H. Example images about benthic macroinvertebrates can be found, for example, from [4]–[6], [8]–[10], [19], [20].

We used in classification the usual training, validation, test sets techinique. We divided the data 100 times into training, validation and test sets such that 10% of the data was selected to be a test set and another 10% was a validation set and the

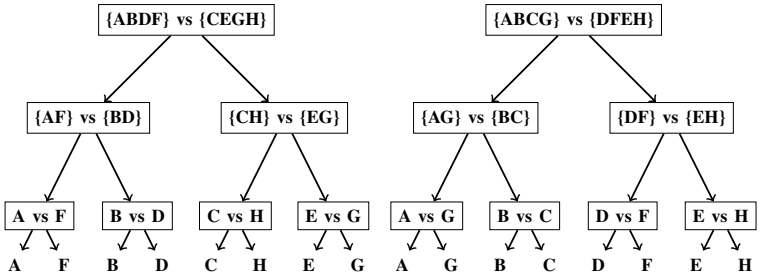


Figure 1: Scatter HAH-SVM on the left and random HAH-SVM on the right.

rest of the data, 80%, was left to the training set. Firstly, we trained the SVMs with the training data. Secondly, we evaluated the performance of a trained model with the validation set. Thirdly, we took the mean of the 100 accuracies obtained from the validation sets. Parameter values were selected for the final testing phase by selecting those parameter values which gained the best mean accuracies. When the parameter values were chosen, SVMs were trained again with the training data including the validation set.

We had four different parameters depending on the kernel function. All kernel functions have one common parameter, box constraint, and whose parameter space was $\{0.1, 0.2, \ldots, 10\}$. RBF has two parameters, σ and box constraint, and they had the same parameter space $\{0.1, 0.2, \ldots, 10\}$. Sigmoid kernel function has three parameters: box constraint, $\kappa > 0$ and $\delta < 0$. For the box constraint and κ parameter spaces were the same as before, but now $\delta \in \{-10.0, -9.9, \ldots, -0.1\}$. The obtained optimal parameter values can be seen in Table I.

We tested 100 parameter values in the case of polynomial kernel functions. Furthermore, the RBF and Sigmoid kernel functions were tested with 10000 parameter combinations and for Sigmoid kernel we made an agreement of $\kappa = -\delta$ because, otherwise, the number of parameter combinations would have increased from 10000 to 100^3. In feature selection we have used the knowledge from [4]–[6] and one feature set, 17D, was a mixture of 15D and 25D features. Three other feature sets were 7D, 15D and 25D. The data had altogether 25 features and the features can be divided into two categories: simple shape features and grey value features. Features were extracted from the images by using ImageJ [3]. The descriptions of 7D, 15D and 25D feature sets can be found from [4]–[6] and 17D feature set was {Area, Mean, Max, X, Y, XM, YM, Perimeter, BX, BY, Width, Height, Major, Circularity, Feret, Integrated Density, Median}. The definitions of all features can be found from [3] and more details about the preprocessing stage can be found from [20].

We performed our tests with Matlab and we used the SVM implementation of Bioinformatics Toolbox of Matlab. Furthermore, we used the Least Square [18] method in finding the optimal hyperplane. In the result tables we compressed the results so that every row indicates the mean of classwise classification rates (true positive rates) with a specific kernel function. Accuracies obtained from the kernel functions and their statistical comparison are presented in the last result table. Accuracies were compared statistically with the two sided Wilcoxon signed ranks test [17] when $p < 0.05$. In the result tables we boldfaced the best result of each column to make the analysis more easier for a reader.

B. Results

The smallest feature set 7D achieved very good results in [4], [6] when one-vs-one, one-vs-all and DAGSVM were used. The same tendency continued also for the present HAH-SVM case. Tables II and III show the exact results of 7D feature set and different kernel functions. A noticeable fact is that in every result table Sigmoid kernel function obtained the worst results. This may be a consequence from the preprocessing stage where the data was only standardized to have zero mean and unit variance and no transformations such as linear scaling or normalization were made. The results show that randomly formed and with the Scatter method formed HAH-SVMs obtained good results with 7D features. In both cases classes B, D, F, G and H were classified with classification rates over 90% which is an excellent achievement. The rest of the classes were classified very well also since they reached above 88% classification rates. From the kernel functions RBF was the best one in Tables II and III. Also, polynomial kernel functions from the cubic to the 5th degree of polynomial kernel function gained good results.

Table IV shows the results of 15D feature set. The general level of the classification rose from the 7D case a bit. Now, RBF kernel function achieved the best classification rate in seven classes from eight possible. Class H was the only class which was classified better with other than RBF kernel. Classes D and E were classified with 98% and 98.4% classification rates and all the rest of the classes were classified above 90% classification rate.

In Table V there is more deviation compared to the previous table. The topmost results were spread over the quadratic, cubic and RBF kernel functions. Overall, the highest classification rates classwise examined did not alternate very much between Scatter and random HAH-SVMs. A suprising aspect is that

Table I: The best kernel parameters obtained with the Scatter HAH-SVM and random HAH-SVM

	Scatter				Random			
Kernel	7D	15D	17D	25D	7D	15D	17D	25D
Linear	(1.7)	(7.0)	(9.3)	(9.9)	(2.5)	(9.2)	(5.9)	(8.5)
Pol. $deg = 2$	(9.5)	(2.1)	(4.9)	(1.7)	(9.6)	(1.7)	(6.2)	(0.9)
Pol. $deg = 3$	(1.3)	(0.1)	(0.3)	(0.1)	(1.9)	(0.1)	(0.2)	(0.1)
Pol. $deg = 4$	(0.5)	(0.1)	(0.1)	(0.1)	(0.4)	(0.1)	(0.1)	(0.1)
Pol. $deg = 5$	(0.1)	(0.1)	(0.1)	(0.1)	(0.1)	(0.1)	(0.1)	(0.3)
RBF	(9.9,1.0)	(10.0,1.7)	(9.9,2.2)	(9.6,2.8)	(9.7,1.0)	(10.0,1.6)	(10.0,2.4)	(9.9,3.2)
Sigmoid	(2.0,0.1,-0.1)	(0.4,0.1,-0.1)	(0.3,0.1,-0.1)	(0.1,0.1,-0.1)	(1.8,0.1,-0.1)	(0.2,0.1,-0.1)	(0.1,0.1,-0.1)	(0.1,0.1,-0.1)

Table II: Scatter HAH: Classification rates (%) with kernel functions and 7D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	41.2	71.8	54.2	84.2	76.9	67.9	0.0	27.6
Pol. $deg = 2$	76.3	78.1	87.3	83.2	**93.5**	87.8	78.6	60.4
Pol. $deg = 3$	87.9	89.2	87.0	92.3	92.3	95.6	87.0	85.7
Pol. $deg = 4$	88.4	89.4	87.2	93.0	91.2	**97.0**	92.4	91.8
Pol. $deg = 5$	85.7	86.5	83.3	92.5	91.4	95.6	92.6	**92.2**
RBF	**89.7**	**96.1**	**88.2**	**98.5**	91.4	96.5	**93.2**	90.2
Sigmoid	53.4	35.9	42.7	46.8	49.1	55.2	41.8	46.2

Table III: Random HAH: Classification rates (%) with kernel functions and 7D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	**93.3**	95.2	70.5	34.1	60.4	96.9	14.4	20.8
Pol. $deg = 2$	87.1	98.3	85.7	80.8	74.0	95.0	87.4	79.9
Pol. $deg = 3$	91.6	92.4	88.1	87.8	86.6	96.0	94.8	88.6
Pol. $deg = 4$	88.8	91.3	88.1	91.4	87.5	96.5	93.7	92.4
Pol. $deg = 5$	84.5	87.3	84.4	91.8	**89.0**	95.7	95.0	**93.2**
RBF	91.1	**98.7**	**88.8**	**98.5**	88.2	**97.0**	**96.2**	91.3
Sigmoid	78.8	71.7	52.0	54.3	63.7	59.1	61.6	31.9

the linear kernel function did not contrive well from the classification although in [4] it gained good results. The reason behind this may be in the structure of HAH-SVM. For instance, in the root node there were training examples from more than two classes to be separated with a single binary SVM, so the input space (or the feature space) is too complicated to be separated with linear kernel function. On the other hand, in one-vs-one method each SVM classifier has only training examples from no more than two classes, and, therefore the space is easier to separate using the linear kernel function.

In the 17D feature set case the level of classification decreased compared to the previous result tables. Classes C and H were the most difficult classes to recognize when the HAH-SVM was created by the Scatter method and in the case of a random choice class C was the only one which remained below 90% classification rate. In the Scatter case the topmost classification rates were spread over four kernel functions and in the random HAH-SVM the highest classwise classification rates were distributed within three kernel functions. The cubic kernel function and RBF were the best choices with the 17D feature set. More details about the 17D results can be found from Tables VI and VII.

In Tables VIII and IX there are the results given by 25D feature set. Now the level of classification is similar to the 7D results. In Table VIII the worst class to recognize was class H and the best one was class E. The topmost classification rates were yielded by the quadratic and RBF kernels. Random method results differ slightly from the results obtained by the Scatter method. By the random method every class had above 90% classification rates including class H which obtained below 90% classification rate in the previous case. Table IX shows that the highest classification rates were among quadratic, cubic and RBF kernel functions. A suprising detail is that the 25D features set did not manage to classify benthic macroinvertebrate images better than 7D feature set although it has over three times more features than in 7D feature set.

In the Table X we see an interesting result. When comparing the best accuracies of the Scatter and random columns in each feature set case, we see that the same kernel functions obtained the best performances. Moreover, the difference between the best accuracies using the Scatter method and random choice was below 1%. In every feature set case a random choice produced the better result, but from the practical point of view these results are equally good since their difference was so small. The best feature set choice would be 15D together with RBF kernel function. The 7D and 25D feature sets had quite similar results and the 17D feature set was the poorest one.

IV. Discussion

In this paper we applied Half-Against-Half multi-class SVMs in automated taxa identification of benthic macroinvertebrates. We formed two binary trees where the first one was created by using Scatter method [7] recursively in each node where was an SVM classifier. The other decision tree was created randomly to have a point of comparison for the decision tree builded by Scatter method. Experimental tests were extensive. We tested

Table IV: Scatter HAH: Classification rates (%) with kernel functions and 15D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	73.7	91.2	71.0	58.0	91.6	92.5	68.1	40.3
Pol. $deg = 2$	90.3	92.8	91.7	93.7	93.8	93.1	84.8	84.7
Pol. $deg = 3$	90.2	89.8	89.3	91.9	96.9	95.4	90.5	91.5
Pol. $deg = 4$	87.5	71.8	77.8	76.9	93.3	94.1	87.2	**92.4**
Pol. $deg = 5$	76.4	51.5	56.7	51.5	86.8	89.1	76.1	81.9
RBF	**92.5**	**95.9**	**93.2**	**98.0**	**98.4**	**96.1**	**94.5**	90.7
Sigmoid	48.2	52.5	25.3	47.0	52.2	54.1	28.3	33.5

Table V: Random HAH: Classification rates (%) with kernel functions and 15D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	81.7	95.4	79.9	85.3	91.8	95.9	49.6	60.5
Pol. $deg = 2$	93.1	97.1	92.0	89.4	91.7	**97.8**	92.4	88.9
Pol. $deg = 3$	**94.8**	94.1	**94.2**	88.7	93.2	96.2	96.0	**94.5**
Pol. $deg = 4$	88.7	63.4	83.1	63.6	91.5	95.8	92.7	91.6
Pol. $deg = 5$	76.2	36.9	68.7	39.6	84.6	85.4	86.1	83.7
RBF	92.1	**98.4**	93.7	**98.0**	**97.1**	97.1	**98.3**	92.1
Sigmoid	21.3	75.0	50.1	47.4	37.4	47.2	45.2	19.1

Table VI: Scatter HAH: Classification rates (%) with kernel functions and 17D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	67.8	76.3	62.3	79.0	**94.1**	85.3	33.5	8.4
Pol. $deg = 2$	90.1	91.6	85.8	93.6	93.2	**95.0**	93.0	68.0
Pol. $deg = 3$	94.7	87.0	82.7	89.3	92.9	94.1	**93.1**	85.7
Pol. $deg = 4$	92.5	58.1	67.2	63.7	85.4	93.1	74.2	**88.7**
Pol. $deg = 5$	79.4	42.6	48.3	45.4	69.9	85.1	56.7	77.9
RBF	**96.4**	**95.0**	**88.5**	**94.1**	93.4	90.2	92.9	73.3
Sigmoid	39.8	43.0	36.7	33.7	30.0	43.5	25.7	28.5

Table VII: Random HAH: Classification rates (%) with kernel functions and 17D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	91.4	95.0	74.3	83.3	79.8	94.8	35.1	47.5
Pol. $deg = 2$	**96.9**	**96.7**	87.0	83.6	87.3	**96.8**	97.3	79.9
Pol. $deg = 3$	96.1	85.8	89.9	82.1	87.4	96.3	98.0	91.9
Pol. $deg = 4$	92.3	44.1	70.7	50.7	86.0	95.3	95.1	**95.2**
Pol. $deg = 5$	81.8	28.8	59.6	34.5	67.5	83.4	73.7	84.6
RBF	95.6	97.0	**88.3**	**91.6**	**90.8**	94.7	**98.1**	74.6
Sigmoid	41.2	31.3	25.0	43.8	60.6	54.3	47.0	20.5

Table VIII: Scatter HAH: Classification rates (%) with kernel functions and 25D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	72.6	91.2	80.4	58.0	91.4	91.6	76.5	35.5
Pol. $deg = 2$	93.4	92.0	**92.2**	95.1	**96.1**	**94.9**	**93.5**	**84.2**
Pol. $deg = 3$	89.5	80.8	81.5	87.8	91.9	93.1	87.8	88.4
Pol. $deg = 4$	77.1	60.0	59.3	64.3	79.2	78.6	67.0	79.5
Pol. $deg = 5$	74.6	51.2	62.4	54.8	69.5	71.5	77.1	77.4
RBF	**95.6**	**95.9**	89.8	**95.2**	94.0	93.0	91.5	81.5
Sigmoid	28.8	49.8	30.6	33.4	32.6	28.5	19.2	30.4

Table IX: Random HAH: Classification rates (%) with kernel functions and 25D feature set.

Kernel	Class A	Class B	Class C	Class D	Class E	Class F	Class G	Class H
Linear	83.8	93.4	79.3	86.8	89.1	95.0	58.2	64.8
Pol. $deg = 2$	**96.9**	**97.1**	**94.0**	87.1	92.0	95.4	95.0	88.8
Pol. $deg = 3$	91.3	78.4	84.0	71.3	89.3	91.9	94.6	**91.7**
Pol. $deg = 4$	79.0	48.7	72.3	49.9	75.6	76.0	88.3	82.7
Pol. $deg = 5$	73.7	43.3	68.4	44.4	67.0	67.8	84.0	78.7
RBF	95.9	96.4	89.7	**92.5**	**93.1**	**95.9**	**97.2**	84.6
Sigmoid	42.6	28.5	23.1	33.8	37.9	32.5	36.5	22.4

Table X: Accuracies from the scatter HAH and random HAH (%) and the results of the statistical tests. Statistical significance between the accuracies obtained by Scatter method and random choice is marked with an asterisk.

	7D		15D		17D		25D	
Kernel	Scatter	Random	Scatter	Random	Scatter	Random	Scatter	Random
Linear	58.1	66.6*	77.3	83.8*	68.6	78.5*	78.1	84.1*
Pol. $deg = 2$	82.7	85.8*	91.3	93.3*	89.2	90.6*	**93.0**	**93.4**
Pol. $deg = 3$	90.5	90.8	92.9	94.2*	**90.3**	**91.2***	88.6*	87.6
Pol. $deg = 4$	91.9*	91.4	87.2*	86.5	80.9	81.7*	72.7	72.7
Pol. $deg = 5$	90.6	90.5	75.3*	74.0	67.2	67.7	68.1*	66.4
RBF	**93.0**	**93.2***	**95.2**	**95.9***	90.0	90.9*	91.9	93.0*
Sigmoid	47.8	58.4*	44.7*	42.4	35.9	43.1*	31.7	32.0

four different feature sets and every feature set was tested with seven kernel functions. Altogether we made 56 different test arrangements.

In the class division Scatter method worked well although it obtained a bit lower accuracies than the random division. Overall, the data was very well classifiable and it may be that even if we made the feature selection and class division in the nodes by using whatever algorithm, we would not get any better results. An important aspect is that there was less than 1% difference between the highest accuracies gained by the random and scatter methods in each feature set case.

In the future we need to research HAH-SVM with larger benthic macroinvertebrate datasets and also with some more "exotic" kernel functions not used in this paper. For the class division problem in the future we need to examine how clustering methods such as K-Means work in it. Furthermore, it is interesting to apply other classification methods, for example, linear discriminant analysis, k-nearest neighbour method, Naïve Bayes and Logistic regression with the HAH structure in benthic macroinvertebrate image classification.

ACKNOWLEDGMENT

The author wants to thank Finnish Environmental Institute, Jyväskylä, Finland for the data. The author is also thankful to the Tampere Graduate Program in Information Science and Engineering for the support. The author also wants to thank Markku Siermala, Ph.D., for the Scatter method and Jorma Laurikkala, Ph.D., for the statistical tests.

REFERENCES

[1] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, Vol. 20, No.3, pp. 273–297, 1995.

[2] T. Howley and M.G. Madden,"The genetic evolution of kernels for support vector machine classifiers," *Proceedings of AICS-2004 15th Irish Conference on Artificial Intelligence & Cognition Science*, 2004.

[3] ImageJ:public domain Java-based image processing program. Available: http://rsbweb.nih.gov/ij/docs/index.html

[4] H. Joutsijoki and M. Juhola, "Comparing the one-vs-one and one-vs-all methods in benthic macroinvertebrate image classification," *Lecture Notes in Artificial Intelligence*, 6871, Springer-Verlag, pp. 399–413, 2011.

[5] H. Joutsijoki and M. Juhola, "Kernel selection in multi-class support vector machines and its consequence to the number of ties in majority voting method," *Artificial Intelligence Review*, DOI:10.1007/s10462-011-9281-3, In press.

[6] H. Joutsijoki and M. Juhola, "Automated benthic macroinvertebrate identification with decision acyclic graph support vector machines," *Proceedings of the IASTED 2nd International Conference on Computational Bioscience*, pp. 323–328, 2011.

[7] M. Juhola and M. Siermala,"A scatter method for data and variable importance evaluation," *Integrated Computer-Aided Engineering*, In press.

[8] S. Kiranyaz, M. Gabbouj, J. Pulkkinen, T. Ince and K. Meissner, "Network of evolutionary binary classifiers for classification and retrieval in macroinvertebrate databases," *Proceedings of 2010 IEEE 17th International Conference in Image Processing*, pp. 2257–2260, 2010.

[9] S. Kiranyaz, M. Gabbouj, J. Pulkkinen, T. Ince and K. Meissner, "Classification and retrieval on macroinvertebrate image databases using evolutionary RBF neural networks," *Proceedings of the International Workshop on Advanced Image Technology*, 2010.

[10] S. Kiranyaz, T. Ince, J. Pulkkinen, M. Gabbouj, J. Ärje, S. Kärkkäinen, V. Tirronen, M. Juhola, T. Turpeinen, K. Meissner, "Classification and retrieval on macroinvertebrate image databases," *Computers in Biology and Medicine*, Vol. 41, No.7, pp. 463–472, 2011.

[11] N. Larios, J. Lin, M. Zhang, D. Lytle, A. Moldenke, L. Shapiro and T. Dietterich, "Stacked spatial-pyramid kernel: An object-class recognition method to combine scores from random trees," *2011 IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 329–335, 2011.

[12] N. Larios, B. Soran, L. G. Shapiro, G. Martinez-Muñoz, J. Lin and T. G. Dietterich, "Haar random forest features and SVM spatial matching kernel for stonefly species identification," *Proceedings of 20th International Conference on Pattern Recognition (ICPR)*, pp. 2624–2627, 2010.

[13] H. Lei and V. Govindaraju, "Half-against-half multi-class support vector machines," *Lecture Notes in Computer Science*, 3541, Springer-Verlag, pp. 156–164, 2005.

[14] D.A. Lytle, G. Martinez-Muñoz, W. Zhang, N. Larios, L. Shapiro, R. Paasch, A. Moldenke, E.N. Mortensen, S. Todorovic and T.G. Dietterich, "Automated processing and identification of benthic invertebrate samples," *Journal of North American Benthological Society*, Vol. 29, No.3, pp. 867–874, 2010.

[15] G. Martinez-Muñoz, W. Zhang, N. Payet, S. Todorovic, N. Larios, A. Yamamuro, D. Lytle, A. Moldenke, E. Mortensen, R. Paasch, L. Shapiro, and T. Dietterich, "Dictionary-free categorization of very similar objects via stacked evidence trees," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 549–556, 2009.

[16] M.J. Sarpola, R.K. Paasch, E.N. Mortensen, T.G. Dietterich, D.A. Lytle, A.R. Moldenke and L.G. Shapiro, "An aquatic insect imaging system to automate insect classification," *Transactions of the ASABE*, Vol. 51, No.6, pp. 2217-2225, 2008.

[17] S. Siegel and N.J. Jr. Castellan, *Nonparametric Statistics for the Behavioral Sciences*, McGraw-Hill, 1988.

[18] J.A.K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, Vol. 9, pp. 293–300, 1999.

[19] V. Tirronen, A. Caponio, T. Haanpää and K. Meissner, "Multiple order gradient feature for macro-invertebrate identification using support vector machines," *Lecture Notes in Computer Science*, 5495, pp. 489–497, 2009.

[20] J. Ärje, S. Kärkkäinen, K. Meissner and T. Turpeinen, " Statistical classification and proportion estimation - an application to macroinvertebrate image database," *Proceedings of 2010 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010)*, pp. 373–378, 2010.

[21] V.N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd edition, Springer, 2000.

Publication IV

A survey of classification methods in automated taxa identification of benthic macroinvertebrates

Henry Joutsijoki and Martti Juhola

Submitted to *Neural Computing and Applications*.

Neural Computing and Applications manuscript No.
(will be inserted by the editor)

A survey of classification methods in automated taxa identification of benthic macroinvertebrates

Henry Joutsijoki · **Martti Juhola**

Received: date / Accepted: date

Abstract In this research we examined the automated taxa identification of benthic macroinvertebrates. Benthic macroinvertebrates are in an important role in biomonitoring. They can be used in the assessment of a water quality. Identification of benthic macroinvertebrates is made usually by biologists, but this approach has high costs and, hence, the automation of this identification process could reduce the costs and would make wider biomonitoring possible. The automated taxa identification of benthic macroinvertebrates returns to image classification. We applied altogether 11 different classification methods to the image dataset of eight taxonomical groups of benthic macroinvertebrates. Wide experimental tests were performed. The best results, around 94% accuracies, were achieved when Quadratic Discriminant Analysis, Radial Basis Function network and Multi-Layer Perceptron were used. Also, Minimum Mahalanobis Distance Classifier obtained almost 93% accuracy. On the basis of the results, it can be said that the automated taxa identification of benthic macroinvertebrates is possible with high accuracy.

Keywords Benthic macroinvertebrates · Classification · Machine learning · Water quality

1 Introduction

Freshwater areas are in a minority position when considered all aquatic environments in the Globe. Hence, it is important to keep the current freshwater areas in good condition. Water quality monitoring has gained more and more interest when the environmental issues have come into the centre in all levels of the society. One way to monitor the water quality is to use benthic macroinvertebrates. Benthic macroinvertebrates are excellent indicators of the state of a freshwater area, like rivers, ponds or lakes. Moreover, benthic macroinvertebrates are good in environmental research (see for example [2, 4, 7, 25]). Benthic macroinvertebrates or more simply benthos are sensitive to changes in water quality. A common way to investigate the water quality is to take chemical samples from a freshwater area, but this approach only gives a snapshot about the situation of a freshwater area [26]. Benthic macroinvertebrates instead can give not only a view of current situation, but also a broader point of view about changes in the water quality after a long period. The life cycle of benthic macroinvertebrates is usually 1-2 years [26] and this fact supports the use of benthic macroinvertebrates in water quality assessments. Benthic macroinvertebrates have several advantages, why they should be used in biomonitoring. Firstly, benthic macroinvertebrates appear in all aquatic habitats and we know plenty about the consequenes of environmental effects to them [24]. Secondly, benthic macroinvertebrates are relatively immovability, so they express well localized environmental conditions [24]. Thirdly, benthic macroinvertebrates are easily to collect and, thus, they are suitable for experimental purposes [24].

Benthic macroinvertebrates are diverse organisms, since there are hundreds or even thousands of different species of benthic macroinvertebrates. A common way to interpret the situation of a freshwater area is to present the taxa richness instead of presenting the full list of species. In practice there are still benthic macroinvertebrate which cannot be identified to a species level. So, taxa richness is one measure to investigate the condition of a freshwater area. Taxa richness and water quality are connected to each other. If the taxa

H. Joutsijoki · M. Juhola
Kanslerinrinne 1
FI-33014 Tampere, Finland
Tel.: +358-50-3185860
Fax: +358-3-35516070
E-mail: Henry.Joutsijoki@uta.fi
M. Juhola
E-mail: csmajuh@sis.uta.fi

richness suddenly decreases, it indicates that the water quality has also got worse. On the other hand, if the taxa richness or the number of benthic macroinvertebrates in several species has increased, it can point out that the water quality has also improved. If a sudden decrease in the taxa richness happens, it can indicate that something unnatural has occured.

Support Vector Machines (SVMs) have become a very popular classification method. SVM was developed for binary classification, but soon the interest moved to expand SVM to also concern multi-class cases. Different multi-class expansions were quickly developed and from these the most frequently used methods are One-Vs-One (OVO), One-Vs-All (OVA) and Directed Acyclic Graph Support Vector Machines (DAGSVM). In [12] OVO strategy was applied to benthic macroinvertebrate classification and the problem of tie situations in OVO was examined. The OVO strategy was used also in [13,26] for automated taxa identification of benthic macroinvertebrates. Moreover, in [11] OVO and OVA strategies were used in benthic macroinvertebrate classification and the tie situations were closely concerned. DAGSVM was applied to the same application with a great success in [10]. Lastly, in [9] a bit rarely used variant of multi-class SVMs, the Half-Against-Half [18] strategy, was used for the benthic macroinvertebrate classification. All these articles showed that the automated taxa identification of benthic macroinvertebrates is possible to made with a high accuracy. Furthermore, SVM proved to be a very good choice for the automated taxa identification of benthic macroinvertebrates.

Generally speaking, the automated taxa identification of benthic macroinvertebrates [13,14,15,26,27] is a relatively new application area compared to areas such as handwritten digit recognition [20], text classification [23] or ECG classification [22]. The research around automated taxa identification of benthic macroinvertebrates has many advantages. Usually, the identification process is made by biologists (or taxonomists), but due to human-made identification, costs are high and the identification is a slow process. Hence, the automation of the identification process would cut costs greatly. Often the identification of benthic macroinvertebrates can be routine work for human experts. Hence, the automation of this process could relieve the workload of biologists to solve some other problems. An automated process would also enable biologists to collect larger numbers of samples, which is recommended when benthic macroinvertebrates are used in biomonitoring.

Identifying benthic macroinvertebrates from images is a demanding task from the pattern recognition point of view since differences between species or even genera can be small. There are still some taxonomical groups which are difficult to define even for taxonomists [24], so it raises the level of the problem. Furthermore, the sizes, positions and shapes of the benthic macroinvertebrates vary in each image and there can be overlapped benthic macroinvertebrates in the images which need special attention. The classification of benthic macroinvertebrates need to be reliable because, if samples are classified wrong, this can give a wrong view of the current situation of an aquatic habitat.

In this research the goal is to compare different classification methods in the automated taxa identification of benthic macroinvertebrates. Altogether 11 different classification methods are used. These are: *k*-Nearest-Neighbour Searching (with four different distance alternatives), Linear Discriminant Analysis, Quadratic Discriminant Analysis, Minimum Mahalanobis Distance Classifier, Classification Tree, Multinomial Logistic Regression, Naïve Bayes, K-Means, Self-Organizing Map, Multi-Layer Perceptron and Radial Basis Function network. Experiments with Learning Vector Quantization [16] were so poor that it was left out from this research.

In Section 2 the theory of used classification methods are presented in briefly. Section 3 explains the test arrangements, data description and the experimental results and their analysis. Section 4 concludes the research.

2 Method

2.1 *k*-Nearest-Neighbour

The *k*-Nearest-Neighbour (*k*-NN) method [5] is one of the most used classification methods. In *k*-NN the classes of *k* nearest examples are investigated by using some distance function. The class label of a new example is defined by the majority principle. That is, the class having the most examples within the *k*-nearest training examples of a new example assigns the final class label for this new example. To decrease the opportunity of a tie situation, it is a common habit to use only the odd values of *k*. In this research, for instance, odd *k* values $1,3,5,\ldots,51$ were used. There are no any exact rules for choosing the best *k* value. Thus, the usual approach is to try different values and to choose the value which gives the best performance. Another important aspect in *k*-NN method is the choice of distance function. There are numerous alternatives to choose, likewise Euclidean, Cityblock or L_∞ metrics for example. We performed the tests with four measures. These were:
Euclidean distance

$$D(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2},$$

Cityblock distance

$$D(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n}|x_i - y_i|,$$

Cosine measure

$$D(\mathbf{x},\mathbf{y}) = 1 - \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}$$

and Correlation measure

$$D(\mathbf{x},\mathbf{y}) = 1 - \frac{(\mathbf{x}-\overline{\mathbf{x}})\cdot(\mathbf{y}-\overline{\mathbf{y}})}{\|\mathbf{x}-\overline{\mathbf{x}}\|\|\mathbf{y}-\overline{\mathbf{y}}\|},$$

where $\overline{\mathbf{x}}$ and $\overline{\mathbf{y}}$ are the mean vectors. In the presentation of the distance functions we assumed that $\mathbf{x},\mathbf{y} \in \mathbb{R}^m$ and the norm in the cosine and correlation measures is Euclidean.

2.2 Quadratic Discriminant Analysis

The following representation is based on [3] and [27]. Let us have a set of classes $\{c_1, c_2, \ldots, c_N\}$. Let $P(c_i)$ denote a priori probability of the ith class, $i = 1,2,\ldots,N$. Now we have

$$\sum_{i=1}^{N} P(c_i) = 1.$$

Assume that an example $\mathbf{x} \in \mathbb{R}^m$. The class conditional probability density function is $p(\mathbf{x} \mid c_i)$ for a class c_i, $i = 1,2,\ldots,N$. By applying the Bayes theorem we obtain

$$P(c_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid c_i)P(c_i)}{p(\mathbf{x})} = \frac{p(\mathbf{x} \mid c_i)P(c_i)}{\sum_{i=1}^{N} p(\mathbf{x} \mid c_i)P(c_i)},\ i = 1,2,\ldots,N$$

where $p(\mathbf{x})$ is the unconditional probability density function for an example $\mathbf{x}$. Bayes' classification rule assigns the example to the class with the highest a posteriori conditional probability $P(c_i \mid \mathbf{x})$, $i = 1,2,\ldots,N$.

Bayes classification rule can be presented by means of discriminant functions

$$d_i(\mathbf{x}) = \ln p(\mathbf{x} \mid c_i) + \ln P(c_i),\ i = 1,2,\ldots,N. \quad (1)$$

We need to assume that example $\mathbf{x}$ has a multivariate normal Gaussian distribution within each class. Hence, every component of example $\mathbf{x}$ is normally distributed within all classes. Now the probability density function in the class c_i is

$$p(\mathbf{x} \mid c_i) = \frac{1}{(2\pi)^{\frac{m}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp[-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)], \quad (2)$$

$i = 1,2,\ldots,N,$

where $\boldsymbol{\mu}_i$ is the mean vector of the ith class feature vector and $|\boldsymbol{\Sigma}_i|$ is the determinant of the ith class covariance matrix. By substituting equation (2) into equation (1) and after eliminating the constant term $\frac{m}{2}\ln 2\pi$ we obtain

$$d_i(\mathbf{x}) = -\frac{1}{2}\ln|\boldsymbol{\Sigma}_i| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln P(c_i), \quad (3)$$

when $i = 1,2,\ldots,N$. In equation (3) there is the quadratic form for the discriminant function and, hence, the Bayes classifier can be called as Quadratic Discriminant Analysis (QDA). QDA assumes that the covariance matrices are not equal, i.e., $\boldsymbol{\Sigma}_i \neq \boldsymbol{\Sigma}_j$ when $i \neq j$. In the discriminant form a new sample will be assigned to the class having the greatest discriminant value.

2.3 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a very important special case from Quadratic Discriminant Analysis. It can obtained when assuming the covariance matrices to be equal for all classes, i.e., $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}$, $i = 1,2,\ldots,N$. Then the discriminant function from equation (3) can be expressed as follows:

$$d_i(\mathbf{x}) = -\frac{1}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln P(c_i), \quad (4)$$

$i = 1,2,\ldots,N.$

Since the term $-\frac{1}{2}\ln|\boldsymbol{\Sigma}|$ is not dependent on the classes, we can drop it out. When multiplying the vector-matrix-vector-product open, we notice that $\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i = \frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{x}$, because covariance matrix and its inverse are symmetric matrices. Furthermore, term $\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}$ is class independent, so it can be dropped out. Thus, equation (4) has the form

$$d_i(\mathbf{x}) = \boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \tfrac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i + \ln P(c_i), \quad (5)$$

where $i = 1,2,\ldots,N$. Equation (5) states now a linear discriminant function of $\mathbf{x}$. The class i, which has the greatest discriminant value $d_i(\mathbf{x})$, will be chosen as a final class for $\mathbf{x}$.

2.4 Minimum Mahalanobis Distance Classifier

Assume that we have equal covariance matrices for all i, $i = 1,2,\ldots,N$ and for all classes $P(c_i) = P$. Since $\ln P$ and $-\frac{1}{2}\ln|\boldsymbol{\Sigma}|$ are independent from the classes we can leave these terms out from the equation (3). Moreover, we can neglect the constant $\frac{1}{2}$. Hence, the discriminant function is

$$d_i(\mathbf{x}) = -(\mathbf{x}-\boldsymbol{\mu}_i)^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}_i), i = 1,2,\ldots,N.$$

Now $d_i(\mathbf{x})$ defines the squared Mahalonobis distance of $\mathbf{x}$. The classifier selects the class c_i for which $\mathbf{x}$ is the closest (when dealing with the Mahalanobis distance) to the mean vector $\boldsymbol{\mu}_i$. In other words we seek

$$\arg\min_i -(\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i),\ i = 1,2,\ldots,N.$$

2.5 Classification Tree

The following presentation is based on [5]. Let us assume that we have a classification problem of N classes. Classification tree (CT) is a general classification method which can be used with numeric and categorical variables. One commonly used classification tree algorithm is CART developed by Breiman et al. CART is a general approach to form different decision trees. A decision tree consists of nodes where in each one of them a split is made. The forming of a decision tree begins at the root node where the full training data is split. In other nodes split can be made recursively and the total number of splits is not uniquely determined and it can vary throughout a tree. An important fact is that every decision tree can be presented with binary decisions. Information about multiway splits can be found for example from [5].

Our goal is to make a decision tree as simple as possible. We want to find such features that divide the data well, since then CART works best. For this reason we need the concept of impurity. We define $i(A)$ to denote the impurity of a node A and we want that $i(A) = 0$ for all samples that reach the node having the same class label. Moreover, we want $i(A)$ to be large, if the classes are equally represented. Very often entropy impurity is used as a measure

$$i(A) = -\sum_j P(c_j)\log_2 P(c_j),$$

where $P(c_j)$ is the fraction of samples at node A that are in class c_j, $j = 1,2,\ldots,N$. Another definition for the impurity is the Gini impurity

$$i(A) = \sum_{i\neq j} P(c_i)P(c_j) = \frac{1}{2}[1 - \sum_j P^2(c_j)]$$

that CART algorithm also uses.

When using decision trees, an important question is when to stop splitting. If a decision tree is constructed so that every leaf node corresponds to the lowest impurity, data can be overfitted. Hence, the cross-validation technique suits very well for this purpose. Another alternative is to set a threshold value for impurity. It is also possible to use statistical significance, likewise χ^2-value, as a stopping criterion. Pruning is a relevant topic when CTs considered and it is an alternative to stopped splitting. In pruning nodes linked to a common antecedent node, one level above, are considered for elimination. Any pair whose elimination decreases the impurity is eliminated and, hence, the common antecedent node becomes a leaf. By this means we can simplify the structure of a decision tree and to increase the generalization ability of CT. After pruning a tree, it is common that it can be unbalanced. A different pruning method is based on rules. In this case a full tree can be represented by a large list of rules where a rule is reserved for each leaf.

One of the easiest tasks in CT is to assign the class labels for leaf nodes. If the full CT is used, each leaf node corresponds to samples in a single category. If stopped splitting or pruning is used and the leaf nodes have positive impurity, each leaf should be labelled by the class having the most samples represented. If there are missing attribute values, we can evaluate the impurity at a node A using only present attribute information. There are also other methods to handle cases with missing attribute values and more information about this can be found from [5].

2.6 Multinomial Logistic Regression

The following presentation is based on [1]. Multinomial Logististic Regression (MNLR) belongs to the group of multinomial logit models. MNLR is a generalization of the traditional logistic regression where a response can have only two values. Moreover, MNLR can have both categorical and ordinal responses and the explanatory variables can be continuous or discrete. Let $\pi_j(\mathbf{x}_i)$ denote the probability of response j, $j = 1,2,\ldots,N$, at the ith setting of values of r explanatory variables $\mathbf{x}_i = (1, x_{i1}, x_{i2}, \ldots, x_{ir})'$. Now the generalized logit model in terms of the response probabilities is

$$\pi_j(\mathbf{x}_i) = \frac{\exp(\boldsymbol{\beta}_j'\mathbf{x}_i)}{\sum_{h=1}^{N}\exp(\boldsymbol{\beta}_h'\mathbf{x}_i)} \tag{6}$$

where $\boldsymbol{\beta}$'s are vectors for the regression coefficients. For identifiability, we take $\boldsymbol{\beta}_N = \mathbf{0}$. Hence, the equation (6) obtains the form

$$\pi_N(\mathbf{x}_i) = \frac{1}{\sum_{h=1}^{N}\exp(\boldsymbol{\beta}_h'\mathbf{x}_i)}.$$

Moreover, we obtain

$$\log\left[\frac{\pi_j(\mathbf{x}_i)}{\pi_N(\mathbf{x}_i)}\right] = \boldsymbol{\beta}_j'\mathbf{x}_i,\ j = 1,2,\ldots,N-1.$$

Hence, we need $N-1$ logit equations in order to define response variable with $N-1$ categories.

When seeking the maximum likelihood estimates (parameter values which maximizes this function), we need to maximize the independent multinomial likelihood with respect to constraint in equation (6). Multinomial likelihood function is

$$\prod_{i=1}^{N}\pi_i^{\#_i}$$

where $\#_i$ is the number of responses in class i. By taking logarithm from the multinomial likelihood function we obtain the log likelihood function

$$L = \sum_{i=1}^{N}\#_i\log\pi_i,$$

which is concave and, therefore, we can find the estimate by using some iterative process such as Newton-Raphson method. More information about MNLR can be found from [1].

2.7 Naïve Bayes

The following text is based on [19]. Assume that we have a set of classes $C = \{c_1, c_2, \ldots, c_N\}$ and an example $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. The goal is to find class c_i, $i = 1, 2, \ldots, N$, which has the highest posterior probability for $\mathbf{x}$. Naïve Bayes can be derived from Bayes theorem. Bayes theorem for the class c_i and example $\mathbf{x}$ can be stated as:

$$P(c_i \mid \mathbf{x}) = \frac{P(c_i)P(\mathbf{x} \mid c_i)}{P(\mathbf{x})}. \tag{7}$$

Because we do not know $P(c_i \mid \mathbf{x})$, it must be estimated from the data, which can be a difficult task to make it directly. Bayes rule suggests to estimate probabilities $P(\mathbf{x} \mid c_i)$, $P(c_i)$ and $P(\mathbf{x})$ to evaluate $P(c_i \mid \mathbf{x})$. Estimation of $P(\mathbf{x} \mid c_i)$ consists of a problem, since there can be an arbitrary number of values for $\mathbf{x} = (x_1, x_2, \ldots, x_m)$. Hence, a commonly used method is to assume decomposition

$$P(\mathbf{x} \mid c_i) = \prod_{j=1}^{m} P(x_j \mid c_i)$$

where the occurrence of particular value of x_j is statistically independent of any other $x_{j'}$ when the example $\mathbf{x}$ is from the class c_i. Thus, we obtain equation (7) to the form

$$P(c_i \mid \mathbf{x}) = \frac{P(c_i)\prod_{j=1}^{m} P(x_j \mid c_i)}{P(\mathbf{x})}. \tag{8}$$

In classification problems we choose the class c_i for the new sample such that $P(c_i \mid \mathbf{x})$ is the highest. Equation (8) now defines the Naïve Bayes classifier. Because the denominator of the equation (8) is class independent, it can be dropped out and the equation obtains a form

$$P(c_i \mid \mathbf{x}) = P(c_i)\prod_{j=1}^{m} P(x_j \mid c_i).$$

2.8 K-Means

K-Means algorithm [3] is one of the first clustering methods. The basic idea is very simple. Assume that we have samples $x_1, x_2, \ldots, x_n$ and $x_l \in \mathbb{R}^m$, $l = 1, 2, \ldots, n$ and we are interested in dividing the samples into c clusters. Before we can represent *K*-Means algorithm, we need to define some concepts. Firstly, the sum of dispersion is

$$Q = \sum_{i=1}^{c}\sum_{l=1}^{n} u_{il}\|\mathbf{x}_l - \mathbf{v}_i\|^2 \tag{9}$$

where the squared norm is the Euclidean distance between $\mathbf{x}_l$ and prototypes $\mathbf{v}_i$. Secondly, in the equation (9) $U = [u_{il}]$ is the partition matrix, which allocates the samples to the clusters. For the partition matrix

$$u_{il} = \begin{cases} 1 \text{ if } \mathbf{x}_l \text{ belongs to cluster } i, \\ 0 \text{ otherwise.} \end{cases}$$

Partition matrix U satisfies the following two conditions:

$$0 < \sum_{l=1}^{n} u_{il} < n,\ i = 1, 2, \ldots, c \quad \text{and} \quad \sum_{i=1}^{c} u_{il} = 1,\ l = 1, 2, \ldots, n.$$

Our task is to minimize Q and to construct partition matrix U and a set of prototypes.

K-Means algorithm can be represented with a four stage algorithm.

1. Choose randomly one prototype for each cluster. Hence, we have a set prototypes $\mathbf{v}_i$, $i = 1, 2, \ldots, c$.
2. Iterate.
   2.1. Construct a partition matrix U such that
   $$u_{il} = \begin{cases} 1 \text{ if } d(\mathbf{x}_l, \mathbf{v}_i) = \min_{i \neq j} d(\mathbf{x}_l, \mathbf{v}_j) \\ 0 \text{ otherwise.} \end{cases}$$
   2.2. Update the prototypes by evaluating weighted average
   $$\mathbf{v}_i = \frac{\sum_{l=1}^{n} u_{il}\mathbf{x}_l}{\sum_{l=1}^{n} u_{il}}$$
   until Q does not change anymore, or until the changes are negligible.

2.9 Self-Organizing Map

The following text is based on [6]. Self-Organizing Map (SOM) known as Kohonen map is a widely used clustering method. The main idea is to transform an input vector into a one- or two-dimensional lattice and to make the transform adaptively in a topologically ordered fashion. Assume that the input space is m-dimensional. Let $\mathbf{x} = (x_1, \ldots, x_m)^T$ be an input vector and $\mathbf{w}_j = (w_{j1}, \ldots, w_{jm})$ be the synaptic weight vector of neuron j. After the initialization of a network, there are three processes to be performed in forming the feature map:

1. In competition the neurons in the lattice compute a discriminant value for each input vector (or minimize the Euclidean distance between the vectors $\mathbf{x}$ and $\mathbf{w}_j$). Such a neuron wins that has the highest discriminant value and, hence, the input vector is assigned for this neuron.

2. Cooperation means that the winning neuron determines the activated neurons inside its topological neighborhood.
3. In synaptic adaptation phase neurons inside the topological neighborhood can increase their discriminant values related to the input vectors by adjusting their weights.

There are four important properties in SOM.

1. A continuous input space of activation samples.
2. Network topology is normally in the form of one- or two-dimensional lattice of neurons defining a discrete output space.
3. A time-varying neighborhood function $h_{j,i(\mathbf{x})}(t)$ defined around a winning neuron.
4. A learning-rate parameter $\eta(t)$ which has the initial value of η_0 and decreases when $t \to \infty$, but never reaches zero.

For the neighbourhood function we define

$$h_{j,i(\mathbf{x})}(t) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(t)}\right), t = 0,1,2,\ldots, \quad (10)$$

where $d_{j,i}^2 = \|\mathbf{r}_j - \mathbf{r}_i\|^2$ is the Euclidean distance between the position of activated neuron j and the discrete position of the winning neuron i. Moreover, we have

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\tau_1}\right)$$

where σ_0 is the initial value of the SOM algorithm and τ_1 is a time constant. For the learning parameter $\eta(t)$ we have

$$\eta(t) = \eta_0 \exp\left(-\frac{t}{\tau_2}\right), t = 0,1,2,\ldots, \quad (11)$$

where τ_2 is another time constant in SOM algorithm.

SOM algorithm can be summarized with five steps:

1. Choose randomly initial values for weight vectors $\mathbf{w}_j(0)$ such that $\mathbf{w}_j(0)$ is different for each $j = 1,2,\ldots,l$ where l is the number of neurons in the lattice. Another alternative for initialization is to choose weight vectors $\mathbf{w}_j(0)$, $j = 1,2,\ldots,l$ randomly from the set of input vectors.
2. Take some input vector $\mathbf{x} \in \mathbb{R}^m$ from the input space. This vector $\mathbf{x}$ represents the activation sample, which is applied in the lattice.
3. Seek the winning neuron $i(\mathbf{x})$ at the time step t by using criterion

$$i(\mathbf{x}) = \arg\min_j \|\mathbf{x}(t) - \mathbf{w}_j\|, j = 1,2,\ldots,l.$$

4. Update the weight vectors of all neurons by the formula

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t) h_{j,i(\mathbf{x})}(t)(\mathbf{x}(t) - \mathbf{w}_j(t))$$

where $\eta(t)$ is a learning-rate parameter and $h_{j,i(\mathbf{x})}$ is the neighbourhood function centred around the winning neuron $i(\mathbf{x})$. Both of these parameters are dynamically changed during learning.
5. Repeat the steps 2-4 until the feature map is unchanging.

2.10 Multi-Layer Perceptron

The presentation of Multi-Layer Perceptron (MLP) and back-propagation algorithm follows the presentation from [6] and [14]. MLP belongs to the group of artificial neural networks (ANNs) and it is a feed-forward network which has an input layer, one or more hidden layers and an output layer. Hidden layers contain neurons that are no part of the input or output layer. An input layer is only a passive layer where no computations are made. Every neuron (also known as node) in MLP usually includes a nonlinear activation function. An activation function in MLP is smooth, i.e., differentiable everywhere and the common alternative for the activation function is sigmoid or hyperbolic tangent (tanh). The use of a nonlinear activation function in MLP is important since, otherwise, using a linear activation function would reduce to a single-layer-perceptron [14]. All elements in an input layer are connected to the first hidden layer neurons with the corresponding weights. Moreover, hidden layer neurons are connected with all neurons in the next hidden layer or with the neurons in the output layer. An example of an MLP network is illustrated in Figure 1. It has an input layer, two hidden layers and an output layer.

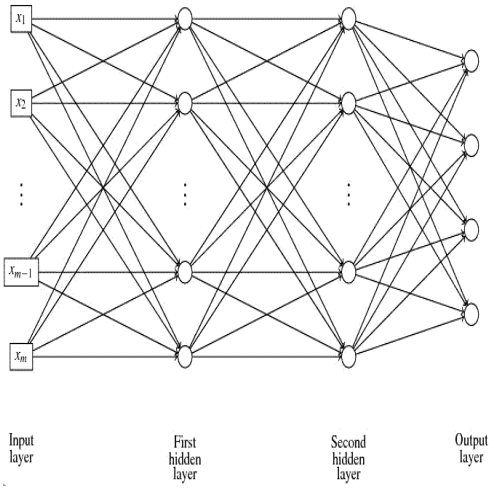


Fig. 1 An example MLP with two hidden layers.

For feed-forward ANNs the most used training algorithm is the back-propagation (BP) algorithm. It has been applied also to the automated taxa identification of benthic macroinvertebrates in [13, 14, 15]. BP is a gradient descent algorithm in the error space which can remain to a local minimum. Therefore, the choice of weights in the beginning is an important question. In BP algorithm the goal of the learning process is to minimize average error energy E_{av}. Minimizing E_{av} can need several epochs, the complete presentations of

the training set. In practice, we usually need several epochs to achieve the best training result, but too large a number of epochs can lead to overfitting to be avoided. Overfitting can reduce the generalization ability of a network.

Before we can present the summary of the BP algorithm, we need to define some notations. In notations n denotes the iteration round, i.e., presentation of the nth training example.

1. The symbol $\mathbf{x}(n)$ depicts the nth input vector and $x_j(n)$ is the jth element of the nth input vector.
2. The notation $E(n)$ depicts the sum of error energy and E_{av} is the average of all values of $E(n)$.
3. The symbol $e_j(n)$ means the error signal at the output of neuron j
4. The notation $r_j(n)$ denotes the desired output for the neuron j.
5. The notation $y_j(n)$ denotes the function signal appearing at the output of neuron j.
6. The symbol $w_{ji}(n)$ refers to the connection weight between the output of neuron i to the input of neuron j.
7. The notation $v_j(n)$ refers to the weighted sum of all inputs plus threshold of neuron j.
8. The symbol $\varphi_j(\cdot)$ denotes the activation function in neuron j.
9. The notation b_j denotes the bias applied to neuron j. Furthermore, the effect of b_j is represented by the weight $w_{j0} = b_j$ being connected to a fixed input equal to 1.
10. The symbol $o_k(n)$ refers to the kth element of the overall output vector.
11. The notation m_l denotes the number of nodes in layer l, $l = 0,1,2,\ldots,L$.

Back-propagation algorithm can be described with five steps:

1. Initialize the weights $w_{ji}^{(l)}$ and the biases $b_j^{(l)}$ randomly, when $l = 0,1,2,\ldots,L$.
2. Present the training examples for the network in some order and perform the forward and backward computations, presented in Steps 3 and 4, for each example.
3. Compute (layer-by-layer)

   $$v_j^{(l)}(n) = \sum_{i=0}^{m_l} w_{ji}^{(l)}(n) y_i^{(l-1)}(n)$$

   where $y_i^{(l-1)}(n)$ is the output signal of neuron i in a layer $l-1$ at the iteration n. Moreover, $w_{ji}^{(l)}(n)$ is the weight of neuron j in layer l that is fed from neuron i in layer $l-1$. If $i = 0$, we have $y_0^{(l-1)}(n) = 1$ and $w_{j0}^{(l)}(n) = b_j^{(l)}(n)$ which is the threshold in neuron j in layer l. Thus, the output signal of neuron j in layer l is

   $$y_j^{(l)} = \varphi_j(v_j(n)) \quad \text{or} \quad y_j^{(0)} = x_j(n),$$

   if neuron j is in the first hidden layer. If neuron j is in the output layer L, we have $y_j^{(L)}(n) = o_j(n)$. Furthermore, compute the error $e_j(n) = r_j(n) - o_j(n)$.
4. Compute the local gradients, the δs, of the network

   $$\delta_j^{(l)}(n) = e_j^{(L)} \varphi_j'(v_j^{(L)}(n)),$$

   if neuron j is in the output layer and

   $$\delta_j^{(l)}(n) = \varphi_j'(v_j^{(l)}(n)) \sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n)$$

   if neuron j is in the hidden layer l. Furthermore, index k goes through from 1 to the maximum number of neurons in the output layer. Update the weight in layer l according to the rule

   $$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n-1)] + \eta \delta_j^{(l)}(n) y_i^{(l-1)}(n)$$

   where η is the learning-rate parameter and α is the momentum constant which controls the feedback loop acting around the weight correction.
5. Repeat forward and backward computation in steps 3 and 4 with the new epochs (complete presentation of training examples) of training examples until the stopping criterion is achieved.

More details about BP algorithm and MLPs can be found from [6].

2.11 Radial Basis Function Network

The following presentation is based on [6] and [14]. Another commonly used feed-forward ANN is the Radial Basis Function network (RBFN). Compared to MLP, RBFN has a slightly different structure. RBFN has an input layer, one hidden layer and a linear output layer. Now the neurons in the hidden layer apply a nonlinear transformation to input signals. An illustration of RBFN is in Figure 2.

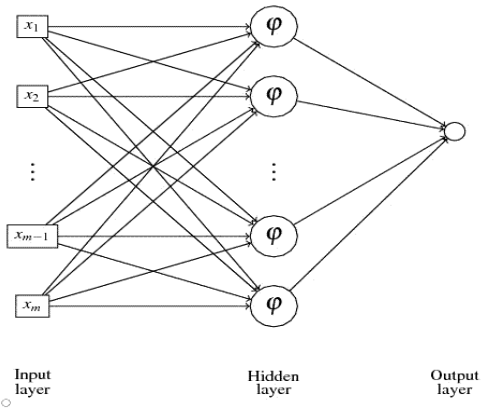


Fig. 2 An example of a Radial Basis Function network.

A difference to MLP is also that RBFN has linear weights only between the hidden layer and the output layer. Every neuron in the hidden layer contains an activation function. Activation function for the ith RBF unit is

$$y_i = \varphi\left(\frac{\|\mathbf{x}-\mu_i\|}{\sigma_i^2}\right)$$

where φ is the radial basis function, μ_i is the center of radial basis function and σ_i is the width of the peak around the center μ_i [15]. The most used activation function is the Gaussian basis function

$$y_i = \exp\left(-\frac{\|\mathbf{x}-\mu_i\|}{2\sigma_i^2}\right),$$

where μ_i and σ_i^2 are the mean and the variance. Moreover, in the Gaussian basis function the Euclidean norm is used. More information about RBFN can be found from [6].

3 Experimental Results

3.1 Test arrangements and the data description

Our dataset (1350 images) contains images from eight different taxonomical groups of benthic macroinvertebrates. These are: *Baetis rhodani*, *Diura nanseni*, *Heptagenia sulphurea*, *Hydropsyche pellucidulla*, *Hydropsyche siltalai*, *Isoperla* sp., *Rhyacophila nubila* and *Taeniopteryx nebulosa*. Seven of these taxonomical groups were identified to a species level and one, *Isoperla* sp., was recognized only to a genus level. We will refer to the groups in tables and in the following text with the abbreviations BAE, DIU, HEP, PEL, SIL, ISO, RHY and TAE. Corresponding group sizes were 116, 129, 172, 102, 271, 311, 83 and 166.

In the testing phase we used 10 times 10-fold cross-validation to the dataset. Hence, we obtained 100 training and test sets. Cross-validation distributions were selected so that every training set had as equal number of training examples from every group as possible. The same cross-validation distributions were used with all classification methods. In the case of RBF network and Multi-Layer Perceptron we divided every training set into a smaller training set and validation set. The best configuration and parameter values were selected according to the mean accuracy of the validation sets. When the best configuration was found, RBF network and Multi-Layer Perceptron were trained again with the full training set (union of smaller training set and validation set). Finally, RBFN and MLP were tested with the test sets obtained by cross-validation and a mean of the results was evaluated as a final result.

In RBF network we varied the value of σ (width of the Gaussian basis function) from $0.5, 1.0, \ldots, 20.0$ and the best value of σ was determined according to the mean accuracy of the validation sets. Moreover, MLP was tested with a single hidden layer and with two hidden layer configurations. More specifically, we tested MLP with configurations $15 \times i \times 8$ where $i = 1, 2, \ldots, 15$ and $15 \times i \times j \times 8$ where $i, j = 1, 2, \ldots, 15$. Altogether, MLP was tested with 240 different configurations. In RBF networks we limited the number of epochs to 100 for ensuring that no overfitting would happen and in MLP we set the maximum number of epochs to 150. Otherwise, we used the default values of Matlab.

We tested k-NN method with four different distance functions: Euclidean and cityblock metrics and cosine and correlation measures presented in Subsection 2.1. The k-NN method was tested with the odd integers k from 1 to 51. For the LDA, QDA, MMDC, MNLR and NB the default values of Matlab were used as well as in the case of CT, where the pruning of trees was made automatically such that the splitting criterion was 10 or more observations in impure node. We performed the classification with SOM altogether for different 43 lattices. The number of neurons varied from 8 to 50 in a lattice. Moreover, K-Means was tested with the cluster numbers ranging from 8 to 100. Because SOM is an unsupervised method, we had to define the class tag for each neuron in a lattice. This was made according to the majority principle where the class tags were determined based on the number of class members in the neuron. A class having the most samples in a neuron determined the class tag. Class tags were determined with a similar way in K-Means and, furthermore, if a tie situation happened when using the majority principle, the closest sample to the centroid in the cluster determined the final class tag for the cluster.

The dataset had altogether 25 features where 15 of them were selected to this paper. These features were divided into a geometrical and statistical features. Geometrical features included {Area, Perimeter, Width, Height, Feret's Diameter, Major, Minor, Circularity} and statistical features included {Mean, Standard Deviation, Mode, Median, Integrated Density, Kurtosis, Skewness}. More specifically, Feret's Diameter is the longest distance between any two point along the selection boundary. Major and minor are the major and minor axes of the smallest ellipse. Integrated Density is the sum of the pixel values of a image or selection [8]. More detailed information and exact definitions about the features used can be found from [8]. Before presenting the data to the classifiers, the dataset was standardized to have zero mean and unit variance. We did not make any other transformation such as normalization of the features or features scaling into intervals $[-1, 1]$ or $[0, 1]$, because we wanted to keep the classification process as natural as possible. Every transformation moves the data farther from the input space. About the preprocessing stage of the data, i.e., how the features were extracted from the images and how the scanning of the benthic macroinvertebrate were made can be found from [27]. All the tests were made with Matlab together with Sta-

Fig. 3 An example image on every taxonomical group

tistical Toolbox, Neural Network Toolbox and Bioinformatics Toolbox of Matlab.

3.2 Results

In the result tables the boldfaced numbers in the diagonal are the classification rates. Moreover, the rows of the results tables indicate the true classes and the columns indicate predicted classes. Because the group sizes vary, the contents of tables are not symmetric. In the case of *k*-NN we present the classwise classification rates with all *k* values used and we do not present the complete mean confusion matrices. Accuracies were determined by evaluating the trace of a confusion matrix (not changed into percentages) divided by the sum of the elements in a confusion matrix.

Firstly, we consider the results of *k*-NN. Figure 4 shows interesting results. The *x* axis presents the specific *k* value and the *y* axis is the corresponding classification rate with the *k* value. Class BAE was identified with all distance alternatives very well and the classification rates were similar with all distances. Generally, the level of classification in BAE was around 90% with all distances. A small increase to the classification rates came with correlation and cosine distances when $k > 20$. The second class, class DIU, was classified nearly perfectly with all distance alternatives. The same tendency continued although the *k* value was increased. Class HEP had a bit different kind of curves with the classification rates. Now for the first time, we obtained clear differences between the distance alternatives. Euclidean and cityblock measures were the best ones in the case of class HEP. Both of them obtained the best classification rates with small *k* values, likewise $k = 1, 3, 5$. The best classification rate was obtained when $k = 1$. Classification rates decreased when the *k* value became larger and this occurred with all distance alternatives. The order of the distances was that Euclidean and cityblock measures were the best ones. The third was cosine measure and the poorest results were achieved with correlation measure. The interval, in which the results spread, was quite wide since the best classification rate was above 90% and the lowest classification rate was around 50%.

For class PEL, classification rates formed interesting curves. All distance alternatives achieved similar results with all *k* values. When $k = 1$, classification rates were as their highest being nearly 100% and after that the classification rates decreased almost linearly until for $k > 11$ the classification rates stabilized with all distances to a level of around 80%. Class SIL again obtained very good results and the level of classification remained steady being within the interval of 90%-100%. The change of a distance did not bring any crucial differences to the results. Compared to the classes earlier analyzed, SIL managed likewise BAE and DIU. In class SIL correlation and cosine measures were slightly worse than Euclidean and cityblock measures but the differences were minimal. The next taxonomical group was *Isoperla* sp. (identified only to a genus level). Class ISO managed from the classification relatively well except with the correlation measure which obtained about 20% lower classification rates with every *k* value compared to other distances used. An interesting detail is that cosine measure achieved the highest classification rates when $k > 7$. Otherwise, Euclidean and cityblock measures were equally good and the results were above 90%.

Class RHY obtained very different results compared to the previous taxonomical groups. Now the diversity of the results was wider than before. A noticeable detail is that class RHY is the smallest taxonomical group in the data. With small *k* values Euclidean metric and cityblock and cosine measures obtained similar results, whereas for larger *k* value the results with cosine measure dropped dramatically. Classification rates with Euclidean and cityblock measures remained similar despite *k* value, but the general trend was downwards, when *k* value was increased. In the beginning the results with correlation measure were the poorest, but when $k > 35$ the roles between correlation and cosine measures changed. Then the classification rates with corre-

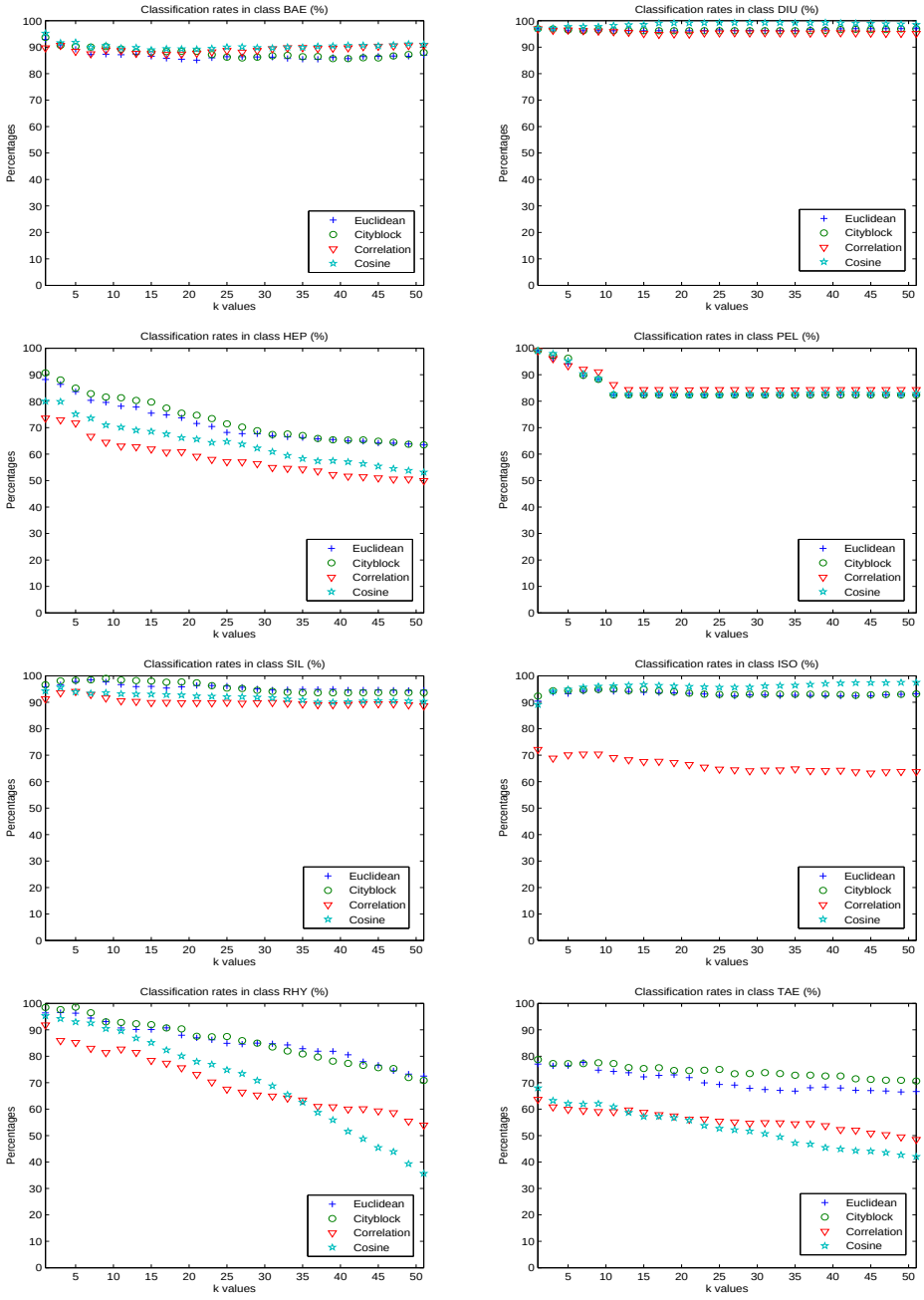


Fig. 4 Results (%) when *k*-NN used with different *k* values and measures.

lation measure were higher than with cosine. The drop in the classification rates when using cosine was significant. The highest results were achieved when $k = 1$, and it was then above 90%. On the contrary the lowest classification rate was obtained when $k = 51$, and it was then below 40%. The last class to analyze in k-NN was TAE. TAE was the only class which did not exceed 90% classification rate with any k value or distance alternative. The results of TAE were dichotomous. Euclidean and cityblock metrics remained within 70%-80% interval when the results with correlation and cosine were located in the interval of 50%-70%. Classification rates of these distances did not alter despite the increment to the k values. When considering all classes together, we noticed that the smaller k values were better than the larger k values. Especially when $k = 1$ almost every class obtained the highest classification rate.

Results, when Linear Discriminant Analysis (LDA) was applied to the benthic macroinvertebrate classification, can be seen from Table 1. LDA proved to be a good choice as the results showed. Altogether six classes from eight possible were classified above 90% classification rate. The smallest class RHY was classified perfectly and this can mean that the class RHY could be a totally separate cluster in the input space compared to the other taxonomical groups. From the first and the third row of Table 1 it can be seen that the majority of the misclassified points of the classes BAE and HEP were located to the class SIL. Moreover, nearly 15% of the misclassified samples of class PEL were identified as class RHY samples. Classes HEP and PEL were the hardest classes to classify and these two classes were the only ones which remained below 90% classification rates. In the case of class ISO nearly all misclassified samples were classified as class TAE samples and nearly 8% of TAE samples were identified incorrectly to class ISO. Overall, LDA classified well the benthic macroinvertebrate samples and the highest number of classes where the misclassified samples were spread was three and this happened in the case of class HEP which had also the lowest classification rate.

In Table 2 there are the results given by Minimum Mahalanobis Distance Classifier (MMDC). MMDC achieved very good classification rates in seven classes. Class SIL was the only class having below 90% classification rate. It obtained around 83% classification rate and the misclassified points of class SIL spread among classes BAE, DIU and HEP. From these classes BAE and HEP were the same as in the results of LDA. Classes DIU, HEP and SIL were identified with nearly perfect score. There was a significant improvement, over 16%, in classes HEP and PEL compared to the corresponding results in Table 1. Moreover, in class RHY the classification rate decreased nearly 10% from LDA results being now a bit over 90%. Moreover, all misclassified points were located in class PEL. In classes ISO and TAE all wrong classified samples were identified to the same classes as in LDA results and these classes were classified better than in Table 1.

Next we have the results of Quadratic Discriminant Analysis (QDA) in Table 3. An interesting detail is that the class TAE had identical results than in Table 2. Furthermore, in the rest of the classes, except class DIU, the misclassified samples were classified identically into the same classes as in Table 2. This might stem from the reason that QDA and MMDC are related to each other in theoretical sense. QDA obtained, generally speaking, better results than LDA or MMDC. More closely considered QDA obtained above 95% classification rates in six classes. The only exceptions were classes SIL and ISO. When compared to MMDC results, betterments were achieved in classes RHY and SIL and the greatest decrease in classification rates was in class HEP which was classified into MMDC results with nearly perfect score.

Table 4 shows the results when Classification Tree method (more specifically CART algorithm) was applied. Compared to the previous tables we can notice immediately a phenomenon that there was much more diversity in the results than before. Firstly, a majority of the classes obtained below 90% classification rates. Secondly, the misclassified samples were spread into more classes than in Tables 1-3. We still got some similarities with the previous tables. Firstly, the majority of the misclassified samples in class HEP were classified into class SIL as in Tables 1 and 3. Secondly, the classes of wrong classified samples in classes ISO and TAE were the same as in Tables 1 and 3. Thirdly, nearly 8% of the class SIL samples were identified as class HEP members and this confusion was also in the results of MMDC. When considering the diagonal elements of Table 4 it can be noticed that class TAE was identified below 80% classification rate and this result was the lowest classification rate hitherto. Classes DIU and PEL were the only ones which rose above 90% classification rate. Also, classes BAE, SIL and ISO obtained nearly identical classification rates. Although the general level of the results decreased from the previous ones, the results were still reasonably good.

Next we had the results of the Naïve Bayes (NB) method. At the first sight we can notice that only two classes, classes BAE and DIU, had above 90% classification rate, which can be thought as a limit for very good result. Especially, class DIU with 97% identification was a high-class result. Moreover, only class PEL together with the aforementioned ones reached above 80% classification rate. The rest of the classes remained below 80%. Classes HEP and RHY were classified around 62% and 66% classification rates. These results were the lowest ones. Classes SIL and ISO were identified with very close results to each other. The analysis of the misclassified samples is again an important thing. From Table 5 it can be seen the same phenomenon as

Table 1 Results (%) when Linear Discriminant Analysis used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**94.2**	0.0	0.0	0.0	5.8	0.0	0.0	0.0
DIU	0.0	**92.7**	6.9	0.0	0.4	0.0	0.0	0.0
HEP	1.2	5.7	**78.5**	0.0	14.6	0.0	0.0	0.0
PEL	0.0	0.0	0.0	**82.4**	0.0	2.9	14.7	0.0
SIL	3.0	0.0	3.9	0.0	**93.1**	0.0	0.0	0.0
ISO	0.0	0.0	0.0	0.0	0.3	**90.2**	0.0	9.5
RHY	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0
TAE	0.0	0.0	0.0	0.0	0.0	7.7	0.0	**92.3**

Table 2 Results (%) when Minimum Mahalanobis Distance Classifier used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**94.5**	0.0	1.0	0.0	4.5	0.0	0.0	0.0
DIU	0.0	**97.3**	2.7	0.0	0.0	0.0	0.0	0.0
HEP	0.0	0.4	**99.0**	0.0	0.6	0.0	0.0	0.0
PEL	0.0	0.0	0.0	**99.0**	0.0	0.0	1.0	0.0
SIL	2.3	2.0	13.1	0.0	**82.6**	0.0	0.0	0.0
ISO	0.0	0.0	0.2	0.0	0.0	**92.3**	0.0	7.5
RHY	0.0	0.0	0.0	9.6	0.0	0.0	**90.4**	0.0
TAE	0.0	0.0	0.0	0.0	0.0	4.6	0.0	**95.4**

Table 3 Results (%) when Quadratic Discriminant Analysis used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**96.6**	0.0	0.8	0.0	2.6	0.0	0.0	0.0
DIU	0.0	**97.1**	2.5	0.0	0.4	0.0	0.0	0.0
HEP	0.0	0.2	**94.5**	0.0	5.3	0.0	0.0	0.0
PEL	0.0	0.0	0.0	**98.6**	0.0	0.0	1.4	0.0
SIL	6.5	1.6	3.4	0.0	**88.5**	0.0	0.0	0.0
ISO	0.0	0.0	0.0	0.0	0.0	**92.0**	0.0	8.0
RHY	0.0	0.0	0.0	2.7	0.0	0.0	**97.3**	0.0
TAE	0.0	0.0	0.0	0.0	0.0	4.6	0.0	**95.4**

Table 4 Results (%) when Classification Tree used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**85.4**	0.0	1.6	1.0	5.9	3.4	0.0	2.7
DIU	0.0	**95.8**	1.3	0.0	2.9	0.0	0.0	0.0
HEP	0.3	0.8	**80.2**	0.3	11.8	1.2	2.9	2.5
PEL	2.0	0.0	1.9	**91.6**	0.8	2.2	1.4	0.1
SIL	1.7	0.8	7.6	0.6	**85.3**	1.7	0.0	2.3
ISO	1.4	0.0	1.2	0.5	2.4	**85.8**	0.6	7.9
RHY	0.0	0.0	5.3	3.6	0.4	3.0	**87.4**	0.3
TAE	1.4	0.0	3.0	0.0	3.3	14.8	0.2	**77.3**

Table 5 Results (%) when Naïve Bayes used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**93.1**	0.0	0.0	0.0	1.7	2.6	0.0	2.6
DIU	0.0	**97.0**	3.0	0.0	0.0	0.0	0.0	0.0
HEP	1.7	4.0	**66.1**	0.0	26.1	0.9	0.0	1.2
PEL	3.0	0.0	0.0	**82.4**	8.8	0.0	5.8	0.0
SIL	13.1	1.2	6.1	0.4	**76.6**	1.4	0.0	1.2
ISO	8.2	0.0	0.6	0.0	1.3	**74.0**	0.0	15.9
RHY	2.9	2.1	0.1	7.0	8.5	5.1	**62.1**	12.2
TAE	2.4	0.0	3.0	0.0	0.0	15.6	0.0	**79.0**

from Tables 1,3 and 4: a great number of the misclassified samples in class HEP were classified as class SIL members. Furthermore, a majority of the wrong classified samples in class ISO were located to class TAE and the same in vice versa. Overall, NB did not contrive very well from the classification compared to the previous classification methods.

Multinomial Logistic regression (MNLR) is never before used in the benthic macroinvertebrate classification. Results from Table 6 showed that MNLR was a relatively good choice for this classification problem. There were three classes, HEP, RHY and TAE, having below 90% classification rate. Now the best class was SIL recognized with nearly 96% classification rate and, also, DIU was identified very well since the classification rate achieved nearly 95%. Table 6 indicated that misclassified samples from classes BAE, DIU, HEP and PEL spread into exactly the same classes as in Table 1 where LDA was used. Furthermore, the tendency that the majority of the wrong classified samples from class ISO were identified as TAE members and vice versa, happened again. This phenomenon can also be seen from the result tables in article [12] where SVM together with one-vs-one method was applied to the benthic macroinvertebrate classification.

The first clustering method applied to the benthic macroinvertebrate classification was *K*-Means and the corresponding results can be seen in Table 7. This table was achieved by using 100 clusters. Results with *K*-Means were promising but they did not manage to win LDA, QDA or MMDC results. The results were comparable with the obtained Classification Tree results. Now there were three classes (DIU, PEL and SIL) which gained classification rates over 90%. Otherwise, the classification rates remained to 80%-90% except with class HEP, which achieved below 80% classification rate and class TAE having below 70% classification rate. The same phenomenon occured with the misclassified examples of classes BAE, HEP, ISO and TAE as in many previous result tables. Compared to the results in Table 4, the diagonal entries of classes BAE, DIU, ISO and RHY were quite close to each other. Moreover, in both methods class TAE was identified with the lowest classification rate. A noticeable detail was that class RHY was classified quite well, although it was the smallest class in the dataset.

Another clustering method used in the benthic macroinvertebrate classification was SOM and the corresponding results can be seen from Table 8. The results showed similar behaviour as in Table 7. Now classes DIU and SIL obtained over 90% classification rates. Classes ISO and TAE were classified in the same way. When considering the misclassified samples, we noticed the similar phenomena in the classes BAE, HEP, ISO and TAE as before. Moreover, we obtained similarity between *K*-Means and SOM when examined more closely wrong classified samples in class RHY. The majority of these samples were located to class ISO. So, we obtained generally interesting patterns, how some of the classes interfere with each other despite the classification method. Overall, SOM achieved a bit worse results than *K*-Means. There were two classes below 80% classification rate and one class yielded below 70% classification rate. The class with the lowest result was the same as in Tables 4, 6 and 7.

The last two result tables considered artificial neural networks and the first one of them was Multi-Layer Perceptron. Compared to the previous result tables we obtained a significant improvement to the results. The results of QDA in contrast to Table 9 are similar since in both cases the results are very good. Class TAE was the only class having below 90% classification rate and it was 89.6% result. In a misclassified sample analysis there did not happen any dramatic changes. Classes DIU, PEL, SIL and ISO were identified above 95% classification rates which is always a noticeable detail. Compared to Table 3, the results contained some individual differences. Firstly, the first four classes were classified better than with QDA, but classes SIL and ISO were, on the contrary, classified better with MLP. Especially, in the case of class SIL the difference was significant being nearly 8%. The last two classes were again recognized better with QDA.

The last classification method was RBF networks which achieved results at quite the same level as MLP did. Class TAE was the worst class to identify as in Tables 4,6,7 and 8. Misclassified samples of the classes BAE, HEP, ISO and TAE were located in the similar manner as before. Compared to Table 9 individual differences appeared. The greatest improvement happened in class RHY where RBF network classified it nearly 4% better than MLP. The other, but smaller, improvements occurred in classes BAE, HEP and SIL. Classes DIU and PEL were recognized quite evenly with both ANN methods. Class TAE was identified worse with RBF network than MLP. Overall, the results of RBF network were similar to the QDA results.

From Figure 5 we can see the accuracies of the *k*-NN method with different *k* values and distances. Cityblock and Euclidean measures were very close to each other with all *k* values, but cityblock was little better than Euclidean measure. The best accuracy was obtained with the cityblock measure together with $k = 1$ being a bit over 90%. Cosine measure was below Euclidean measure all the time but it still achieved relatively good results. The poorest results were obtained with the correlation measure, which had over 80% accuracy as its best. A common fact for distance alternatives was that the increase in *k* decreased the accuracy. Overall, it can be said that $k = 1$ is the best *k* value for this dataset.

From Table 11 the obtained mean accuracies of the 10 different classification methods can be seen. The analysis of Tables 1-10 can be confirmed with the observations in Table 11. Naïve Bayes obtained the lowest accuracy among all classification methods. Self-Organizing Map, *K*-Means and

Table 6 Results (%) when Multinomial Logistic Regression used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**92.2**	0.0	0.0	0.0	7.8	0.0	0.0	0.0
DIU	0.0	**94.7**	3.8	0.0	1.5	0.0	0.0	0.0
HEP	1.5	1.2	**83.6**	0.0	12.7	0.0	0.0	1.0
PEL	0.0	0.0	0.0	**92.6**	0.0	3.2	4.2	0.0
SIL	0.8	0.2	3.1	0.0	**95.8**	0.0	0.0	0.1
ISO	0.0	0.0	0.0	0.6	0.0	**92.8**	0.0	6.6
RHY	0.0	0.0	0.0	4.4	0.0	5.7	**89.9**	0.0
TAE	2.2	0.0	0.0	0.5	0.0	15.2	0.4	**81.7**

Table 7 Results (%) when K-means with 100 clusters used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**84.6**	0.0	0.3	0.0	14.1	0.0	0.0	1.0
DIU	0.0	**94.7**	3.4	0.0	1.9	0.0	0.0	0.0
HEP	1.6	1.1	**77.9**	0.0	18.7	0.6	0.0	0.1
PEL	0.1	0.0	0.3	**90.8**	2.5	3.1	3.1	0.1
SIL	2.8	0.1	5.3	0.2	**91.1**	0.5	0.0	0.0
ISO	0.0	0.0	0.0	0.3	1.4	**83.8**	1.9	12.6
RHY	0.0	0.0	1.0	2.8	0.0	10.3	**85.5**	0.4
TAE	0.7	0.0	0.0	0.4	0.2	29.6	0.3	**68.8**

Table 8 Results (%) when Self-Organizing Map with 50 clusters used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**80.1**	0.0	0.2	0.0	18.7	0.0	0.0	1.0
DIU	0.0	**92.5**	6.2	0.0	1.3	0.0	0.0	0.0
HEP	1.2	1.7	**72.1**	0.0	23.4	1.0	0.0	0.6
PEL	1.3	0.0	0.2	**83.4**	3.0	4.9	7.2	0.0
SIL	1.8	0.2	5.4	0.0	**92.2**	0.3	0.0	0.1
ISO	0.0	0.0	0.0	0.0	1.8	**84.9**	1.3	12.0
RHY	0.0	0.0	1.0	1.7	0.0	16.2	**79.8**	1.3
TAE	0.7	0.0	0.0	0.0	0.1	30.7	0.2	**68.3**

Table 9 Results (%) when Multi-Layer Perceptron with configuration $15 \times 15 \times 7 \times 8$ used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**93.4**	0.0	0.8	0.0	5.2	0.3	0.1	0.2
DIU	0.2	**95.9**	1.7	0.0	1.4	0.1	0.4	0.3
HEP	1.3	0.5	**92.6**	0.0	4.9	0.2	0.1	0.4
PEL	0.1	0.0	0.4	**95.8**	0.1	0.4	3.1	0.1
SIL	1.3	0.5	2.1	0.0	**96.0**	0.0	0.0	0.1
ISO	0.1	0.0	0.1	0.0	0.1	**95.3**	0.2	4.2
RHY	0.0	0.0	0.3	6.0	0.0	1.2	**92.2**	0.3
TAE	0.0	0.0	0.4	0.0	0.1	9.8	0.1	**89.6**

Table 10 Results (%) when RBF network with $\sigma = 3.0$ used. The rows of the results tables indicate the true classes and the columns indicate predicted classes.

	BAE	DIU	HEP	PEL	SIL	ISO	RHY	TAE
BAE	**90.7**	0.0	0.0	0.0	9.3	0.0	0.0	0.0
DIU	0.0	**96.0**	2.5	0.0	1.5	0.0	0.0	0.0
HEP	1.7	0.0	**90.7**	0.0	7.0	0.0	0.0	0.6
PEL	0.0	0.0	0.0	**96.3**	1.0	1.5	1.2	0.0
SIL	0.6	0.3	1.0	0.0	**98.0**	0.1	0.0	0.0
ISO	0.0	0.0	0.0	0.3	0.0	**94.0**	0.0	5.7
RHY	0.1	0.0	0.0	2.5	0.0	1.4	**96.0**	0.0
TAE	0.0	0.0	0.0	0.0	0.0	12.7	0.0	**87.3**

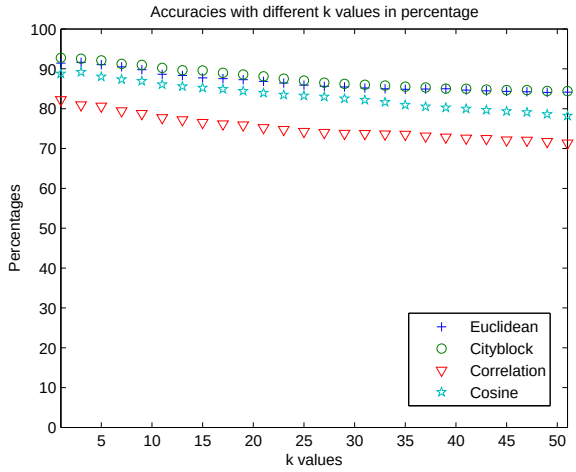


Fig. 5 Accuracies with different k values and used measures.

Table 11 Obtained weighted mean accuracies (%) with different methods.

Method	Accuracy	Method	Accuracy
LDA	90.1	MMDC	92.6
QDA	93.7	CT	85.4
NB	77.8	MNLR	90.8
K-means (100 clusters)	84.4	SOM	82.6
RBFN	93.7	MLP	94.1

Classification Tree (CT) obtained mean accuracies close to each other, whereas CT achieved the highest score. The rest of the classification methods reached above 90% accuracy. LDA and MNLR had less than 1% difference between their accuracies. Moreover, QDA and RBF network obtained the same accuracy and their difference to the best classification method, Multi-Layer Perceptron, was only 0.4%. Although QDA, RBF network and MLP achieved very good results together with the high accuracies, they did not manage to beat SVM together with one-vs-one method, which obtained above 96% accuracy with 15D features in [12].

4 Conclusion

In this research we examined the automated taxa identification of benthic macroinvertebrates. This application is an infrequently researched area. In this research we applied altogether 11 different classification methods consisting of both unsupervised and supervised methods. The dataset included 25 features from which 15 were selected to the classifications. These features were the same as used in [12, 10, 9, 13, 14, 15, 26, 27] and they are the union of geometrical and statistical features.

We made extensive experimental tests where k-NN was tested with 26 different k values and with four different distance alternatives. Moreover, the tests with K-Means were repeated with 93 different K values. Self-Organizing Map was tested with 43 different numbers of neurons and RBF network with 40 different values of σ. Finally, Multi-Layer Perceptrons were tested with configurations $15 \times i \times 8$, when $i = 1, 2, \ldots, 15$ and $15 \times i \times j \times 8$, when $i, j = 1, 2, \ldots, 15$. Altogether MLP was tested with 240 different configurations.

The obtained results were good. Many of the classification methods reached above 90% accuracy. Especially, Quadratic Discriminant Analysis, RBF network, Multi-Layer Perceptron and Minimum Mahalanobis Distance Classifier showed their power in the classification. MLP achieved the best mean accuracy being over 94% and RBF network and QDA obtained nearly 94% accuracies. Furthermore, MMDC reached nearly 93% accuracy. Although the results were good, they did not managed to win SVM used in [12, 11, 10, 9].

Our future research will concentrate on a larger, 50 species, dataset of benthic macroinvertebrates. With this dataset SVM together with different multi-class extension will be tested. Also, other classification alternatives, employed in this research, will be applied to the larger dataset. An interesting research topic is how different multi-class extensions which are developed for SVM will work on other classification methods. This topic is an infrequently researched area.

Acknowledgements We thank Finnish Environment Institute, Jyväskylä, Finland for the data. The first author is also thankful to the Tampere Graduate Program in Information Science and Engineering for the support.

References

1. Agresti A (1990) Categorical Data Analysis. John Wiley & Sons, New York, USA
2. Ambelu A, Lock K, Goethals P (2010) Comparison of modelling techniques to predict macroinvertebrate community composition in rivers of Ethiopia. Ecol Inform 5:147–152
3. Cios KJ,Pedrycz W, Swiniarski RW, Kurgan LA (2007) Data Mining: A Knowledge Discovery Approach. Springer-Verlag, New York, USA
4. Dominguez-Granda L, Lock K, Goethals PLM (2011) Using multi-target clustering trees as a tool to predict biological water quality indices based on benthic macroinvertebrates and environmental parameters in the Chaguana watershed (Equador). Ecol Inform 6(5):303–308
5. Duda RO, Hart PE, Stork DG (2001) Pattern Classification, 2nd ed. John-Wiley & Sons, USA
6. Haykin S (1999) Neural Networks:A Compherensive Foundation, 2nd ed. Prentice-Hall, New Jersey, USA
7. Hoang TH, Lock K, Mouton A, Goethals PLM (2010) Application of classification trees and support vector machines to model the presence of macroinvertebrates in rivers Vietnam. Ecol Inform 5:140–146
8. ImageJ: public domain Java-based image processing program, Available: http://rsbweb.gov.nih/ij/docs/index.html
9. Joutsijoki H (2011) Half-against-half multi-class support vector machines in classification of benthic macroinvertebrate images. In: Proceedings of the International Conference on Computer & Information Science, IEEE, pp. 1–6
10. Joutsijoki H, Juhola M (2011) Automated benthic macroinvertebrate identification with decision acyclic graph support vector machines, In: Proceedings of 2nd IASTED International Conference on Computational Bioscience (CompBio), pp. 323–328
11. Joutsijoki H, Juhola M (2011) Comparing the one-vs-one and one-vs-all methods in benthic macroinvertebrate image classification In: Proceedings of 7th International Conference on Machine Learning and Data Mining, LNCS 6871, pp. 399–413
12. Joutsijoki H, Juhola M (2011) Kernel selection in multi-class support vector machines and its consequence to the number of ties in majority voting method. Artif Intell Rev (in press), doi:10.1007/s10462-011-9281-3
13. Kiranyaz S, Gabbouj M, Pulkkinen J, Ince T, Meissner K (2010) Classification and retrieval on macroinvertebrate image databases using evolutionary RBF neural network. In: Proceedings of the International Workshop on Advanced Image Technology (IWAIT), Kuala Lumpur, Malaysia
14. Kiranyaz S, Ince T, Pulkkinen J, Gabbouj M, Ärje J, Kärkkäinen S, Tirronen V, Juhola M, Turpeinen T, Meissner K (2011) Classification and retrieval on macroinvertebrate image databases. Comput Biol Med 41:463–472
15. Kiranyaz S, Gabbouj M, Pulkkinen J, Ince T, Meissner K (2010) Network of evolutionary binary classifiers for classification and retrieval in macroinvertebrate databases. In: Proceedings of 2010 IEEE 17th International Conference on Image Processing (ICIP), Hong Kong, pp. 2257-2260
16. Kohonen, T, Somervuo P (1998) Self-organizing maps of symbol strings. Neurocomputing 12:19-30
17. Larios N, Deng H, Zhang W, Sarpola M, Yuen J, Paasch R, Moldenke A, Lytle DA, Correa SR, Mortensen EN, Shapiro LG, Dietterich TG (2008) Automated insect identification through concatenated histograms of local appearance features: feature vector generation and region detection for deformable objects. Mach Vision Appl 19:105–123
18. Lei H, Govindaraju V (2005) Half-against-half multi-class support vector machines. In: Proceedings of the 6th International Workshop on Multiple Classifier Systems (MCS 2005), LNCS 3541, pp. 156-164
19. Lewis DD (1998) Naive (Bayes) at forty: The independence assumption in information retrieval, In: Proceedings of the European Conference on Machine Learning, Chemnitz, Germany, LNCS 1398, pp. 4–15
20. Liu C-L, Nakashima K, Sako H, Fujisawa H (2003) Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recogn 36:2271-2285
21. Lytle DA, Martínez-Muñoz G, Zhang W, Larios N, Shapiro L, Paasch R, Moldenke A, Mortensen EN, Todorovic S, Diettrich TG (2010) Automated processing and identification of benthic invertebrate samples. J N Am Benthol Soc 29(3):867–874
22. Mar T, Zaunseder S, Martinéz JP, Llamedo M, Poll R (2011) Optimization of ECG classification by means of feature selection. IEEE T Bio-Med Eng 58(8):2168-2177
23. Vikramjit M, Wang C-J, Banerjee S (2007) Text classification: A least square support vector machine approach. Appl Soft Comput 7:908-914
24. Riverlife project. Available in Finnish: http://www.ymparisto.fi/riverlife (partly English). Accessed 14.2.2012
25. Song M-Y, Park Y-S, Kwak I-S, Woo H, Chon T-S (2006) Characterization of benthic macroinvertebrate communities in a restored stream by using self-organizing map. Ecol Inform 1:295–305
26. Tirronen V, Caponio A, Haanpää T, Meissner K (2009) Multiple order gradient feature for macroinvertebrate identification using support vector machines. In: Proceedings of ICANNGA 2009, LNCS 5495, pp. 489-497
27. Ärje J, Kärkkäinen S, Meissner K, Turpeinen T (2010) Statistical classification and proportion estimation - an application to macroinvertebrate image database. In: Proceedings of the 2010 IEEE International Workshop on Machine Learning for Signal Processing, Kittilä, Finland, pp. 373-378

Publication V

DAGSVM vs. DAGKNN: An Experimental Case Study with Benthic Macroinvertebrate Dataset

Henry Joutsijoki and Martti Juhola

Available at:
`http://dx.doi.org/10.1007/978-3-642-31537-4_35`

DAGSVM vs. DAGKNN: An Experimental Case Study with Benthic Macroinvertebrate Dataset

Henry Joutsijoki and Martti Juhola

School of Information Sciences
University of Tampere, Kanslerinrinne 1, FI-33014 Tampere, Finland
henry.joutsijoki@uta.fi, csmajuh@sis.uta.fi

Abstract. In this paper we examined the suitability of the Directed Acyclic Graph Support Vector Machine (DAGSVM) and Directed Acyclic Graph k-Nearest Neighbour (DAGKNN) method in classification of the benthic macroinvertebrate samples. We divided our 50 species dataset into five ten species groups according to their group sizes. We performed extensive experimental tests with every group, where DAGSVM was tested with seven kernel functions and DAGKNN with four measures. Feature selection was made by the scatter method [8]. Results showed that the quadratic and RBF kernel functions were the best ones and in the case of DAGKNN all measures produced quite similar results. Generally, the DAGSVM gained higher accuracies than DAGKNN, but still DAGKNN is a respectable option in benthic macroinvertebrate classification.

Keywords: Directed acyclic graph support vector machine, directed acyclic graph k-nearest neighbour, machine learning, benthic macroinvertebrates, water quality, kernel function.

1 Introduction

Biological issues are an important part of the modern society. Different threats are constantly present in our everday life and we need to invent new methods for monitoring and predicting the state of the surrounding nature. Freshwater areas are a sensitive part of the environment and changes in it are quickly seen with the naked eye. Illegal dumping, oil emissions and other effluents can be some of the reasons for destruction of the sensitive fauna in the water systems. How can we investigate the exact consequences of the human induced actions? Benthic macroinvertebrates live on the bottom of the waterbodies and they quickly react to any changes in the state of the aquatic environment [18]. This is why the benthic macroinvertebrates are commonly used in biomonitoring.

Benthic macroinvertebrates consist of a large variety of species. One freshwater area can have dozens of species from many taxonomical groups. Wide diversity of the benthic macroinvertebrates makes their automatic taxa identification a challenging task. Differences between species can be very small making the automated identification process even harder from the pattern recognition point of view. Traditional approach to the identification is human-based when usually biological experts, taxonomists, perform the classification. A disadvantage of this approach is that it is time-consuming and, hence, the costs are high. The main idea is to automatize the classification procedure as

P. Perner (Ed.): MLDM 2012, LNAI 7376, pp. 439–453, 2012.

far as it can be done. Thus, taxonomists and other biologists can focus their attention from often so routine identification process on more difficult and interesting problems. Moreover, biological experts can centralize their energy into solving the reasons behind the changes in the aquatic environments and to find out the solutions for these problems.

The classification of the benthic macroinvertebrates [5,6,7,10,11,12,13,14,17,18] is a difficult problem. Since the differences can be small between taxonomical groups, classification requires reliable and efficient methods. For the classification of benthic macroinvertebrates, the benthic animals are scanned and each scan was saved as an individual image. The identification of the benthic macroinvertebrates becomes even harder because they are not imaged in the same position. Moreover, the size and shape of the benthic macroinvertebrates vary in each image. Data is then heterogeneous and reflects the diversity of nature.

In this paper we have two aims to solve. Firstly, we want to investigate how Directed Acyclic Graph Support Vector Machine (DAGSVM) [5,15], a multi-class extension of SVM [1,2], succeeds in the classification of the benthic macroinvertebrate samples. Secondly, we present rarely in benthic macroinvertebrate classification used Directed Acyclic Graph k-Nearest Neighbour method (DAGKNN) and we examine how it works in this application. In feature selection we use a novel approach called scatter method [8]. In Section 2 we give a short overview of the SVM in a binary case [2,6,9] and we introduce DAGSVM [15] and DAGKNN. In Section 3 we describe data and test arrangements and, moreover, we analyse results. Section 4 is left for the discussion and further research questions.

2 Methods

2.1 Support Vector Machine

Suppose that we have a training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)$ where $\mathbf{x}_i \in \mathbb{R}^n$ are the training examples and $y_i \in \{-1, 1\}$ is the corresponding class label of $\mathbf{x}_i$. In the input space a separating hyperplane is $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ where $\mathbf{w} \in \mathbb{R}^n$ is a weight vector and $b \in \mathbb{R}$ is a bias term. A decision function can now be stated as the sign of the $f(\mathbf{x})$. If we have a linearly separable training data, we can rescale weight vector and bias term such that the closest members of both classes lie in the canonical hyperplanes $|\langle \mathbf{w}, \mathbf{x} \rangle + b| = 1$. In other words the closest training points to the hyperplane are at the distance of $\frac{1}{\|\mathbf{w}\|}$ from the hyperplane. Hence, the distance between the canonical hyperplanes equals $\frac{2}{\|\mathbf{w}\|}$. We can maximize the margin by minimizing $\frac{1}{2}\|\mathbf{w}\|^2$ subject to $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, i = 1, 2, \ldots, l$. This optimization problem can be solved by means of Lagrangian theory. Now, we want to minimize the primal Lagrangian:

$$\min_{\mathbf{w},b} L_P(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1]$$

where the Lagrange multipliers α_i's are non-negative. Moreover, L_P is maximized subject to $\boldsymbol{\alpha}$. By evaluating the derivatives respect to $\mathbf{w}$ and b and making a suitable

resubstitution, we obtain the dual form of the optimization problem. The dual form is more convenient to solve:

$$\max L_D(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \tag{1}$$

subject to $\alpha_i \geq 0$ and $\sum_{i=1}^{l} \alpha_i y_i = 0$. Training examples having positive α_i are called as the support vectors. When we have linearly non-separable problems, we need more tools. An important factor in SVM theory was the invention to use the kernel trick where the training examples in the input space are mapped with a nonlinear transformation into a higher dimensional feature space where the optimal hyperplane can be constructed again. This method can be justified according to the Cover's theorem [3]. The difference between the equation (1) and the feature space version is that in the latter one $\mathbf{x}$ is replaced with $\phi(\mathbf{x})$. It is

$$\max L_D(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle,$$

subject to $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{l} \alpha_i y_i = 0$. An important fact is that actually we do not need to make mapping into a higher dimensional space and compute the inner products $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ there, because we can use kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$. Now, the decision function can be expressed as a sign of $f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b$ where α_i's are optimal.

2.2 DAGSVM and DAGKNN

Decision Directed Acyclic Graph (DDAG) is a learning structure introduced by Platt et al. [15]. DDAG is a graph where there are no cycles and the edges have directions. The main point for this structure is to combine the binary classifiers to a multi-class classifier. The classification of a test sample begins at the root node where the classification continues via the left or right edge depending on the result of a classifier in a node. In this way we get an evaluation path for the test sample from the root node to the leaf where the final class label for the test sample exists. In M class case we need only $M - 1$ comparisons in order to solve the final class for the test sample.

DDAG structure can also be represented on a list, where every node eliminates one class from the list. In this approach a test sample is evaluated against the node which is formed from the first and the last element on a list. A test sample obtains either the class label i or j from the node and the class label that the classifier gives for a test sample remains in the list and the other will be removed from the list. Hence, we get the same result that a test sample needs only $M - 1$ comparisons in order to solve the final class. DDAG has $\frac{M(M-1)}{2}$ nodes where in everyone there is an SVM (or k-NN) classifier. DAGSVM has some advantages. Firstly, the training phase is similar that of one-vs-one method and, hence, it is computationally lighter than one-vs-all method. Secondly, the evaluation phase is fast and we do not need to handle any tie situations, when the order

of the list (or DDAG) is fixed. DAGKNN uses the same learning structure as the DAGSVM but now in the node there is a k-NN classifier instead of an SVM classifier. An example of a four-class DAGSVM can be found from [5,9]. DAGSVM and DAGKNN also contain a disadvantage which exists in the graph construction itself. We can form the list (or DDAG) in different orders. For instance, if we have an M-class classification task, the list can be formed up to $M!$ different orders and each one of these can produce different results. One of the problems is to find the optimal order. Platt et al. [15] made some limited experimental tests with different orders and they did not notice any crucial differences between the results. Because the term $M!$ grows extremely fast when M increases, it is in practise impossible (or computationally very heavy) to go through every possible order. For example, in ten class cases list can be put to 3628800 different orders.

3 Experimental Tests

3.1 Data Description and Test Arrangements

Our dataset has altogether 50 species of benthic macroinvertebrates. Benthic macroinvertebrates were scanned three times and the scannings were identified with a label set1, set2 or set3 [18]. The following preprocessing steps were made according to data including all scannings. Firstly, we sorted the species in the data into decreasing order according to their group sizes. Secondly, we divided the species into five disjoint groups such that the number of samples of each species within a group would be as equal as possible. More specifically, ten largest species were chosen to group 1 and the next ten species were chosen to group 2 etc. until the ten smallest classes formed group 5. When the division was made, the final step was to take the samples from the first scanning for the classification. This procedure explains why in Tables 1 and 2 the sizes are not always in a decreasing order. From Tables 1 and 2 the species and their sizes can be seen. Moreover, from the [5,6,7,10,11,12,17,18] some example images of the benthic macroinvertebrates can be found.

Table 1. Species and their corresponding number of samples in groups 1, 2 and 3

Group 1		Group 2		Group 3	
Species	Size	Species	Size	Species	Size
Asellus aquaticus	328	*Leuctra*	176	*Micrasema gedium*	70
Baetis muticus	290	*Limnius volckmari*	167	*Ceratopogodinae*	61
Bithynia tentaculata	292	*Baetis rhodani*	136	*Caenis rivulorum*	70
Micrasema setiferum	291	*Cheumatopsyche lepida*	126	*Arctopsyche ladogensis*	65
Nemoura	246	*Hydropsyche pellucidulla*	117	*Ephemera mucronata*	55
Ephemara aurivillii	236	*Ephemera ignita*	116	*Sericostoma personatum*	60
Myxas glutinosa	228	*Hydraena*	113	*Caenis luctuosa*	66
Ceratopsyche silfvenii	222	*Ameletus inopinatus*	113	*Pisidium*	50
Elmis aenea	185	*Callicorixa wollastoni*	84	*Heptagenia sulphurea*	54
Baetis niger	181	*Habrophlebia*	81	*Chimarra marginata*	52

Table 2. Species and their corresponding number of samples in groups 4 and 5

Group 4		Group 5	
Species	Size	Species	Size
Tanypodinae	62	*Agapetus*	24
Leptophlebia	40	*Radix balthica*	24
Sigas semistriata	39	*Gyraulus*	20
Ceratopsyche nevae	36	*Heptagenia fuscogrissea*	19
Lepidostoma hirtum	37	*Baetis digitatus*	20
Atherix ibis	31	*Oulimnius tuberculatus larvae*	13
Oulimnius tuberculatus	31	*Athripsodes*	13
Gammarus lacustris	38	*Ophiogomphus cecilia*	11
Capnia	29	*Paraleptophlebia*	12
Dicranota	27	*Wormaldia subnigra*	11

Collected benthic macroinvertebrate samples were scanned by a flatbed scanner (HP Deskjet 4850) and saved in the JPG format. Features were extracted from the images and calculated with a public Java-based ImageJ program [4]. The data has 32 features altogether. Features can be divided into two categories: simple shape features and grey value features. Accurate definitions from every feature can be found from [4]. Features in the data consists of the same features as what are used in [5,6,7,10,11,12,18] and the features {FeretX, FeretY, FeretAngle, MinFeret, AR, Round, Solidity}. The final feature selection for the classification was made with the help of the scatter method [8]. The full description of the scatter method algorithm can be found from [8]. We ran the scatter method with ten iterations for every group case to ensure that the obtained features are appropriately chosen. Furthermore, a criteria for the selected features was that their separation power in the scatter method was atleast 0.1. With the scatter method we got the following feature sets:

- Group 1:{Min, Integrated Density, Area, Perimeter, Minor, Circularity, Feret, MinFeret, AR, Round, Solidity}
- Group 2:{Mean, Mode, Min, YM, Integrated Density, Median, Skewness, Area, Y, Perimeter, Major, Minor, Feret, FeretX, MinFeret, AR, Round}
- Group 3:{Mean, Standard Deviation, Mode, Min, Max, XM, YM, Integrated Density, Median, Skewness, Kurtosis, Area, X, Y, Perimeter, Width, Height, Major, Minor, Circularity, Feret, FeretX, MinFeret, AR, Round}
- Group 4:{Mean, Standard Deviation, Mode, Min, Max, YM, Integrated Density, Median, Skewness, Area, Y, Perimeter, Major, Minor, Circularity, Feret, MinFeret}
- Group 5:{Mean, Standard Deviation, Mode, Integrated Density, Median, Skewness, Area, Perimeter, BX, Major, Minor, Circularity, Feret, MinFeret, AR, Round, Solidity}.

Because the range of the sizes of the species alternate greatly, we applied different techniques. We used a crossvalidation technique in every classification such that in case of groups 1 and 2 we utilised 10 times 10-fold crossvalidation. For group 3 10 times 5-fold crossvalidation was used and in the case of groups 4 and 5 10 times 3-fold crossvalidation was applied. Hence, we got enough samples for each training and test sets. Before

representing the data for SVM and k-NN classifiers, we standardized the columns of the data matrix in every group case to have zero mean and unit variance. Other tranformations, such as normalization or principal component analysis or linear scalings, were not used because we wanted to perform the classification procedure as close as possible to the original input space. Thus, the classification is more truthfully. We used the same parameter spaces for different kernel parameters as in [6]. Thus, for box constraint, σ and κ parameter space was $\{0.5, 1.0, \ldots, 20.0\}$ and for δ in Sigmoid parameter space was $\{-20.0, -19.5, \ldots, -0.5\}$. Hence, the RBF and Sigmoid kernel functions were tested with 1600 parameter combinations and the linear and polynomial kernel functions (degrees of 2, 3, 4 and 5) were tested with 40 parameter values. Furthermore, we made an agreement of $\kappa = -\delta$ because, otherwise, the number of the parameter combinations in Sigmoid kernel function would have increased from 1600 to 64000.

Table 3. Classification times with DAGSVM

Kernel	Group 1	Group 2	Group 3	Group 4	Group 5
Linear	1h 54min 36s	29min 19s	4min 35s	1min 55s	1min 40s
Pol. $d = 2$	2h 27min 47s	35min 32s	5min 2s	1min 59s	1min 40s
Pol. $d = 3$	2h 45min 28s	38min 18s	5min 21s	2min 3s	1min 42s
Pol. $d = 4$	3h 3min 31s	41min 9s	5min 30s	2min 7s	1min 43s
Pol. $d = 5$	3h 19min 52s	42min 53s	6min 21s	2min 15s	1min 53s
RBF	191h 27min	41h 53min 20s	5h 41min 10s	1h 50min 37s	1h 26min 10s
Sigmoid	155h 30min 20s	35h 45min	4h 37min 35s	1h 40min 14s	1h 15min 11s

We performed all the experimental tests with Dell Latitude E6500 laptop having 4GB of memory and 2.8GHz Intel Core 2 Duo processor. From the Table 3 we can see how much time was spent to classifications with different kernel functions, when all parameter combinations were tested. For the DAGKNN we performed the classification with the odd k values, which were less or equal to smallest species size in the group. Furthermore, we repeated the classification procedure with the DAGKNN altogether with four different measures. These were standard Euclidean and cityblock metrics and correlation and cosine measures and the spent time for the DAGKNN classification can be seen from Table 4. Results show that DAGKNN is faster than DAGSVM, but we need to remember that in DAGKNN we tested less k values than the parameter combinations in DAGSVM. We used the binary SVM implementation of Bioinformatics Toolbox of Matlab as a basis for our tests and all tests were made with Matlab. We used the Least Square method [16] in finding the optimal hyperplane.

Optimal parameter values in DAGSVM were chosen the following way. We present it in a general way. We had $10 \times \mu$ disjoint training and test sets. Firstly, we trained each binary SVM using suitable subsets from the full training data. Secondly, we evaluated the accuracy of the training set by giving the full training set as a test set to trained SVMs. Thirdly, we evaluated the accuracy of the real test set with the trained SVMs. The final accuracy for the specific parameter combination was the average of the $10 \times \mu$ accuracies. Hence, for all parameter combinations we obtained a pair of values where

Table 4. Classification times with DAGKNN

Distance	Group 1	Group 2	Group 3	Group 4	Group 5
Euclidean	1h 24min 54s	11min 47s	2min 27s	36s	11s
Cityblock	1h 23min 26s	11min 45s	2min 26s	35s	11s
Correlation	1h 27min 29s	13min 10s	2min 43s	40s	12s
Cosine	1h 27min 9s	12min 48s	2min 40s	39s	12s

Table 5. Kernel parameter values

Kernel	Group 1	Group 2	Group 3	Group 4	Group 5
Linear	(20.0)	(19.5)	(1.5)	(18.5)	(5.5)
Pol. $d = 2$	(16.5)	(7.0)	(0.5)	(0.5)	(0.5)
Pol. $d = 3$	(0.5)	(0.5)	(0.5)	(0.5)	(0.5)
Pol. $d = 4$	(0.5)	(0.5)	(0.5)	(0.5)	(0.5)
Pol. $d = 5$	(0.5)	(0.5)	(0.5)	(1.0)	(0.5)
RBF	$(20.0, 1.0)$	$(20.0, 1.5)$	$(20.0, 1.5)$	$(20.0, 3.5)$	$(18.0, 3.5)$
Sigmoid	$(19.5, 20, -20)$	$(20, 19.5, -19.5)$	$(9.5, 19.5, -19.5)$	$(17.5, 10, -10)$	$(16, 4.5, -4.5)$

the first element was the mean accuracy of the training sets and the second element was the mean accuracy of the test sets.

Overfitting is always an existent problem when using SVM. If too large parameter values are given to SVMs, a model becomes too complex and its generalization ability weaken. Thus, the classification error in a training set tends to zero and in a test set it tends to one. Hence, the final parameters were chosen with an easy method. We calculated

$$\arg\min_i[(1 - ACC_{TRAIN,i}) + 2 \cdot (1 - ACC_{TEST,i})]$$

where i is the index for parameter combination and ACC is the accuracy. In other words we sought that parameter combination index which gained the minimum of the weighted sum of the training and test set classification errors. Weightening was made in order to prevent possible tie situations and to separate those parameters which caused overfitting. By this means we do not always get those parameters which give the best accuracy in the test set, but we get a compromise where the accuracy of a training set is determined from a full training set and we take also into account the accuracy of the test set. Other possible ways to determine the best parameter values are nested cross-validation which is time-consuming or to use a validation set technique. In both cases a disadvantage is that the optimal parameter values are determined from a smaller set than the actual training data. In Table 5 we see the obtained kernel parameter values from each group. In kernel functions from the linear to the 5th degree of polynomial kernel function, we had only one parameter, box constraint, and it is in the parenthesis. In RBF the first value is the box constraint and the second one is σ. In Sigmoid the first value in the parenthesis is the box constraint, the second value is κ and the last one is δ.

3.2 Results

In the following tables we have compressed the results such that when the results of DAGSVM are presented, every row in the result table indicates the classification rates with a specific kernel function. When DAGKNN is in question, a row of the result table indicates the classification rates with specific distance alternative and k value. Furthermore, the last column in the result tables depicts accuracy obtained from a specific kernel function or specific distance alternative with some k value for easyning the analysis of the results. Class labels in result tables are abbreviations from the latin-based names of the species in Tables 1 and 2. We boldfaced the best classification rate (or classification rates in the case of tie situations) from each column of the result tables for facilitating analysis. Sigmoid kernel function was the worst kernel function in each group so we do not take it into account in our analysis. Reasons behind the poor results with Sigmoid kernel funtions may lie in the preprocessing. Normalization of the data or linear scaling to interval $[-1, 1]$ or $[0, 1]$ after the stardardization could have increased the general level of Sigmoid, but every transformation what we make to the data draws the situation more away from the original input space. Throughout all classification results with DAGKNN small k values (odd integers from 1 to 9) gave the best results. From Tables 6 and 7 we find the results when DAGKNN and DAGSVM

Table 6. DAGKNN: Results (%) with different distance alternatives and k values in group 1

		ASE	MUT	BIT	SET	NEM	AUR	MYX	SIL	ELM	NIG	Mean accuracy
Euclidean	$k=5$	78.5	71.5	88.9	90.4	47.3	65.9	84.2	96.2	**89.4**	61.2	77.7
	$k=7$	78.9	72.9	89.4	89.7	47.8	65.8	83.1	**96.5**	87.9	60.7	77.7
	$k=9$	78.8	73.2	89.2	89.6	47.7	64.7	83.1	96.0	88.4	63.1	77.7
Cityblock	$k=5$	**80.4**	73.6	88.4	**91.3**	49.3	**66.2**	84.1	95.7	88.9	**66.5**	**78.7**
	$k=7$	78.1	**75.2**	89.6	91.1	49.1	**66.2**	84.5	96.1	88.1	66.2	**78.7**
	$k=9$	77.8	**75.2**	**90.6**	90.3	48.9	66.1	84.0	95.5	86.9	**66.5**	78.5
Correlation	$k=3$	79.4	64.4	87.1	89.7	**54.8**	62.3	79.4	94.8	74.9	56.2	75.1
	$k=5$	77.1	66.8	87.1	89.5	54.5	62.5	81.9	94.2	77.8	54.3	75.3
	$k=7$	76.1	68.7	88.2	89.4	53.8	62.3	81.3	95.4	76.1	55.0	75.4
Cosine	$k=3$	77.6	66.8	85.6	90.6	53.6	64.1	82.2	94.8	87.1	60.3	76.6
	$k=5$	75.9	67.0	85.6	91.1	53.7	63.6	83.3	95.4	87.9	59.6	76.5
	$k=7$	73.8	67.9	86.0	90.4	54.5	64.4	**84.6**	96.3	88.3	60.9	76.8

were used to classify group 1. The DAGKNN achieved very similar accuracies with all measures. Accuracies in Table 6 were within 4% interval, but the best accuracy, nearly 79%, was obtained by the cityblock metric when $k = 5$ and $k = 7$. Classes BIT, SET and SIL were identified above 90% classification rate and from these classes SIL was recognized with classification rate over 95% which is a very good result. Class ELM got nearly 90% classification rate and other classes which obtained above 80% classification rate were MUT and MYX. The poorest results were in the classes NEM, AUR and NIG. These classes had below 67% classification rates and especially NEM clearly separated from other classes having below 55% classification result.

Table 7. DAGSVM: Results (%) with different kernel functions in group 1

	ASE	MUT	BIT	SET	NEM	AUR	MYX	SIL	ELM	NIG	Mean accuracy
Linear	77.8	71.2	77.2	96.0	67.6	71.0	86.8	93.1	89.3	76.0	80.3
Pol. $d = 2$	**85.0**	**80.3**	92.4	97.2	**73.5**	**77.6**	**93.5**	**96.6**	93.5	76.5	**86.7**
Pol. $d = 3$	84.7	78.8	**94.1**	97.4	71.0	76.4	92.5	96.4	**94.5**	**76.8**	86.4
Pol. $d = 4$	78.3	75.8	91.5	**98.0**	70.3	69.5	89.2	83.4	93.6	68.1	82.1
Pol. $d = 5$	61.8	66.9	86.3	97.0	64.5	58.0	86.2	56.1	93.0	57.1	73.0
RBF	84.6	76.7	92.3	96.4	66.6	73.6	90.4	96.2	93.3	**76.8**	84.8
Sigmoid	0.7	21.0	34.8	67.5	32.3	33.6	77.4	57.5	38.7	13.5	36.8

DAGSVM succeeded in the classification of group 1 better than DAGKNN. Five from the seven kernel funtions obtained over 78.7% accuracy. Particularly the quadratic kernel function obtained a high accuracy being 86.7%. Almost the same accuracy was the cubic kernel function which was only 0.3% inferior to the quadratic kernel function. The third noteworthy kernel was RBF which obtained nearly 85% accuracy. The same classes, as in DAGKNN, NEM, AUR and NIG were the hardest classes to identify. Classes BIT, SET, MYX, SIL and ELM got very high classification rates, since the results of these classes were over 93%. Classes NEM, AUR and NIG were the three hardest classes to classify as in the corresponding DAGKNN case.

Table 8. Results (%) with different distance alternatives and k values in group 2

		LEU	LIM	BAE	CHE	PEL	IGN	HYD	AME	CAL	HAB	Mean accuracy
Euclidean	$k = 3$	74.7	87.9	66.8	78.4	68.9	**68.4**	99.9	61.4	91.5	**43.8**	75.1
	$k = 5$	75.5	**89.2**	76.8	78.6	68.5	68.0	99.6	**63.6**	94.2	40.2	**76.5**
	$k = 7$	74.5	88.7	**77.4**	**79.1**	67.2	63.1	**100.0**	62.7	93.9	35.7	75.5
Cityblock	$k = 5$	75.5	87.5	71.7	76.9	69.0	65.5	**100.0**	60.7	**97.6**	38.0	75.3
	$k = 7$	73.5	88.1	74.9	76.9	68.4	64.9	**100.0**	60.8	96.0	35.1	75.0
	$k = 9$	72.3	88.0	76.8	78.7	66.6	64.2	**100.0**	60.5	96.2	32.3	74.8
Correlation	$k = 3$	80.1	85.7	66.6	77.5	68.1	58.8	99.2	55.6	88.5	30.9	72.8
	$k = 5$	**81.6**	86.9	71.3	78.4	68.1	57.9	98.4	52.5	87.3	26.4	73.0
	$k = 7$	81.2	85.6	70.3	78.8	66.2	55.2	98.2	52.6	88.2	28.1	72.4
Cosine	$k = 3$	80.9	86.0	72.5	77.3	**73.3**	58.1	98.4	56.5	90.7	43.0	75.0
	$k = 5$	80.1	86.2	69.8	77.6	71.5	55.5	98.2	56.7	92.1	43.4	74.3
	$k = 7$	81.0	85.6	68.2	76.2	70.6	53.4	98.2	57.8	91.5	39.2	73.6

Tables 8 and 9 show the compelling results for group 2. Compared to Table 6 the general level of classification decreased a bit. Now, the best accuracy was gained by the Euclidean metric when $k = 5$, but all the accuracies still were within 5% range. An eye-catching detail was that the class HYD obtained a perfect 100% classification rate four times when using DAGKNN, but in Table 9 we do not see any kernel function which would have managed to do this. Another very well recognized class was CAL which had almost 98% classification rate. Furthermore, classes LEU and LIM achieved above 80%

Table 9. Results (%) with different kernel functions in group 2

	LEU	LIM	BAE	CHE	PEL	IGN	HYD	AME	CAL	HAB	Mean accuracy
Linear	80.5	92.6	65.5	79.5	71.8	67.1	99.1	61.4	92.3	74.8	78.7
Pol. $d = 2$	80.0	**94.7**	74.6	80.0	71.7	**73.8**	99.2	**69.4**	**96.3**	72.5	81.4
Pol. $d = 3$	75.7	91.0	73.7	**81.1**	62.3	70.2	99.2	68.0	85.3	70.5	78.1
Pol. $d = 4$	53.5	81.2	61.5	67.9	43.2	47.4	99.6	59.8	71.3	50.4	63.9
Pol. $d = 5$	35.2	73.0	48.3	60.3	25.5	50.3	98.4	50.6	70.9	35.2	54.5
RBF	**82.6**	92.4	**75.5**	80.9	**83.6**	71.3	**99.8**	65.8	91.9	**75.2**	**82.2**
Sigmoid	27.8	7.1	37.7	43.3	44.6	24.4	94.5	49.4	72.3	4.4	38.5

classification rates which is always a good result. Other classes obtained below 80% classification rates and especially HAB distinguished from all classes having clearly under 50% classification rate. Also, classes IGN and AME were quite poorly recognized since they obtained below 70% classification rates. Rest of the classes were identified above 70%, but still under 80% classification rates.

Table 10. Results (%) with different distance alternatives and k values in group 3

		GED	CER	CAE	ARC	MUC	SER	LUC	PIS	SUL	CHI	Mean accuracy
Euclidean	$k = 1$	**95.7**	**98.4**	77.4	97.0	55.5	74.2	75.9	90.6	67.5	87.5	82.4
	$k = 3$	95.1	**98.4**	77.7	97.1	51.7	78.7	79.6	87.9	70.0	90.6	83.1
	$k = 5$	93.5	**98.4**	76.0	98.5	51.1	77.7	**81.1**	88.1	71.2	89.3	82.9
Cityblock	$k = 1$	95.3	**98.4**	77.5	98.2	**56.4**	76.0	78.9	89.2	73.1	84.8	83.2
	$k = 3$	94.7	**98.4**	78.3	97.9	54.5	**79.3**	79.0	87.4	71.0	**91.4**	**83.6**
	$k = 5$	94.0	**98.4**	77.7	98.5	53.3	78.5	79.3	89.0	**73.3**	89.9	83.5
Correlation	$k = 1$	93.9	**98.4**	78.0	99.2	54.0	71.5	69.4	86.7	56.9	83.9	79.8
	$k = 3$	94.7	98.2	80.5	**100.0**	51.8	73.2	74.2	85.7	44.3	84.5	79.6
	$k = 5$	95.1	98.2	77.5	**100.0**	48.1	76.4	77.3	85.7	44.3	79.1	79.1
Cosine	$k = 1$	94.2	**98.4**	79.0	99.8	56.3	68.0	73.6	**90.7**	50.3	84.6	80.1
	$k = 3$	95.5	**98.4**	**80.7**	**100.0**	51.4	71.8	79.0	86.3	48.8	87.0	80.7
	$k = 5$	95.1	**98.4**	76.5	**100.0**	49.3	73.7	81.0	87.8	48.0	84.4	80.2

When in group 1 five kernel functions achieved better accuracies than the maximum accuracy with DAGKNN, now in group 2 four kernel functions had better accuracies than the maximum accuracy, 76.5%, with the DAGKNN. However, the best kernel functions were again the quadratic and RBF kernels which obtained classification rates over 80%. The RBF obtained the highest classification rate being 82.2%. Class HYD had a nearly perfect score and was clearly the most distinguished class together with CAL and LIM among all classes. Below 90% but above 80% classification rates were obtained for LEU, CHE and PEL. The lowest identifications were attained to the classes BAE, IGN, AME and HAB which got below 80% classification rates. When doing a search for equal classwise results between Tables 8 and 9, our eyes are focused on classes LEU, BAE, CHE, HYD and CAL. Other classes obtained larger differences. The general level

of DAGSVM results was lower than in Table 7 and this tendency was seen analogously in DAGKNN results. It needs to remember that the results between groups are not directly comparable since they consist of a totally different species and the group sizes vary greatly between groups.

Next we have classification results in group 3. In Table 10 we obtained interesting results. Firstly, the level of classification between different species was widely spread. Secondly, classes GED, CER, ARC, PIS and CHI were identified very well having above 90% classification rates and, specially CER was recognized with a nearly perfect score with all measures and ARC obtained a perfect classification when using correlation and cosine measures. Classes CAE and LUC got above 80% classification rates with cosine and Euclidean measures. Other classes were left below 80% classification rate and the most difficult class to recognize was MUC having the maximum classification rate below 60%. Accuracies were also separated since Euclidean and cityblock metrics achieved accuracies over 82%, but cosine and correlation measures were left below 81% accuracies.

Table 11. Results (%) with different kernel functions in group 3

	GED	CER	CAE	ARC	MUC	SER	LUC	PIS	SUL	CHI	Mean accuracy
Linear	92.7	**96.7**	**81.3**	**98.5**	**66.2**	79.9	73.4	89.6	**84.8**	88.9	85.3
Pol. $d = 2$	**95.0**	**96.7**	78.6	97.4	61.9	72.5	**79.7**	**93.2**	73.7	83.5	83.5
Pol. $d = 3$	92.3	95.6	79.9	78.7	58.3	66.3	71.6	86.1	66.2	72.3	77.2
Pol. $d = 4$	89.2	91.4	75.8	64.4	53.9	64.9	65.7	82.7	69.2	75.4	73.5
Pol. $d = 5$	77.8	87.6	72.9	54.2	52.4	62.4	63.7	78.0	65.9	69.0	68.5
RBF	93.1	**96.7**	80.9	98.3	61.3	**81.9**	**79.7**	91.7	81.2	**92.3**	**85.9**
Sigmoid	52.7	59.0	49.6	93.1	15.4	32.8	45.0	81.6	17.8	7.8	46.5

When compared DAGKNN accuracies with the DAGSVM accuracies, DAGKNN accuracies were left behind the DAGSVM accuracies. The linear and RBF kernel functions gained better accuracies than the maximum accuracy in the DAGKNN results. Three kernel functions of the seven possible were distinguished from the rest. These were the linear, quadratic and RBF kernel functions and these three kernel functions were also the best ones in Table 11. It seems that the quadratic and RBF kernels are the best choices for this classification task. In classwise examination, again, classes GED, CER, ARC, PIS and CHI were the best classifiable classes. This is consistent with the DAGKNN results. Furthermore, in the cases of CAE, SER and LUC differences between the best results of DAGKNN and DAGSVM were not large. The most significant differences came in classes MUC and SUL. Class MUC was identified almost 10% better with DAGSVM than with DAGKNN. Moreover, class SUL was recognized over 11% better in DAGSVM with the linear kernel function. With the linear and RBF kernel functions above 85% accuracy was achieved when the quadratic kernel obtained 83.5% result. The cubic, 4th and 5th degree of the polynomial kernel functions and Sigmoid got below 80% accuracies.

DAGKNN succeeded in the classification of the fourth group quite similarly to group 1 classification. The Euclidean and cityblock metrics, when $k = 3$ or $k = 5$, got 79%

Table 12. Results (%) with different distance alternatives and k values in group 4

		TAN	LEP	SIG	NEV	HIR	IBI	OUL	GAM	CAP	DIC	Mean accuracy
Euclidean	$k=1$	86.1	71.8	95.3	93.9	59.3	51.3	88.0	**89.6**	65.6	**70.4**	78.4
	$k=3$	90.6	80.9	95.3	93.0	54.6	60.0	94.3	88.3	60.3	63.7	79.8
	$k=5$	93.0	80.0	96.0	94.2	48.2	51.3	97.7	84.8	60.8	65.4	79.0
Cityblock	$k=3$	90.9	**83.1**	95.8	93.8	57.1	56.0	97.8	88.8	63.1	58.6	**80.4**
	$k=5$	92.7	80.7	96.3	95.0	48.6	47.0	98.4	84.5	63.4	63.1	78.9
	$k=7$	92.9	82.2	97.1	95.8	43.2	40.9	97.8	82.4	62.1	61.4	77.7
Correlation	$k=1$	89.0	63.1	89.9	92.2	56.2	**61.9**	93.3	89.3	**67.5**	46.0	76.6
	$k=3$	94.5	60.3	91.9	93.1	58.0	56.5	99.7	87.2	60.7	47.4	77.1
	$k=5$	**96.9**	58.9	94.5	94.2	55.7	47.4	**100.0**	85.3	61.6	55.2	77.2
Cosine	$k=1$	88.3	67.8	90.1	93.2	**63.2**	59.3	95.8	**89.6**	64.7	52.2	78.1
	$k=3$	91.0	69.6	**97.4**	**96.6**	58.0	55.2	99.7	88.8	58.1	48.4	78.4
	$k=5$	93.2	72.0	96.6	93.9	48.7	48.1	**100.0**	84.7	61.9	51.2	77.2

or higher accuracy, but the difference between the worst and the best accuracy was only less than 4%, so the same trend continued in group 4 classification than in the previous ones. Three from seven kernel functions achieved better accuracy than DAGKNN with the cityblock metric when $k = 3$. In this particular case DAGKNN obtained an accuracy over 80%. The linear, quadratic and RBF kernels performed above 83% accuracy which is a very good result and the highest accuracy 86.3% was reached by the RBF. When examining the DAGKNN results more closely, we noticed that a large part of the highest classification rates were among the correlation and cosine results.

Table 13. Results (%) with different kernel functions in group 4

	TAN	LEP	SIG	NEV	HIR	IBI	OUL	GAM	CAP	DIC	Mean accuracy
Linear	90.6	72.0	88.1	96.9	70.4	**77.3**	**94.7**	89.8	**81.4**	75.0	84.2
Pol. $d=2$	88.5	83.3	93.0	**97.5**	70.1	68.3	93.5	80.9	78.5	70.0	83.2
Pol. $d=3$	80.5	78.9	81.3	90.3	53.8	64.4	91.6	67.8	64.3	54.4	73.8
Pol. $d=4$	68.2	73.0	69.8	77.8	50.4	55.9	85.8	62.3	63.7	51.6	66.3
Pol. $d=5$	65.2	68.2	63.8	70.5	47.1	48.7	81.6	59.7	61.0	52.2	62.2
RBF	**93.7**	**83.6**	**94.5**	95.6	**73.7**	71.9	93.9	**91.7**	75.7	**78.3**	**86.3**
Sigmoid	72.4	44.5	41.5	84.5	33.8	36.6	58.8	42.8	34.2	25.3	49.7

The best classification results can be divided into three categories. Firstly, the classes TAN, SIG, NEV and OUL got excellent classification rates being above 96%. Especially, in the case of class OUL we obtained a perfect 100% classification rate with correlation and cosine measures. Compared to DAGSVM, where the classes also achieved above 90% classification rates, there are not any perfect scores. Secondly, in the DAGKNN classification rates of classes LEP and GAM were located into interval 83%-90%. The best classification rate of class LEP in the DAGSVM was nearly identical with the classification rate in DAGKNN. Thirdly, the rest of the classes in DAGKNN had below 71% classification rates and these classes were consistently classified better with DAGSVM. The RBF kernel function in Table 13 obtained six topmost classification rates

Table 14. Results (%) with different distance alternatives and k values in group 5

		AGA	RAD	GYR	FUS	DIG	TUB	ATH	OPH	PAR	WOR	Mean accuracy
Euclidean	$k=1$	90.7	84.7	62.8	81.2	71.5	65.8	46.3	72.7	56.0	**88.8**	73.8
	$k=3$	**93.7**	75.4	76.0	85.8	80.0	65.2	49.0	54.9	56.8	86.4	75.0
	$k=5$	91.2	80.3	82.1	86.4	84.1	62.1	44.7	46.9	53.0	82.0	75.0
Cityblock	$k=1$	90.3	**88.0**	78.3	85.7	81.5	66.5	**63.2**	67.4	56.0	83.8	78.4
	$k=3$	89.9	80.4	81.2	**88.0**	87.8	74.2	55.7	62.4	**58.5**	**88.8**	**79.0**
	$k=5$	89.4	82.5	**84.1**	85.3	86.6	69.2	46.8	47.2	52.2	**88.8**	76.5
Correlation	$k=1$	85.6	72.4	65.3	74.8	85.6	64.7	40.5	**73.0**	44.3	74.8	70.4
	$k=3$	83.5	69.0	73.4	79.5	**94.4**	76.5	38.8	70.5	53.7	84.5	73.9
	$k=5$	78.4	68.3	76.8	76.4	92.3	**80.0**	29.0	72.2	34.4	83.9	71.2
Cosine	$k=1$	89.0	71.6	61.7	76.7	85.2	61.5	47.3	71.3	56.0	82.1	72.0
	$k=3$	84.8	71.7	73.8	81.1	93.0	76.7	46.5	70.5	56.0	87.3	75.6
	$k=5$	82.6	69.6	78.0	77.5	91.8	77.7	33.0	71.5	38.2	86.7	72.7

from all classes and the rest of the classes were classified with the highest classification rates when the linear and quadratic kernel function were used. More details about the results of group 4 can be found from Tables 12 and 13.

Table 15. Results (%) with different kernel functions in group 5

	AGA	RAD	GYR	FUS	DIG	TUB	ATH	OPH	PAR	WOR	Mean accuracy
Linear	**91.8**	75.8	82.5	83.9	**86.9**	**84.5**	**86.5**	69.6	**74.4**	80.7	82.6
Pol. $d=2$	85.7	78.3	78.1	**89.3**	83.4	69.8	75.8	51.2	60.9	**87.7**	77.9
Pol. $d=3$	84.5	62.3	75.1	87.6	77.9	70.3	73.0	32.7	47.8	71.7	70.7
Pol. $d=4$	82.8	57.0	68.6	87.2	70.7	62.5	69.2	22.9	48.8	65.0	66.1
Pol. $d=5$	76.8	57.0	60.8	85.9	66.1	55.7	70.3	26.6	52.2	64.6	63.7
RBF	85.6	**89.1**	**87.1**	83.2	86.6	79.7	72.2	**70.7**	74.1	**87.7**	**82.9**
Sigmoid	27.4	60.3	64.0	68.9	46.7	51.4	21.8	54.4	16.2	31.6	46.4

In the last group the classification level of DAGKNN was spread out more wide interval than in the previous cases. Table 14 shows that the accuracies were spread from 70.4% to 79% which was achieved by the cityblock metric. The highest classification rates were obtained among the Euclidean, cityblock and correlation distances. Two of ten classes were identified above 90% classification rate and these two classes AGA and DIG gained 93.7% and 94.4% results. Classes RAD, FUS and WOR were recognized with classification rate 88% or 88.8%. Also, classes GYR and TUB gained classification rates of 80% or higher. Classes ATH, OPH and PAR were the hardest classes to classify and from these OPH gained 73% and the rest were left below 64% results. Compared to DAGKNN, DAGSVM (see Table 15) did not succeeded much better. Now, only two of seven kernel functions got higher accuracies than 79%. These were the linear and RBF kernel functions and their corresponding accuracies were 82.6% and 82.9%. The linear kernel obtained five times the topmost classification rate among all classes and the RBF got four times the topmost results. Only the class AGA was classified with a

classification rate over 90% in DAGSVM, in DAGKNN there were two classes which got above 90% classification rate. Classes ATH, OPH and PAR were also the hardest classes to identify as it was in DAGKNN case. The best classification rates of the rest of the classes were between 84.5% and 89.1%. Accuracies of the linear and RBF kernel function were very close to each other since they had only 0.3% difference. The linear and RBF kernel functions were the only ones that achieved above 80% accuracies when taking into account also the results of DAGKNN.

4 Discussion

We applied in this paper DDAG learning structure to SVM and k-NN classifiers. DAGSVM was applied to benthic macroinvertebrate identification in [5] with great success and it inspired us to examine how DAGKNN succeeds in this classification problem compared to DAGSVM. Generally, in all groups the linear, quadratic and RBF kernel functions and from these kernels especially the quadratic and RBF showed their power in this classification task. DAGKNN method did not manage to obtain higher accuracies than DAGSVM, but it is still a very comparable classification method since the simplicity of k-NN compared to SVM is from the pratical and computational point of view much more user-friendly. The DAGKNN method contains only two parameters: the choice of distance and k value. In SVM the choice of a kernel function and the tuning of the parameters are the key factors for successful classification. How to find the right kernel and the right parameter values can be computationally demanding problem as Table 6 showed. How to speed up the parameter tuning and how to choose the right kernel functions are problems to be researched more closely in the future. Also, we need to examine how other machine learning methods such as Linear Discriminant Analysis or Naïve Bayes manage in benthic macroinvertebrate classification when using the DDAG learning structure. Furthermore, other multi-class methods of SVMs and classification methods need to be considered in the further research. Feature selection is an important factor in classification problems. In this paper we solved this problem by using the scatter method [8] which is a novel approach. From the results we can conclude that the scatter method is a valid feature selection method for the classification of benthic macroinvertebrates. Results also showed that the fully automated benthic macroinvertebrate identification is possible when the classifiers are tuned up.

Acknowledgments. We thank Finnish Environment Institute, Jyväskylä, Finland, for the data. The first author is also thankful to the Tampere Graduate Program in Information Science and Engineering for the support. We also want to thank Markku Siermala, Ph.D., for the scatter method.

References

1. Christiani, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press, Cambridge (2000)
2. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20, 273–297 (1995)
3. Cover, T.M.: Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE Transactions on Electronic Computers EC-14 (1965)

4. ImageJ: public domain Java-based image processing program, http://rsbweb.nih.gov/ij/docs/index.html
5. Joutsijoki, H., Juhola, M.: Automated benthic macroinvertebrate identification with decision acyclic graph support vector machines. In: Proceedings of the 2nd IASTED International Conference Computational Bioscience (CompBio 2011), Cambridge, United Kingdom, pp. 323–328 (2011)
6. Joutsijoki, H., Juhola, M.: Comparing the one-vs-one and one-vs-all methods in benthic macroinvertebrate image classification. In: Perner, P. (ed.) MLDM 2011. LNCS (LNAI), vol. 6871, pp. 399–413. Springer, Heidelberg (2011)
7. Joutsijoki, H., Juhola, M.: Kernel selection and its consequence to the number of ties in majority voting method. Artificial Intelligence Review (in press), doi:10.1007/s10462-011-9281-3
8. Juhola, M., Siermala, M.: A scatter method for data and variable importance evaluation. Integrated Computer-Aided Engineering 19(2), 137–149 (2012)
9. Kahsay, L., Schwenker, F., Palm, G.: Comparison of Multiclass SVM Decomposition Schemes for Visual Object Recognition. In: Kropatsch, W., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, pp. 334–341. Springer, Heidelberg (2005)
10. Kiranyaz, S., Gabbouj, M., Pulkkinen, J., Ince, T., Meissner, K.: Classification and Retrieval on Macroinvertebrate Image Databases using Evolutionary RBF Neural Networks. In: Proceedings of the International Workshop on Advanced Image Technology, Kuala Lumpur, Malaysia (2010)
11. Kiranyaz, S., Ince, T., Pulkkinen, J., Gabbouj, M., Ärje, J., Kärkkäinen, S., Tirronen, V., Juhola, M., Turpeinen, T., Meissner, K.: Classification and retrieval on macroinvertebrate image databases. Computers in Biology and Medicine 41(7), 463–472 (2011)
12. Kiranyaz, S., Gabbouj, M., Pulkkinen, J., Ince, T., Meissner, K.: Network of evolutionary binary classifiers for classification and retrieval in macroinvertebrate databases. In: Proceedings of 2010 IEEE 17th International Conference on Image Processing (ICIP), pp. 2257–2260 (2010)
13. Larios, N., Deng, H., Zhang, W., Sarpola, M., Yuen, J., Paasch, R., Moldenke, A., Lytle, D.A., Correa, S.R., Mortensen, E.N., Shapiro, L.G., Dietterich, T.G.: Automated insect identification through concatenated histograms of local appearance features: feature vector generation and region detection for deformable objects. Machine Vision and Applications 19(2), 105–123 (2008)
14. Lytle, D.A., Martinez-Muñoz, G., Zhang, W., Larios, N., Shapiro, L., Paasch, R., Moldenke, A., Mortensen, E.N., Todorovic, S., Dietterich, T.G.: Automated processing and identification of benthic invertebrate samples. Journal of North American Benthological Society 29(3), 867–874 (2010)
15. Platt, J.C., Christiani, N., Shawe-Taylor, J.: Large margin dags for multiclass classification. In: Advances in Neural Information Processing Systems (NIPS 1999) 12, Denver, USA, pp. 547–553 (2000)
16. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. Neural Processing Letters 9, 293–300 (1999)
17. Tirronen, V., Caponio, A., Haanpää, T., Meissner, K.: Multiple Order Gradient Feature for Macro-Invertebrate Identification Using Support Vector Machines. In: Kolehmainen, M., Toivanen, P., Beliczynski, B. (eds.) ICANNGA 2009. LNCS, vol. 5495, pp. 489–497. Springer, Heidelberg (2009)
18. Ärje, J., Kärkkäinen, S., Meissner, K., Turpeinen, T.: Statistical classification and proportion estimation - an application to macroinvertebrate image database. In: 2010 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010), Kittilä, Finland, pp. 373–378 (2010)