Specification and assessment of methods supporting the development of neural networks in medicine

Michael Egmont-Petersen

Specification and assessment of methods supporting the development of neural networks in medicine

Shaker Publishing
Maastricht 1996

ISBN 90-423-0002-7

Shaker Publishing B.V., St. Maartenslaan 26, 6221 AX Maastricht
tel. 043-3260500 - fax. 043-3255090

Universiteit Maastricht

Specification and assessment of methods supporting the development of neural networks in medicine

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Universiteit Maastricht, op gezag van de Rector magnificus, Prof. mr. M.J. Cohen, volgens het besluit van het College van Decanen, in het openbaar te verdedigen op donderdag 19 december 1996 om 12.00

door

Michael Egmont-Petersen

Geboren te Charlottenlund, Denemarken, op 17 maart 1967

Promotor:	Prof. dr.ir. A. Hasman
Co-promotor:	Dr.ir. J.L. Talmon
Beoordelingscommissie:	Prof. dr. H.J van den Herik (voorzitter)
	Dr. P. Braspenning
	Dr. W.R.M. Dassen
	Prof. dr. E.S. Gelsema (Erasmus Universiteit Rotterdam)
	Prof. Dr. Dipl. Math. R. Repges (Rheinisch Westfälische Technische Hochschule Aachen)

Het verschijnen van dit proefschrift werd mede mogelijk gemaakt door financiële steun van:

- SMS Cendata

To my parents

Table of contents

Preface

This book is submitted as doctoral dissertation to the Faculty of Medicine at Maastricht University in the Netherlands. Written as research monograph, it is assumed that the reader is familiar with the basic theories of feed-forward neural networks, multivariate statistics, statistical discriminant analysis, maximum-likelihood estimation, linear algebra and basic mathematical analysis. In the dissertation, the following four topics are addressed:

- Quality assessment of neural-net classifiers (chapter 2).
- Attribute/feature selection for (neural-net) classifiers (chapter 3 and 4).
- Estimation of missing values (chapter 5).
- Robustness of classification from noisy measurements (chapter 6).

The book is organized as follows. In the introductory chapter, the background of the dissertation is sketched. The areas of medical decision making, neural networks and knowledge-based systems are briefly considered. The four research topics are motivated. The chapters 2 to 6 are written as independent manuscripts and can be read as such. The reader unfamiliar with feature selection should read chapter 3 before chapter 4 as this topic is introduced in chapter 3. Chapter 5 can be read completely independent of the other chapters. Chapter 6 refers briefly to a notation that is introduced in chapter 4. Except from this, chapter 6 can also be read independently from the other chapters. Chapter 7 contains a general synthesis of the conclusions of the chapters 2 to 6. It also discusses which research questions have been answered and topics for further research are suggested. Eventually, it is considered in to which extent neural networks are likely to be used in clinical practice. The dissertation contains summaries in English, Dutch, German and Danish.

Acknowledgements

Many persons as well as sponsors have contributed to this work. First, I want to thank Jan Talmon for offering me the possibility to join the Department of Medical Informatics in Maastricht to finish my doctoral dissertation. Not only did the possibility of submitting my dissertation in Maastricht mean a crucial change of my situation, it also gave me a possibility to migrate to the Netherlands. I admire Jan's many ideas and his ability to always look at a manuscript with new eyes even though many iterations have already been made. I want to thank Arie Hasman for the possibility of submitting my dissertation with him as promotor. Apart from Arie's critical comments, especially his assistance during discussions between Jan and me

was worshipped. At the Department of Medical Informatics, I want also to thank Margot Hijnens for cheering me up and her help in copying and sending the manuscripts. Ton Ambergen helped me much with multivariate statistics and mathematics. He introduced me to the theory of the Bayesian classifier which is used as theoretical model in the chapters 3, 4 and 6. I wish also to thank Erich Pelikan, Frank Vogelsang and Ingrid Scholl for using the quality measures and giving me feed-back. Erich and I had many fruitful discussions along the way and he inspired me to work with feature assessment. Thanks to Jytte Brender and Peter McNair for many inspiring discussions round quality assessment. Thanks also to my present employer Department of Biophysics. Theo Arts sent me home to work solely with the dissertation 2½ weeks before the deadline of the manuscript; this was not only a good idea but also necessary. Thanks to my colleagues at the Department for cheering me up. Finally, I want to thank especially my family for their support throughout the process. Thanks also to my friends Anne-Marie Langsig, Mads Fischer-Rasmussen, Judith Westen and Esther van Steenbergen.

Finally, thanks to Erich Pelikan for translating the summary into German and to Miriam Hueber for helping me with the dutch summary.

The work was supported financially by the Bursary EF-348 of the Danish Academy of the Technical Sciences (ATV), the commission of the European Communities through the AIM-projet KAVAS-2, A 2019 and the Department of Medical Informatics.

Introduction 1

"Wir müssen wissen. Wir werden wissen."

David Hilbert
Königsberg 1930.

1 Introduction

This dissertation introduces methods and techniques that can aid development and validation of neural networks for medical applications. These methods and techniques give insight into the properties of a feed-forward neural network that is trained to perform a classification task. Topics such as quality assessment, feature assessment and estimation of missing values are addressed. The techniques presented cannot solely support development of neural networks for medical applications; they may aid development of neural networks for other application domains as well.

1.1 Medical decision making

Physicians make many decisions in every-day practice. Important clinical decisions are – among others – establishment of a diagnosis, prescribing a therapy and monitoring a patient [67]. The consequences of the decisions have a large range of variety; prescribing a cough mixture has less consequences for the patient than transplanting a heart.

Decision making is drawing a conclusion from information about the state of the world and, possibly, initiating one or more actions. Compared to decisions in other disciplines such as economy and engineering, medical decisions are often based on data with various degrees of certainty and vagueness [67]. Prior to establishing a diagnosis, the physician may gather information by investigating the medical history, perform a physical examination, request laboratory tests or perform other specific investigations. Some signs and symptoms are "soft" data whereas others such as radiographs and laboratory results constitute "hard" facts. The medical decision maker must often select out of a vast number of alternative information sources that are ranked according to ethical, utilitarian (quality of life) and economical criteria.

A survey among different clinical departments in a Dutch hospital showed that departments working with hard data, e.g., radiology, cardiology and neurology departments, were more willing to use software to support diagnostic and/or therapeutic tasks than other clinical departments [30]. Also in clinical laboratories, electronic data processing helps scheduling tests and calibrating equipment [47].

Since the MYCIN program was developed at Stanford University [58] many researchers in the field of medical informatics have considered also the possibility of developing and using clinical decision support systems. However, as reported in the literature [29,73], only a few such systems are used in clinical practice.

There are considerable differences between on the one hand facilitating automatic image and signal processing – e.g. for CT-imaging and EMG-analysis – and on the other hand performing clinical decision making with knowledge-based systems. We mention a few of them:

- **The amounts and types of data dealt with.** A program for segmentation of digital radiographs works on the grey levels of the pixels in an image. A knowledge-based system for diagnosing abdominal pain asks the user to provide various symptoms.

- **The abstraction level of the "knowledge" embodied in the algorithm.** The segmentation program may use some statistical criterion to decide which pixels form part of a pathologic structure whereas a knowledge-based system that proposes a diagnosis based on a number of symptoms uses inference rules provided by clinical experts.
- **The possible consequences of a wrong decision.** Wrongly classifying a few pixels will have less consequences than proposing a wrong diagnosis to a patient.

Neural networks have been proposed to support clinical decision making (see section 1.4). Apart from performing stand-alone classification tasks, neural networks can be incorporated as part of (larger) knowledge-based systems, see for example [31].

1.2 Neural networks

Neural networks were first proposed during the second world war. McCulloch and Pitts introduced a mathematical model for a typical neuron in the cerebral cortex [37]. Their artificial neuron works like a junction between "nerve" paths. Each path provides the neuron with a numerical input value. The neuron uses a weight to modify each input signal it receives. The size of the weight determines whether the input is *amplified* or *reduced*, the sign whether the input is *exhibitory* or *inhibitory*. The neuron produces an activation that is functionally dependent on the input signals it receives. It sends to other neurons the output 1 if its activation is positive and the output 0 otherwise. By organizing such neurons in clusters, a mapping from input (stimuli) to output (response) is performed. McCulloch and Pitts proved that when the weights have the appropriate values, a synchronously updated cluster of neurons is capable of universal computation.

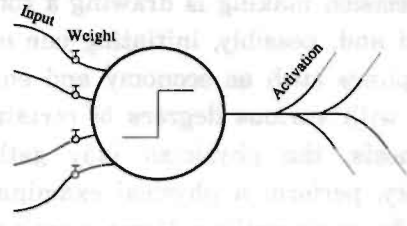


Figure 1. McCulloch and Pitts' artificial neuron.

In 1949, Hebb introduced a theory of how learning takes place in the brain [25]. His theory has later formalized into the so-called Hebb rule, a formula for adjusting the weights in a cluster of artificial neurons. Hebb's learning rule was in 1951 used to train a hard-wired learning machine built by Minsky which could simulate a (biological) neuronal network [39]. The behaviour and computational capabilities of clusters of neurons were extensively studied in the fifties by, among others, the group around Rosenblatt. In 1957, Rosenblatt introduced the notion of the *perceptron*, a cluster of neurons organized into layers [52]. The nodes in each layer are connected to the nodes in the preceding and successive layer. A so-called *simple* perceptron consists of only an input and an output layer. In a simple perceptron, the activation of the output neurons is a sum of the weighted input values. Similar to McCulloch and Pitts' neurons, an output neuron in a perceptron produces either 0 or

1. Rosenblatt developed a perceptron learning rule to adjust the weights such that the perceptron is able to perform a classification task. He was able to prove that his learning rule converged to a stable state. Widrow extended the perceptron learning rule to neurons with continuous outputs [68]. Such a perceptron is closely related to discriminant functions as published by Fischer in 1936 [18].

An important event that impeded research in neural networks for almost 20 years was the proof by Minsky and Papert that simple perceptrons (with no hidden layer) cannot discern classes that are not linearly separable [40]. The most famous example is the XOR function. In 1985, Le Cun proposed an algorithm for training networks composed of layers of nonlinear nodes [9]. In 1986, Rumelhart, McClelland and Williams (all members of the so-called Parallel Distributed Processing research group) proposed a similar learning algorithm for training networks organized in two or more layers of nonlinear nodes. They showed that a Multi-Layer Perceptron (MLP) could learn the XOR classification task [53]. Their training algorithm, called Back-propagation or the Generalized Delta Rule, has obtained widespread use.

1.3 Knowledge-based systems

Research in artificial intelligence grew rapidly during the eighties. The research agenda included questions on the nature of intelligence and on how to build systems that perform like an expert. Programming languages such as LISP or PROLOG and integrated shells such as KEE or NEXPERT were used to develop Knowledge-Based Systems (KBS). In the second half of the eighties, theoretical and practical obstacles showed that the development of a KBS was more complicated than first believed. In many domains, it is an enormous task to elicit and formalize the expertise of a domain expert; this has been called the *knowledge-elicitation bottleneck*. One has to develop a domain model to characterize the relevant *structural* knowledge (the system must know *what* it is talking about). Then, one has to develop a reasoning model that may contain *procedural* knowledge and *inference rules* (the system must know *how* to achieve its goal and it should know *why* a specific solution was obtained). Having built a KBS, the largest task still remains: to validate the system and its knowledge by more than one expert [7,16,72,73].

The knowledge-elicitation bottleneck and the problem of evaluating knowledge-based systems once they were developed caused discouragement. Firms that since the beginning of the eighties had invested in development of KBS stopped funding such projects. The introduction of nonlinear neural networks by the PDP research group and the widespread use of machine learning algorithms such as ID3 [49] and NPPA [59] gave the interest in developing knowledge-based systems a new push. Development of a "traditional" KBS requires an expert to verbalize his "know-how". In contrast, neural networks and machine-learning algorithms analyze the expert's "show-how" as implicit in a set of sample cases. When the essential information in the cases can be formalized, it is relatively simple to train, for example, a neural network to perform the classification task. Development is reduced from years to months.

1.4 Neural networks in medicine

Various statistical techniques for supporting (medical) decision making are available. Two such techniques are *multivariate discriminant analysis* and *logistic regression*, which are both maximum-likelihood based approaches. In contrast, neural networks – originally developed as a cognitive simulation model – have a weaker theoretical basis as they are not related to a specific parametric distribution. Comparative studies have shown that neural networks can sometimes outperform other classifiers including k-nearest neighbour, discriminant analysis, logistic regression and decision trees, especially when classification relies on continuous attributes [4,5,23,28,32,43, 45,46,57,69,74]. One reason for their good performance is that a sufficiently complex MLP with one hidden layer can implement any arbitrary discriminant [63]. Such a neural network does not rely on a specific parametric model of the distribution of the attributes. Furthermore, the output values of an MLP approach the Bayesian a posteriori probabilities when the topology of the MLP is complex enough and the training set sufficiently large [28,38,51]. Hence, an MLP is a maximum-likelihood classifier. These recent theoretical results and the fact that MLPs occasionally outperform other types of classifiers, make it worthwhile to consider application of neural networks for medical classification tasks.

Different types of neural networks have been developed for a large number of medical classification and prediction tasks. We performed a literature survey to make an inventory of different medical applications of neural networks. Most articles were obtained through a search in MEDLINE. We searched for articles published in peer reviewed journals in 1994 or 1995. Our inclusion criterion was that a neural network was designed to solve a clinical decision problem.

We discerned decision problems where a system directly suggests a diagnosis from problems in which the developed system provided information on which the clinician can base his/her decision:

- The first category includes neural networks that have been trained to associate one or more diagnoses with (clinical) findings in the domains of occupational medicine and general practice [1], to locate neurological disorders [6], to detect atrial fibrillation from ECGs [8], to suggest a diagnosis based on ECG-signals [10,34,74], to diagnose coronary artery disorders [12], to diagnose parathyroid adenoma [15], to diagnose malignant melanoma from images [17], to diagnose myocardial infarction [21], to decide whether to send a patient to a cardiac care unit [22], to diagnose neuro-muscular diseases based on the EMG [33], to establish the diagnosis of patients with dementia [42], to diagnose patients with hypertension and to propose therapies [48], to diagnose thyroid disorders [54], to diagnose cerebral tumours from cytological findings [55] and to diagnose pulmonary embolism from ventilation-perfusion lung scans and thorax radiographs [61].

- The second category of applications contain neural networks trained for analysis of MR images [2], screening cervical smears for possible carcinoma

[3], screening blood samples for possible cancer [13], segmentation of radiographs [14,45,46,50,64], classification of the macro motor unit potential [23, 24,56], identification of faces with morphological syndromes [26], monitoring fetal CTG and suggesting which actions to take [31], predicting the survival rate of patients with malignant melanoma [35] and cardiovascular diseases [66], analysis of EEG-signals [36], detection of cancer cells [41], segmentation of CT-images [44], classification of protein chains [62] and analysis of biopsies [65].

In none of the reviewed articles, it is reported that neural networks were used in *clinical practice*. In one case, a neural network is embedded in a knowledge-based system which was to undergo a multi-centre evaluation study [31]. In most publications, however, application in clinical routine is not even considered, though potential application of neural networks for screening [3,13,61], education [50,55] or differential diagnosis [17,55] is occasionally suggested. In a few cases, an MLP is used as a tool in medical research, see for example [17].

In some studies, the developed networks are thoroughly evaluated using large test sets [17,24]. Occasionally, their performance is compared with that of clinicians [3, 31,61]. Although only a few authors mention that it is difficult to verify and validate a neural network [10,11,22,31], thorough evaluation remains a crucial part of developing a classification/decision support system as wrong decisions may have fatal consequences.

The most thorough validation of a neural network in the literature referred above was performed by Keith and Greene [31]. Their neural network decides whether the cardiotocogram shows a deceleration, which is a dichotomous classification task. The performance of the network was compared on a two-by-two basis with six different human decision makers. In addition, the inter- and intra-observer variation of the six human decision makers were assessed.

2 Development, evaluation and application of knowledge-based systems

It is difficult to develop a knowledge-based system that can perform a clinical decision task as well as a human expert. The literature on the development of KBS shows that not only knowledge elicitation but also evaluation is complex. Recently, many topics related to evaluation and assessment of information technology in medicine – including KBS – have been discussed in [19]. Clarke *et al.* state about development and evaluation of KBS:

"...both development and evaluation of knowledge-based systems should be iterative with continuous cycles of development and evaluation taking place" [7].

Development of a KBS will normally begin with a description of the system's concept and scope, its basic requirements. Then follow four major phases that may run partially in parallel [7]:

- Early prototype development
- Evaluation of the system's validity
- Evaluation of the system's functionality
- Evaluation of the system's impact

Normally, one distinguishes *verification* and *validation* in the evaluation of the system's validity. Among the many different definitions, we adopt those from Hoppe and Meseguer [27]:

Verification Verification checks a KBS against the specifications generated by its totally formalizable requirements. It is performed by checking and not by proving KBS properties. It should be considered during the whole KBS life cycle.

Validation Validation assures that the final KBS complies with user needs and requirements. Validity can be achieved only partially by verification, since some informal requirements can only be partially formalized.

In traditional rule-based systems, the expert knowledge is explicitly represented. Although this is not a trivial task, the developers and experts can verify and validate the inference rules, the reasoning mechanism, the procedural knowledge and the performance of the total system.

By virtue of its representation, the knowledge encoded by a neural network cannot be validated as in a traditional rule-based system. The validation problem is the Achilles heel of neural networks. It is, however, to some extent possible to evaluate a trained neural network as properties like sensitivity, specificity, efficiency and predictive value [20] can be computed and compared with the requirements. For some medical classification tasks such as image segmentation or analysis of EMG-signals, knowing that a classifier has a high performance may be sufficient for the clinical user. In other clinical applications, it is imperative to validate also the knowledge encoded by the classifier. A relevant question can be which inputs (signs and symptoms) contribute to the classification [11,22,47]. There is a need for new methods and tools for validation of neural networks if they are to be used in more clinical applications. Not only end-users, but certainly also developers can benefit from insight into properties of the classification model they are currently building. We believe that methods that support assessment of neural-net classifiers for clinical decision making will make these classification models more attractive to a clinical user as well as aid the development of a neural network.

Even when a neural-net classifier performs well in a laboratory setting, its application in practice may be hampered by practical constraints. In the clinic, one or more attributes may be missing. Neural-net classifiers can only be used when all inputs are available, so incomplete cases impedes the use of a neural network.

When the network is to be applied, the user may wish to gain insight into certain results that are obtained for the case at hand. He might also want to establish to which extent the result depends on statistical fluctuations in the input data.

3 Objectives of our study

The major objective of this dissertation is to *explore methods and techniques that can expedite application of neural networks in medicine*. This objective is divided into four topics that are investigated throughout the dissertation. All four topics are closely related to development and assessment of neural-net classifiers for clinical application and comprise both theoretical and practical questions. The first topic is how to measure the quality of neural-net classifiers:

- Which existing metrics are suited to assessing the performance of (neural-net) classifiers? and what are their theoretical underpinnings?
- Are other properties of a neural-net classifier interesting in a clinical setting and how can such properties be measured?
- Do they provide new insight into the quality of a (neural-net) classifier?

The performance of a classifier is an aggregated measure for the ability of the attributes to discriminate the classes. Often, one also wants to identify how much the individual attributes contribute to the classification task. This brings us to the next topic, the relation between the input and output of neural-net classifiers:

- Is it possible to quantify how much information an attribute provides to a classifier?
- Can attributes easily be removed from a classifier and what is the resulting decrease in performance?
- When is an attribute relevant for the classification of a case?
- Which attributes contribute most to the current classification?

Two more topics, related to the applicability of neural-net classifiers are investigated. One is how to cope with missing data; the other regards the credibility in a classification in relation to measurement noise.

It is often not possible to obtain all information that could influence a decision. As all inputs are required to evaluate a neural net, clinical application of neural networks will be impeded by incomplete information.

- Is it possible to cope with missing data by imputation? What are the consequences for the performance of the classifier?

Some attributes are measured under uncertainty as noise is inherent in the measurement process.

- What is the credibility of the classification in relation to attributes that are measured under uncertainty?

Chapter 2 addresses quality assessment of neural-net classifiers. Existing metrics for performance assessment are reviewed and some new metrics are suggested. How to select the best neural network classifier based on different performance measures, is briefly considered. Chapter 3 addresses how to assess which attributes are important for the classification of a case. A method to estimate the relative importance of attributes for a neural-net classifier is developed. In chapter 4, the influence of an attribute is analyzed for a Bayesian classifier. Based on this analysis, measures are defined to compute the maximal decrease in performance when an attribute is removed from the classifier. Also a method to prune attributes is developed. In chapter 5, an approach to estimate missing values is suggested. A neural network is used to predict missing values in cases with different combinations of missing and observed values. Chapter 6 addresses the influence of measurement noise on the credibility of the classification. A novel measure is suggested to estimate the effect on the classification result when noisy attributes are remeasured. In chapter 7, the four research topics are discussed, directions for future research are given and the results of the dissertation placed in a perspective.

References

[1] C.-F. Bassøe. "Automated diagnoses from clinical narratives: A medical system based on computerized medical records, natural language processing, and neural network technology", *Neural networks*, Vol. 8, No. 2, pp. 313-319, 1995.

[2] P. Blonda, A. Carella, R. DeBlasi, F. Dicuonzo, V. LaForgia, D. Miella, G. Pasquariello, G. Satalino. "A neural network modular system for object classification in brain MR images", In: *AIME-93*, S. Andreassen, R. Engelbrecht, J. Wyatt (Eds.), IOS Press, Amsterdam, pp. 477-480, 1993.

[3] M.E. Boon, L.P. Kok, M. Nygaard-Nielsen, K. Holm, B. Holund. "Neural network processing of cervical smears can lead to a decrease in diagnostic variability and an increase in screening efficacy: A study of 63 false-negative smears", *Modern pathology*, Vol. 7, No. 9, pp. 957-961, 1994.

[4] D.E. Brown, V. Corruble, C.L. Pittard. "A comparison of decision tree classifiers with backpropagation neural networks for multimodal classification problems", *Pattern recognition*, Vol. 26, No. 6, pp. 953-961, 1993.

[5] C.H. Chen. "On the relationships between statistical pattern recognition and artificial neural networks", *International journal of pattern recognition and artificial intelligence*, Vol. 5, No. 4, pp. 655-661, 1991.

[6] S. Cho, J.A. Reggia. "Multiple disorder diagnosis with adaptive competitive neural networks", *Artificial intelligence in medicine*, Vol. 5, No. 6, pp. 469-487, 1993.

[7] K. Clarke, R. O'Moore, R. Smeets, J. Talmon, J. Brender, P. McNair, P. Nykänen, J. Grimson, B. Barber. "A methodology for evaluation of knowl-

edge-based systems in medicine", *Artificial intelligence in medicine*, Vol. 6. No. 2, pp. 107-121, 1994.

[8] D. Cubanski, D. Cyganski, E.M. Antman, C.L. Feldman. "A neural network system for detection of atrial fibrillation in ambulatory electrocardiograms", *Journal of cardiovascular electrophysiology*, Vol. 5, No. 7, pp. 602-608, 1994.

[9] Y. Le Cun. "Une procedure d'apprentissage pour reseau a seuil assymetrique", In: *Proceedings of Cognitiva 85*, Paris, pp. 599-604, 1985.

[10] W. Dassen, A. Gorgels, R. Mulleneers, V. Karthaus, H.V. Els, J. Talmon. "Development of ECG criteria to diagnose the number of narrowed coronary arteries in rest angina using new self-learning techniques", *Journal of electrocardiology*, Vol. 27 Supplement, pp. 156-160, 1994.

[11] W.R.M. Dassen, R.G.A. Mulleneers. "The value of artificial neural network technologies to develop diagnostic systems in cardiology", *PACE*, Vol. 17, pp. 1672-1675, 1994.

[12] G. Dorffner, G. Porenta. "On using feedforward neural networks for clinical diagnostic tasks", *Artificial intelligence in medicine*, Vol, 6, No. 6, pp. 417-435, 1994.

[13] S. Dwarakanath, C.D. Ferris, J.W. Pierre, R.O. Asplund, D.L. Curtis. "A neural network approach to the early detection of cancer", *Biomedical sciences instrumentation*, Vol. 30, pp. 94-112, 1994.

[14] M. Egmont-Petersen, J.L. Talmon, E. Pelikan, F. Vogelsang. "Contribution analysis of multi-layer perceptrons. Estimation of the input sources' importance for the classification", In: *Proceedings of pattern recognition in practice IV: Multiple paradigms, comparative studies and hybrid systems*, E.S. Gelsema, L.N. Kanal (Eds.), Elsevier, pp. 347-358, 1994.

[15] A.J. Einstein, J. Barba, P.D. Unger, J. Gil. "Nuclear diffuseness as a measure of texture: definition and application to the computer-assisted diagnosis of parathyroid adenoma and carcinoma", *Journal of microscopy*, Vol. 176, Pt. 2, pp. 158-166, 1994.

[16] R. Engelbrecht, A. Rector, W. Moser. "Verification and validation", In: *Assessment and evaluation of information technologies in medicine*, E.M.S.J. van Gennip, J.L. Talmon (Eds.), IOS Press, Amsterdam, pp. 51-66, 1995.

[17] F. Ercal, A. Chawla, W.V. Stoecker, H.-C. Lee, R.H. Moss. "Neural network diagnosis of malignant melanoma from color images", *IEEE Transactions of biomedical engineering*, Vol. 41, No. 9, pp. 837-845, 1994.

[18] R.A. Fischer. "The use of multiple measurements in taxonomic problems", *Annals of eugenics*, Vol. 7, pp. 179-188, 1936.

[19] E.M.S.J. van Gennip, J.L. Talmon (Eds.). *Assessment and evaluation of information technologies in medicine*, IOS Press, Amsterdam, 1995.

[20] W. Gerhardt, H. Keller. "Evaluation of test data from clinical studies". *Scandinavian journal for clinical laboratory investigation*, Vol. 46, Suppl. 181, pp. 1-74, 1986.

[21] R.F. Harrison, S.J. Marshall, R.L. Kennedy. "A connectionist aid to the early diagnosis of myocardial infarction", In: *AIME-91*, M. Stefanelli, A. Hasman, M. Fieschi, J. Talmon (Eds.), Springer-Verlag, Berlin, pp. 119-128, 1991.

[22] A. Hart, J. Wyatt. "Connectionist models in medicine: an investigation of their potential". In: *AIME-89*, J. Hunter, J. Cookson, J. Wyatt (Eds.), Springer-Verlag, Heidelberg, pp. 115-124, 1989.

[23] M.H. Hassoun, C. Wang, R. Spitzer. "NNERVE: Neural network extraction of repetitive vectors for electromyography-Part I: Algorithm", *IEEE Transactions on biomedical engineering*, Vol. 41, No. 11, pp. 1039-1052, 1994.

[24] M.H. Hassoun, C. Wang, R. Spitzer. "NNERVE: Neural network extraction of repetitive vectors for electromyography-Part II: Performance analysis", *IEEE Transactions on biomedical engineering*, Vol. 41, No. 11, pp. 1053-1061, 1994.

[25] D.O. Hebb. *The organization of behaviour*, John Wiley & Sons, New York, 1949.

[26] R. Herpers, H. Rodax, G. Sommer. "A neural network identifies faces with morphological syndromes", In: *AIME-93*, S. Andreassen, R. Engelbrecht, J. Wyatt (Eds.), IOS Press, Amsterdam, pp. 481-485, 1993.

[27] T. Hoppe, P. Meseguer. "VVT Terminology: A proposal", *IEEE Expert*, June, pp. 48-55, 1993.

[28] G. Hripcsak. "Using connectionistic modules for decision support", *Methods of information in medicine*, Vol. 29, No. 3, pp. 167-181, 1990.

[29] M.E. Johnston, K.B. Langton, R.B. Haynes, A. Mathieu. "Effects of computer-based clinical decision support systems on clinician performance and patient outcome. A critical appraisal of research", *Annals of internal medicine*, Vol. 120, pp. 135-142, 1994.

[30] D. Joones. *Gebruik van informatietechnologie in de gezondheidszorg*, Technical report of the SURF foundation, Amsterdam, 1995.

[31] R.D.F. Keith, K.R. Greene. "Development, evaluation and validation of an intelligent system for the management of labour", *Baillière's clinical obstetrics and gynaecology*, Vol. 8, No. 3, pp. 583-605, 1994.

[32] E.-K. Kim, J.-T. Wu, S. Tamura, Y. Sato, R. Close, H. Taketani, H. Kawai, M. Inoue, K. Ono. "Comparison of neural network and K-NN classification methods in vowel and patellar subluxation image recognitions", *International journal of pattern recognition and artificial intelligence*, Vol. 7, No. 4, pp. 775-782, 1993.

[33] N. Kumaravel, V. Kavitha. "Automatic diagnosis of neuro-muscular diseases using a neural network", *Biomedical sciences instrumentation*, Vol. 30, pp. 245-250, 1994.

[34] S.Y. Kung, J.S. Taur. "Hierarchical perceptron (HiPer) networks for signal/image classifications", In: *Proceedings of the 1992 IEEE workshop on neural networks for signal processing*, S.Y. Kung, F. Fallside, J.A. Sorenson and C.A. Kaufmann (Eds.), IEEE, Piscataway, N.J., pp. 423-435, 1992.

[35] K. Liestøl, P.K. Andersen, U. Andersen. "Survival analysis and neural nets", *Statistics in medicine*, Vol. 13, pp. 1189-1200, 1994.

[36] N. Masic, G. Pfurtscheller. "Neural network based classification of single-trial EEG data", *Artificial intelligence in medicine*, Vol. 5, pp. 503-513, 1993.

[37] W.S. McCulloch, W. Pitts. "A logical calculus of idea immanent in nervous activity", *Bulletin of mathematical biophysics*, Vol. 5, pp. 115-133, 1943.

[38] Z.H. Michalopoulou. L.W. Nolte, D. Alexandrou. "Performance evaluation of multilayer perceptrons in signal detection and classification", *IEEE Transactions on neural networks*, Vol. 6, No. 2, pp. 381-386, 1995.

[39] M. Minsky. *Neural nets and the brain-model problem*. Unpublished doctoral dissertation, Princeton University, 1954 (adapted from: D.E. Rumelhart, J.L. McClelland. *Parallel distributed processing. Exploration into the microstructure of cognition*, Vol. 1, MIT Press, Cambridge, 1986).

[40] M.L. Minsky, S.A. Papert. *Perceptrons*, MIT Press, Cambridge, 1969.

[41] C. Moallemi. "Classifying cells for cancer diagnosis using neural networks", *IEEE Expert*, December, pp. 8-12, 1991.

[42] B.H. Mulsant. "A neural network as an approach to clinical diagnosis", *M.D. Computing*, Vol. 7, No. 1, pp. 25-36, 1990.

[43] Y. Park. "A comparison of neural net classifiers and linear tree classifiers: their similarities and differences", *Pattern recognition*, Vol. 27, No. 11, pp. 1493-1503, 1994.

[44] R. Pasca. *Valutazione di tessiture nelle immagini attraverso indici statistici e reti neurali*, Master thesis (tesi di laurea), Faculty of Mathematics, Physics and the Natural Sciences, Bari, 1994.

[45] E. Pelikan. *Texturorientierte Segmentierungsmethoden in der medizinischen Bildverarbeitung*, Ph.D. thesis from the Faculty of Mathematics and Natural Sciences at the Technical University of Aachen, Aachen, 1995.

[46] E. Pelikan, F. Vogelsang, B. Schulz, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf. "Röntgenbildsegmentierung durch Topologische Karten oder Multilayer Perzeptron - ein Vergleich", In: *Proceedings of the workshop on digital image processing in medicine (Digitale Bildverarbeitung innerhalb der Medizin)*, Freiburg, 1994.

[47] J.F. Place, A. Truchaud, K. Ozawa, H. Pardue, P. Schnipelsky. "Use of artificial intelligence in analytical systems for the clinical laboratory", *Clinica chemica acta*, Vol. 231, pp. S5-S34, 1994.

[48] R. Poli, S. Cagnoni, R. Livi, G. Coppini, G. Valli. "A neural network expert system for diagnosing and treating hypertension", *IEEE computer*, March, pp. 64-71, 1991.

[49] J.R. Quinlan. "Learning from noisy data". *Proceedings of the third International Machine Learning Workshop*, pp. 58-64, 1983.

[50] W.R. Reinus, A.J. Wilson, B. Kalman, S. Kwasny. "Diagnosis of focal bone lesions using neural networks", *Investigative radiology*, Vol. 29, No. 6, pp. 606-611, 1994.

[51] M.D. Richard, R.P. Lippmann, "Neural network classifiers estimate bayesian a posteriori probabilities". *Neural computation*, Vol. 3, pp. 461-483, MIT Press, 1991.

[52] F. Rosenblatt. *Principles of neurodynamics*, Spartan, New York, 1962.

[53] D.E. Rumelhart, G.E. Hinton, R.J. Williams. "Learning internal representations by error propagation", In: *Parallel distributed processing. Exploration into the microstructure of cognition*, D.E. Rumelhart, J.L. McClelland and the PDP research group, MIT Press, Cambridge, Vol. 1, Chap. 8, p. 318, 1986.

[54] T. Schiøler, W. Grimson, P. Sharpe, M. Egmont-Petersen, G. Momsen, R. O'Moore, P. McNair. "Automatic decision support based on voting by independent decision support systems", *Proceedings of the CCL congress*, Dublin, pp. 58-66, 1992.

[55] G. Sieben, M. Praet, H. Roels, G. Otte, L. Boullart, L. Calliauw. "The development of a decision support system for the pathological diagnosis of human cerebral tumours based on a neural network classifier", *Acta neurochirurgica*, Vol. 129, pp. 193-197, 1994.

[56] C.N. Schizas, C.S. Pattchis, T.S. Schofield, P.R. Fawcett. "Artificial Neural Nets in Computer-Aided Macro Motor Unit Potential Classification". *IEEE engineering in medicine and biology*, September, pp. 31-38, 1990.

[57] J.W. Shawlik, R. J. Mooney, G.G. Towell. "Symbolic and neural learning algorithms: An experimental comparison", *Machine learning*, Vol. 6, pp. 111-143, 1991.

[58] E.H. Shortliffe. *Computer-based medical consultations: MYCIN*, Elsevier, New York, 1976.

[59] J.L. Talmon. "A multiclass nonparametric partitioning algorithm". *Pattern recognition letters*, Vol. 4, pp. 31-38, 1986.

[60] S. Thurim, J.A. Reggia, Y. Peng. "High-specificity neurological localization using a connectionist model", *Artificial intelligence in medicine*, Vol. 6, No. 6, pp. 521-532, 1994.

[61] G.D. Tourassi, C.E. Floyd, H.D. Sostman, R.E. Coleman. "Artificial neural network for diagnosis of acute pulmonary embolism: Effect of case and observer selection", *Radiology*, Vol. 194, pp. 889-893, 1995.

[62] J.R. Vargas, G. Bologna, R.D. Appel, D.F. Hochstrasser, C. Pellegrini. "Classification of protein patterns using neural networks: pixel based versus feature based approach", In: *AIME-93*, S. Andreassen, R. Engelbrecht, J. Wyatt (Eds.), IOS Press, Amsterdam, pp. 455-465, 1993.

[63] J.D. Villiers, B. Bernard. "Backprogation neural nets with one and two hidden layers", *IEEE Transactions on neural networks*, Vol. 4, pp. 136-141, 1992.

[64] F. Vogelsang. *Segmentierung radiologisch dokumentierter fokaler Knochenläsionen auf Basis kontextbezogener Vektoren mit neuronalen Netzwerken*, Master Thesis, Faculty of Informatics, Faculty of Medicine, Technical University of Aachen, Aachen, 1993.

[65] A.V. Wangenheim, G.H. Vince, H. Kolles, M.M. Richter, W. Feiden. "Grading of Gliomas in stereotactic biopsies with neural networks", In: *AIME-93*, S. Andreassen, R. Engelbrecht, J. Wyatt (Eds.), IOS Press, Amsterdam, pp. 486-488, 1993.

[66] T. Waschulzik, K. Quandt, M. Lewis, A. Hörmann, R. Engelbrecht, W. Brauer. "Evaluation of an epidemiological data set as an example of the application of neural networks to the analysis of large medical data sets", In: *AIME-93*, S. Andreassen, R. Engelbrecht, J. Wyatt (Eds.), IOS Press, Amsterdam, pp. 466-476, 1993.

[67] M.C. Weinstein. H.V. Fineberg, A.S. Elstein, H.S. Frazier, D. Neuhauser, R.R. Neutra, B.J. McNeil. *Clinical decision analysis*, W.B. Saunders Company, Philadelphia, 1978.

[68] B. Widrow. "Adaptive samples-data systems, a statistical theory of adaptation", *1959 IRE WESCON Convention Record*, part 4, New York, Institute of radio engineers, 1959.

[69] Y.-C. Wu, K. Doi, M.L. Giger, C.E. Metz, W. Zhang. "Reduction of false positives in computerized detection of lung nodules in chest radiographs using artificial neural networks, discriminant analysis, and a rule-based scheme", *Journal of digital imaging*, Vol. 7, No. 4, pp. 196-207, 1994.

[70] Y.-C. Wu, D.H. Gustafson. "Removing the assumption of conditional independence from Bayesian decision models by using artificial neural networks: Some practical techniques and a case study", *Artificial intelligence in medicine*, Vol. 6, No. 6, pp. 437-454, 1994.

[71] J. Wyatt. "Clinical data systems, part 1: data and medical records", *The lancet*, Vol. 344, Dec. 3, pp. 1543-1547, 1994.

[72] J. Wyatt. "Clinical data systems, part 3: development and evaluation", *The lancet*, Vol. 344, Dec. 17, pp. 1683-1688, 1994.

[73] J. Wyatt, D. Spiegelhalter. "Evaluating medical expert systems: what to test, and how?", In: *Knowledge based systems in medicine: Methods, applications and evaluation*, J. Talmon, J. Fox (Eds.), Springer-Verlag, Lecture notes in medical informatics 47, Berlin, pp. 274-289, 1989.

[74] T.-F. Yang, B. Devine, P.W. Macfarlane. "Use of artificial neural networks within deterministic logic for the computer ECG diagnosis of inferior myocardial infarction", *Journal of electrocardiology*, Vol. 27 Supplement, pp. 188-193, 1994.

On the quality of neural net classifiers

2

Appeared in Artificial Intelligence in Medicine, Vol. 6, No. 5, pp. 359-381, 1994.

Authors: M. Egmont-Petersen, J.L. Talmon, J. Brender, P. NcNair.

On the quality of neural net classifiers

Michael Egmont-Petersen [a], Jan L. Talmon [a,*], Jytte Brender [b], Peter McNair [c]

[a] *Dept. of Medical Informatics, University of Limburg, PO Box 616, 6200 MD Maastricht, The Netherlands*
[b] *Medical Informatics Laboratory Aps, Stengaards Allé 33d, DK-2800 Lyngby, Denmark*
[c] *Dept. of Clinical Chemistry, University of Copenhagen, Hvidovre Hospital, Kettegaard Allé 30, DK-2650 Hvidovre, Denmark*

Received June 1993; revised February 1994

Abstract

This paper describes several concepts and metrics that may be used to assess various aspects of the quality of neural net classifiers. Each concept describes a property that may be taken into account by both designers and users of neural net classifiers when assessing their utility. Besides metrics for assessment of the correctness of classifiers we also introduce metrics that address certain aspects of the misclassifications. We show the applicability of the introduced quality concepts for selection among several neural net classifiers in the domain of thyroid disorders.

Keywords: Neural net classifiers; Quality assessment; Quality metrics; Thyroid disorders

1. Introduction

Neural nets (NNs) are being used for classification/diagnostic tasks in various domains, including medicine [8,18,20,21,23,27,35,36,41]. Various issues play a role during the design of such classifiers. Until now, there hardly have been strict rules for how to design the networks apart from trial and error. It is possible to generate several hundreds of networks based on various settings of design parameters. To find the 'optimal' network for a specific classification task some yardstick or metric is required, which facilitates a comparison of different NNs. This problem may be approached in two different ways. First, it is possible to measure the degree to

* Corresponding author. Email: talmon@mi.rulimburg.nl

which the network has generalized the information contained in the set of training instances. Second, the performance of the network can be measured for another set of data. This process is often called cross validation when the metric used reflects how well the correct class labels are assigned [17,40]. These measures provide only limited insight in the performance of a network.
An additional problem is that what is an optimal network will depend on the situation in which the network will be applied. The requirements regarding the network's performance will be different in a screening situation as compared to its application in a highly specialized clinic.

In this paper, we propose several *quality concepts* and *quality metrics* that facilitate *quality assessment* of NN classifiers. The metrics are used to derive actual *quality measures* for a specific NN and they provide the potential user with means to select the most appropriate network among different nets.

In the following, we will briefly describe the kind of NNs we have been experimenting with. We will discuss issues that play a role in the design of these networks. Next we will introduce a number of quality concepts. Each of these quality concepts describes a *property* of a (neural net) classifier. Most of these quality concepts are general and can be applied to other classifiers as well.
We will define a set of metrics that can be used to measure the extent to which specific *properties* are present or absent. We show their applicability for networks in the domain of thyroid disorders.

2. Neural-net classifiers

An NN classifier consists of a set of interconnected processors. The way the processors are connected and what processes are performed by the interconnected processors determine their properties (see for an overview e.g. [39]). We will restrict ourselves to feed-forward networks that are trained by means of the back-propagation learning mechanism [34]. Feed-forward networks consist of a series of (fully) interconnected layers of processing units called nodes or neurons – see Fig. 1.

The first layer – the input layer – takes as input the various attribute values. The output of the nodes in the input layer, multiplied with the weights attached to the links, is passed to the nodes in the hidden layer.

A hidden node collects the incoming weighted output values of the previous layer. Besides that, it receives also the weighted value of a bias node. This bias node always outputs the value 1. It allows for adding an offset to the sum of the weighted inputs, similar to an offset in a regression or discriminant function.
The sum of the weighted input values is passed through a nonlinear *activation function* (see Fig. 2). Various types of activation functions have been proposed, such as sigmoidal, hyperbolic-tangent or logistic functions. The only requirements are that the output values of the function are bounded to an interval and that the nonlinear function can be differentiated. To avoid saturation of the nonlinear functions during training, the total input activation has to be bounded. This can be

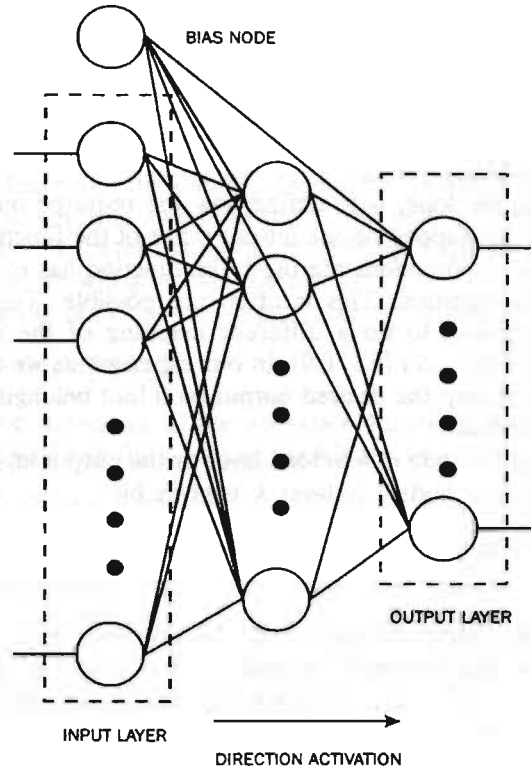


Fig. 1. General structure of a feed-forward network with one hidden layer.

achieved in two ways. One either scales the weights from each input node such that the orders of magnitude of the input value and the weights are reciprocal to each other. The other way is to scale the input values to a certain range. We selected a scaling for each input variable to the interval [0, 1]. By doing so, we can use initial weights of the same order of magnitude throughout the network.

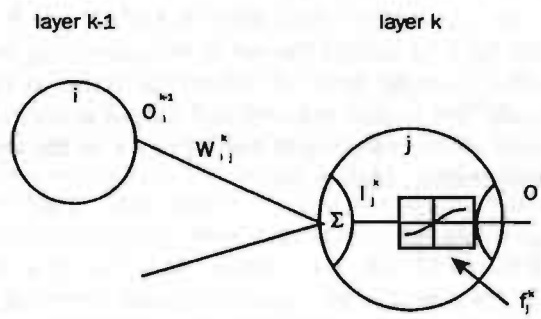


Fig. 2. The structure of a node in the hidden and the output layer.

The output of a node in the hidden layer is fed into the nodes of the next layer, again multiplied with the weights attached to the links. In principle, several hidden layers may be used. However, it has been shown that any arbitrary discriminant function can be built with a classifier with only one hidden layer, if enough hidden nodes are used [25,30]. The nodes of the output layer have the same structure as the nodes of the hidden layer(s).

What remains to be done, is to define how the required output – in our case class membership – is mapped on the output range of the function. When the limit values of the activation functions are used, the function has to be in saturation to achieve a good classification. This is often not possible. Therefore, several researchers have proposed to do a different mapping of the desired values, for example on $[-0.9, 0.9]$ or on $[0.1, 0.9]$. In our experiments we use the hyperbolic-tangent function and map the desired output on 0 (not belonging to the class) and 1 (belonging to the class).

The functioning of a node in a hidden layer or the output layer can be described as follows. The input to node j in layer k is given by

$$I_j^k = \sum_i O_i^{k-1} \times W_{ij}^k \tag{1}$$

where O_i^{k-1} is the output of node i in the previous layer, W_{ij}^k is the weight associated with the link between the nodes i and j and I_j^k is the input to the activation function f_j^k of node j in layer k. The summation over nodes i in (1) also includes the bias node.
The output of the node follows from

$$O_j^k = f_j^k\left(I_j^k\right) \tag{2}$$

To each node in the output layer a class label is assigned. The output value of such a node can be interpreted as the amount of evidence in favor of the corresponding class.

For an optimal network, the weights have to be given values such that for each case the appropriate output is generated. Such a configuration of weight values cannot be set a priori. This has led to the development of learning algorithms that adapt an initial weight configuration by successive passes through a set of learning cases. For each case, the proper class label is known, i.e. it is specified which output node should have as output the value corresponding to 'belonging to the class,' while all others should have as output the value corresponding to 'not belonging to the class'. We denote the required output at output node j for case p as $c_{j,p}$. The measure of the total error for all cases in the learning set, given a certain weight configuration, is given by

$$E = \tfrac{1}{2}\sum_p \sum_j \left(c_{j,p} - O_{j,p}^n\right)^2 \tag{3}$$

with $O_{j,p}^n$ being the output at node j in the output layer for case p.

By backpropagation of the errors in the network from the output layer through the hidden layer(s) to the input layer, the weights in the network are updated according to

$$\Delta_p w_{ij}^k = \eta \delta_{j,p}^k O_{j,p}^k \tag{4}$$

with η being the learning rate and $\delta_{j,p}^k$ dependent on whether one has to update the weights of links entering a node in the output layer or entering a node in one of the hidden layers. We refer the reader to ([34], pp 322–328) for the derivation of the formulas. Here, we will give only the results.
For links entering the nodes in the output layer $\delta_{j,p}^n$ is defined as

$$\delta_{j,p}^n = (c_{j,p} - O_{j,p}^n) f_j^{n'}(I_{j,p}^n) \tag{5}$$

in which $f_j^{n'}$ is the derivative of the activation function of node j in the output layer n. For links entering the nodes in the hidden layers, $\delta_{j,p}^k$ is defined as

$$\delta_{j,p}^k = f_j^{k'}(I_{j,p}^k) \sum_l \delta_{l,p}^{k+1} W_{jl}^{k+1} \tag{6}$$

This learning algorithm is repeatedly applied with the same learning set until some stop criterion is met (percentage of correctly classified cases greater than some threshold, total error, E, less than some threshold, numbers of passes through the learning algorithm greater than some threshold L, etc.). Rather than adapting the weights after processing of each case, the weights in the network were updated after each complete pass through all learning cases, with

$$\Delta w_{ij}^k = \sum_p \Delta_p w_{ij}^k \tag{7}$$

For the design of a feed-forward NN, the following issues play a role:

- *The number of hidden layers and the number of nodes for each hidden layer.* We will restrict ourselves to networks with one hidden layer. How to decide about the number of nodes in the hidden layer, is not straightforward [19,36,37]. It has been argued that the degrees of freedom in the network, defined as the number of weights – including those for the bias terms – has to be less than the number of cases in the learning set [2]. We will restrict ourselves in our experiments to networks that fulfil this requirement.
- *The learning rate η has to be selected.* The learning algorithm searches the space of weight values for an optimal solution, using a gradient descent approach. The learning rate defines the unit step size for ΔW. Therefore, like in all gradient methods, a small value for the learning rate will make the convergence of the network towards a solution slow, whereas a large value may cause the weights in the network to 'jump' back and forth over the appropriate value. We used a small learning rate, taking the slow convergence for granted.
- *The composition and size of the learning set.* As the network learns a mapping between input attribute values and class labels from a set of learning cases, the composition of the learning set plays a role in deriving networks that have

predictive value for new cases. In our experiments we will show that various learning sets, randomly drawn from a larger database, may result in networks with a significantly different performance.

- *The initial weights of the network.* The network has to start with some weight configuration before learning can begin. One often uses a random generator to provide the initial weights. It will be clear that the number of iterations required to achieve a well-performing network will depend on this weight configuration. Furthermore, it is known that the final weight configurations that result from different initial configurations need not to be the same, even when the networks are trained with the same learning set. During learning, the network in general gets stuck in a local minimum. So there is no guarantee that the absolute minimum of the total error is found. However, it has been reported that poor local minima are in practice rarely encountered [22]. To avoid these problems, an algorithm has recently been proposed that provides weight configurations that are closer to the optimal than some randomly generated configuration [29]. In our experiments, we used randomly generated initial weights to study their effect on the performance of the classifiers.
- *The type of nonlinear activation function.* As there is no theoretical argument why one function is better than the other, we have selected more or less arbitrarily a hyperbolic-tangent function in all our experiments.
- *The interpretation of the pattern of output values at the nodes in the output layer.* Most often a 'winner takes it all' approach is used. In this approach the class label belonging to the output node with the highest output is assigned to a case analyzed. This approach has some drawbacks. When all output levels are low, the case has apparently an attribute pattern that is not recognized by the classifier. It seems more useful to leave such a case unclassified. Also when two or more nodes have high output values, it is not clear which class label has to be assigned. A slight variation in the input attribute values may cause a completely different class assignment. For that reason we have selected an approach in which a class label is assigned to a case when one and only one output node has an output that exceeds a (node-dependent) threshold. (In our experiments we have set the threshold for all nodes at 0.5, the range of possible outputs being $[-1..1]$). When this requirement is not fulfilled, the case stays unclassified.

From the above discussion, it is clear that guidelines for how to handle these issues hardly exist. The developer of a neural classifier is left only with the option to generate a large number of neural classifiers using different initial settings and then to select the best one(s).

3. Quality concepts

In the domains of medical informatics, pattern recognition and statistics the quality or performance of classifiers is mostly addressed by means of some measure for the extent to which a classifier assigns the correct class labels [16]. This is, however, only one view on the quality of a classifier. An extensive

literature survey in various domains such as laboratory medicine, computer science and health-care management, revealed many quality concepts. Based on this review, preliminary definitions for these concepts were established [4–6]. From this large set, a number of essential quality concepts have been selected and a few new ones added that are relevant to describe the properties and applicability of NN classifiers. In this paper, we will deal only with a subset of these concepts as metrics have not yet been defined for all of them.

Before we embark on the definition of the quality concepts we need to consider what quality is. Brender defined quality in relation to the validation of information systems as:

"... the degree of fulfilment of the users' expectations" [3].

As the expectations may vary among users, this definition implies that quality cannot be expressed by one quantity. However, it may be represented by a set of quality characteristics (a quality profile).

A SYDPOL working group argued that

"Another aim [of quality assessment of decision support systems] is to emphasize the invisible properties of DSS, often concerned with the system's theoretical basis and prerequisites" [32].

We concluded that quality assessment of both information systems and decision support systems deals with the quantification of properties of the system under analysis. Such properties should assist the potential users in deciding whether the system is of any use in their situation.

An important prerequisite for a useful set of quality concepts is that each concept describes some distinct property of a classifier. This is necessary to characterize a specific classifier and to predict its behaviour for new cases.

Furthermore, the concepts should be generally applicable. When a concept is domain dependent, it will have hardly any value for other applications. Also the independence of the type of classifier analyzed is of value as it will facilitate a comparison of several types of classifiers by means of quality measurements.

It is obvious that the concepts should be interpretable by a (potential) end user of the classifier. When end users cannot interpret a quality concept and measurement, they cannot infer the classifier's appropriateness for their specific situation. Consequently, the users have to know the assumptions on which the metrics rely. For example, the users have to know how the *predictive value* of an outcome of a classifier depends on the prevalence of the classes and what information is necessary to recalculate the predictive value for their own situation.

Most quality concepts currently in use in the assessment of classifiers deal with *success* issues, i.e. they describe how well a classifier is doing on a certain problem. In some domains, like clinical chemistry, one also tries to quantify and to get insight in the *failure* characteristics of the evaluated processes. Examples are concepts like *precision* and *accuracy* for biochemical assays [9,31]. These concepts describe the variability in repeated measurement values from the same sample and how close these values are to the real value, respectively. Knowing what is wrong helps in deciding what to do to correct or improve the situation.

In the following we will briefly describe some of the concepts we developed and

used for the assessment of NN classifiers. More details and additional concepts can be found in [4–6].

3.1. Success quality concepts

There are two *success* quality concepts we will deal with in this paper, viz. *coverage* and *correctness*

Coverage is defined as:

> The extent to which a classifier is able to assign a class label to cases.

It may seem that this concept is introduced mainly because of our approach of interpreting the values of the output nodes of an NN. The concept is, however, equally applicable to other classifiers. An example is a linear discriminant function for which no decision is made when the function value is in a certain range. When the failure of a classifier to assign a class label is quantified, the concept *rejection rate* is often used [10,18].

Correctness is defined as:

> The extent to which a classifier is able to assign the correct class label to the cases that are covered by the classifier.

This is a commonly used concept in assessment of classifiers. Note that in our definition it only pertains to cases that receive a class label by the classifier.

When one is interested in the extent to which a classifier assigns proper class labels in the whole set of cases, a concept named *accordance* can be used. *Accordance* will only be used in our experiments during the learning of NNs as its metric is a composite of the metrics for *coverage* and *correctness*. By using accordance instead of either correctness or coverage, we avoid that the learning algorithm optimizes only correctness by leaving all difficult cases unclassified or optimizes coverage only by assigning a – possibly wrong – class label to each case.

3.2. Failure quality concepts

The failure quality concepts are complementary to the success quality concepts. When a classifier cannot classify all cases and/or misclassifies a certain amount of cases, these failure concepts reveal different causes for the deteriorated performance. There are three concepts that explain why cases remain unclassified: *omittance*, *interference* and *restrictedness*.

Omittance is defined as:

> The extent to which the cases remain unclassified because of missing or invalid input attribute values.

The reason for having *omittance* is not directly related to how the classifier works. *Omittance* reflects the fact that in medicine information is not always available; for example, a test might fail or a certain test may not be performed because it was deemed to be unethical or economically unfeasible to perform.

Interference is defined as:

> The extent to which a classifier cannot assign a class label to a case because there seems to be supportive evidence in the input attributes for more than one class.

For our NNs this occurs when more than one output node has an output value that exceeds the node's threshold (in our case 0.5 for each node). When such a situation occurs, the network is apparently not able to perform a proper mapping between the input values and the output classes. This may occur because the specific input pattern has characteristics of two or more classes or because the network is not able to partition properly the input space. The latter can be a result of either an insufficient number of hidden units or of insufficient training.
Interference is also a useful concept for analysing a classifier with multiple (linear) discriminants. Here, voting schemes are often used to combine the output of the various discriminants. Interference occurs when more than one class gets a relatively large number of votes.
Restrictedness is defined as:

> The extent to which a classifier would leave a case unclassified because it does not seem to resemble any of the descriptions of the available classes.

Restrictedness is just the opposite of *interference*. Here the input pattern of a case is so different from the input patterns of the cases in the learning set that the NN does not contain the mapping for that input pattern to the correct class. Here again, restrictedness is a concept rather specific for NN classifiers.

Also the *correctness* concept has a pair of accompanying concepts that describe the nature of the errors made by the classifier, i.e. *bias* and *dispersion*.
Bias is defined as

> The extent to which a classifier will have a preference to misclassify cases in one or a few categories.

Our definition of *bias* should not be confused with the bias concept in other domains, where it is used to indicate the systematic difference between (the expected value of) an observed value and the target value. The only relation of our concept with the others is the focus on the systematic aspect.
Dispersion is defined as

> the extent to which a classifier evenly distributes the misclassified cases over the categories.

Correctness can be related to the *accuracy* concept as used in e.g. biochemistry. Both express the tendency to provide the correct result. The *bias* and *dispersion* concept are both related to the *precision* concept [31]. *Bias* and *dispersion* as well as *precision* express the variability in the results. Whereas *precision* takes into account all observations, *bias* and *dispersion* are only describing how the misclassified cases are distributed in the contingency table; the table in which co-occurrences of class labels in the database and assigned class labels are tabulated (see Eq. (8)).
Bias in the classification results may be corrected by changing the thresholds that are used to interpret the output values of the nodes in the output layer. There is, however, no guarantee that a higher overall correctness will also be obtained.

3.3. Class conditional quality concepts

So far we have introduced quality concepts that describe the general properties of a classifier. In certain applications and situations, overlooking a certain class

may be costly. In others, one wants perhaps to reduce the misclassifications for one class as it involves a large number of cases. It may also be possible that a user wants to be sure that the assignment of a certain class label has a high likelihood of being correct as the economical and/or ethical costs of further investigations for those cases may be high. To assess the applicability of a classifier for one's own situation, the global quality concepts are insufficient.
For that reason we have also defined the *class conditional* variants of those concepts. The idea is that one does not express the extent to which that quality aspect is present for the whole set of cases but only for those cases that belong to a certain class.
For example, the *class conditional coverage* is defined as

> the extent to which cases from a certain class are assigned a class label by the classifier.

Such class conditional quality concepts are determined by only considering one column or one row in the contingency table.

The idea of class conditional quality concepts is often used in the evaluation of medical classification algorithms. For two-class classifiers *class conditional correctness* is often expressed as the *sensitivity* and *specificity*, [15]. However, these concepts do not take any unclassified cases into account.
Apart from quality concepts, which are conditional on the *class label in the database*, the *correctness*, *bias* and *dispersion* can also be defined conditional on the *class label assigned by the classifier*. The latter type of *class conditional correctness* is also known as the *predictive value of the outcome of a classifier*.

4. Quality metrics

In this section, we will define a set of metrics for measuring the extent to which a classifier has certain properties. *All proposed metrics are based on the information available in a contingency table.* We define a contingency table as follows:

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,c} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,c} \\ \cdots & \cdots & \cdots & \cdots \\ m_{c,1} & m_{c,2} & \cdots & m_{c,c} \\ m_{c+1,1} & m_{c+1,2} & \cdots & m_{c+1,c} \\ m_{c+2,1} & m_{c+2,2} & \cdots & m_{c+2,c} \\ m_{c+3,1} & m_{c+3,2} & \cdots & m_{c+3,c} \end{bmatrix} \tag{8}$$

The last three rows reflect the nonclassified cases, due to *omittance*, *interference* and *restrictedness*, respectively. The elements $m_{i,i}$, $i \leq c$, represent the correctly classified cases for class i.
An example contingency table is given in Table 1. Here the rows $c + 1$, $c + 2$ and $c + 3$ are merged into one row, representing the unclassified cases.

Table 1
An example contingency table for a classifier. The numbers appearing in the shaded cells represent the correctly classified cases

		Database			
Classifier		Class 1	Class 2	Class 3	Marginal
	Class 1	23	3	2	28
	Class 2	8	28	1	37
	Class 3	0	0	26	26
	Non. class.	2	2	4	8
	Marginal	33	33	33	99

The total number of cases that receive a class label, is defined by

$$s = \sum_{i=1}^{c} \sum_{j=1}^{c} m_{i,j} \tag{9}$$

Let the total number of unclassified cases be determined from

$$u = \sum_{i=c+1}^{c+3} \sum_{j=1}^{c} m_{i,j} \tag{10}$$

The coverage of a classifier is easily determined from

$$\Omega = \frac{s}{s+u} \tag{11}$$

The coverage measure Ω follows a binomial distribution because passing $s+u$ cases through a network may be seen as a sequence of $s+u$ trials with two outcomes possible for each case (it either is classified or remains unclassified), and the processing of each case is independent of the processing of other cases.
Omittance, interference and *restrictedness* are expressed as fractions of the unclassified cases:

$$\begin{aligned} \textit{omittance } \iota &= \frac{\sum_{j=1}^{c} m_{c+1,j}}{u} \\ \textit{interference } \phi &= \frac{\sum_{j=1}^{c} m_{c+2,j}}{u} \\ \textit{restrictedness } \psi &= \frac{\sum_{j=1}^{c} m_{c+3,j}}{u} \end{aligned} \tag{12}$$

The class conditional metrics for *coverage, omittance, interference* and *restrictedness* for the various classes in the database are defined in a similar way, by leaving out the summations over j in (9), (10) and (12).

For *correctness* several quality metrics are possible. A simple one is the fraction of correctly classified cases defined as

$$\rho = \frac{\sum_{k=1}^{c} m_{k,k}}{s} \tag{13}$$

Note that this fraction is taken relative to the number of cases that are assigned a class label by the classifier.

Also class conditional *correctness* metrics can be defined. Note that now we can condition the *correctness* on the *true class label* (equivalent to the *sensitivity* and *specificity* concepts) or the *assigned class label* (equivalent to the *predictive value* of a classification). These quality metrics will be denoted as ρ_i^D and ρ_i^C, respectively. [1] Let's define

$$s_i^D = \sum_{k=1}^{c} m_{k,i} \tag{14}$$

and

$$s_i^C = \sum_{k=1}^{c} m_{i,k} \tag{15}$$

Now the class conditional *correctness* metrics are defined as

$$\rho_i^D = \frac{m_{i,i}}{s_i^D} \tag{16}$$

and

$$\rho_i^C = \frac{m_{i,i}}{s_i^C} \tag{17}$$

All metrics defined so far estimate the probability of a binomial distribution. Hence, for any metric – say x – one can compute the standard error of the estimate $\hat{x}$:

$$\sigma_x = \sqrt{\frac{\hat{x}(1-\hat{x})}{n}} \tag{18}$$

with n being the number in the denominator of the formula for the metric. When both $\hat{x}n > 5$ and $(1-\hat{x})n > 5$ the confidence interval for x is given by

$$\hat{x} + \frac{\Phi_{\alpha/2}^2(0.5-\hat{x})}{n} - \Phi_{\alpha/2} \times \sigma_x \leq x \leq \hat{x} + \frac{\Phi_{\alpha/2}^2(0.5-\hat{x})}{n} + \Phi_{\alpha/2} \times \sigma_x \tag{19}$$

with $\Phi_{\alpha/2}$ denoting the value cutting off the area $\alpha/2$ in the upper tail of the

[1] Quality metrics and supporting variables, which are conditional on the true class label, are given the superscript D; those conditional on the assigned class label the superscript C.

standard normal distribution. When both $\hat{x}n > 50$ and $(1-\hat{x})n > 50$ the confidence interval for x can be approximated by

$$\hat{x} - \Phi_{\alpha/2} \times \sigma_x \leq x \leq \hat{x} + \Phi_{\alpha/2} \times \sigma_x \tag{20}$$

Another metric that is often used to quantify the degree of agreement between two observers, but which can also be used to characterize the *correctness* of a classifier, is the kappa (κ) metric [7]. The κ metric determines the degree of agreement exceeding the agreement by chance alone. This metric is defined as

$$\kappa = \frac{\rho - e}{1 - e} \tag{21}$$

with

$$e = \sum_{k=1}^{c} \frac{s_k^D \times s_k^C}{s^2} \tag{22}$$

It has been shown that the interpretation of the κ values is not unproblematic [11,13,42], specifically when the different classes have largely different a priori probabilities. In general, one can say that a κ-value above 0.75 indicates excellent agreement [11]. The standard error for κ is defined in [13,14].

Contrary to the widespread use of the κ-metric is the very sparse use of the class conditional kappas [24,28]. Let's define

$$e_i^D = \frac{s_i^D}{s} \tag{23}$$

and

$$e_i^C = \frac{s_i^C}{s} \tag{24}$$

The term e_i^D denotes the marginal probability that a case belongs to class i. It is used as the expected number of correctly classified cases in row i. The κ-metric. conditional on the assigned class label is given by

$$\kappa_i^C = \frac{\rho_i^C - e_i^D}{1 - e_i^D} \tag{25}$$

A similar formula can be given for the κ-metric conditional on the true class label. The standard error for the conditional κ-metric has been derived as [28]

$$\sigma_{\kappa_i^C} = \sqrt{\frac{\rho_i^C(1-\rho_i^C)}{s(1-e_i^D)^2}} \tag{26}$$

Defining metrics for the *bias* and *dispersion* quality concepts is more difficult. The overall *dispersion* concept can be interpreted as the degree to which misclassifications of true class i into class j are compensated by misclassifications of the true class j into class i. This compensation has to take into account the a priori

class probabilities. When ten cases of class A are classified as B and only one case of class B is classified as class A, then there seems not to be a compensation. However, when there are ten times as many cases in class A as compared to the number of cases in class B, then one can say that the misclassifications compensate each other completely.

To derive a metric for the overall *bias / dispersion* we need to normalize the contingency table such that all columns have the same number of cases. The number of cases belonging to class i is given by

$$R_i = \sum_{k=1}^{c+3} m_{k,i} \tag{27}$$

Define R' as the number of cases in the rarest class:

$$R' = \min\{R_i\}_1^c \tag{28}$$

The elements of the normalized contingency table M' are given by

$$m'_{i,j} = \frac{m_{i,j} \times R'}{R_i} \tag{29}$$

The overall *dispersion* of a classifier is measured as the degree of *symmetry* of the normalized contingency table M'. In [12] it was shown that

$$X = \sum_{j=1}^{c-1} \sum_{i=j+1}^{c} \frac{(m'_{j,i} - m'_{i,j})^2}{m'_{j,i} + m'_{i,j}} \tag{30}$$

follows a χ^2 distribution, provided that the denominator terms in (30) are large enough (at least 1 [33]). The overall *dispersion* is now defined as the likelihood that $m'_{i,j}$ and $m'_{j,i}$, $\forall j \neq i$, are equal:

$$\pi = \chi^2(X, df) \tag{31}$$

with $df = c(c-1)/2$ being the degrees of freedom.
The *bias* metric is defined as

$$\theta = 1 - \pi \tag{32}$$

The *bias* and *dispersion* metrics can also be defined conditional on the class labels. These conditional metrics describe different properties than the overall *bias* and *dispersion*. We assume for a dispersed classifier that if the number of cases in the classes is large enough the distribution or misclassified cases will follow the reduced marginal distribution of cases. If that is not so, there is some systematic tendency in the way the classifier misclassifies cases; the classifier is biased.
The reduced marginal distribution for class i is based on all cases in the database, *excluding the cases belonging to class i* (see Fig. 3). For class i, the reduced marginal probability follows from

$$P_{i,j}^D = \frac{R_j}{\sum_{k=1}^{c} R_k - R_i} \quad \forall j \neq i \tag{33}$$

	Class 1	Class 2	Class 3	Marginal
Class 1				
Class 2				
Class 3				
Unclass.				
Marginal	A	B	C	T
Red. M.		B/(T-A)	C/(T-A)	

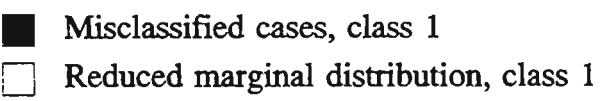


Fig. 3. An illustration of how the reduced marginal distribution of a class relates to the marginal distribution in a contingency table.

The number of misclassified cases among those given class label j is given by

$$n_j^C = \sum_{\substack{k=1 \\ k \neq j}}^{c} m_{j,k} \tag{34}$$

The expected number of misclassified cases with a true class label i follows from

$$\tilde{m}_{j,i} = n_j^C \times P_{i,j}^D \quad \forall i \neq j \tag{35}$$

Now we can compute

$$X_j^C = \sum_{\substack{i=1 \\ i \neq j}}^{c} \frac{\left(m_{j,i} - \tilde{m}_{j,i}\right)^2}{\tilde{m}_{j,i}} \tag{36}$$

X_j^C is χ^2-distributed and the class conditional *dispersion* can be defined as

$$\pi_j^C = \chi^2\left(X_j^C, df\right) \tag{37}$$

with $df = c - 2$. Accordingly, the class conditional *bias* follows from

$$\theta_j^C = 1 - \pi_j^C \tag{38}$$

Furthermore, the sign of

$$m_{j,i} - \tilde{m}_{j,i} \tag{39}$$

gives an indication whether a classifier is biased towards or away from class i when a class label j is erroneously assigned.

5. Experimental results

5.1. Clinical domain

We used a database of biochemical data on patients, who had been tested for thyroid functional disorders, as a basis for a set of experiments. The data were collected between 1981 and 1983 in the department of Clinical Chemistry, Copenhagen University Hospital at Hvidovre, Denmark. In total, the learning database consisted of 20 Hypothyroids (Myxoedema), 50 Hyperthyroids (Thyrotoxic) and 132 Euthyroids (normals) and controls.

In all our experiments we used the following attributes on which the NNs should diagnose the patients:

- the concentration of the triiodothyronine hormone (T3);
- the concentration of the thyroxine hormone (T4);
- the concentration of the thyroxine binding globulin (TBG) and
- the ability of serum to uptake radioactive T3.

In six cases (2 Hyperthyroids, 4 Euthyroids) measurements for TBG and/or T3-uptake were missing, resulting in an overall omittance ι of 3%.

We also had a test set of 174 cases available. In this set ι was 17%. Only 144 cases (12 Myxoedemas, 81 Euthyroids, 51 Thyrotoxics) had all four measurements available. We considered this set sufficiently large to assess the performance of the various networks we had generated from the learning set.

5.2. Factors of variation

As stated before, there are a number of design choices that influence the resulting NN. We have varied the following parameters independently from each other.

- The percentage of cases from the learning set used for training. We used 25, 35 and 50% of the 196 complete cases of the learning set. We grouped our experiments according to these fractions into three experiment series.
- We used for each fraction of cases from the learning set two different levels of *model complexity*, defined as the number of parameters (weights and biases) over the number of learning cases: 0.4 and 0.6 respectively.
- For each topology (defined by the model complexity), we determined three different sets of initial weights, each weight randomly generated in the range of $[-0.5, 0.5]$.
- For each fraction of cases, combined with a certain model complexity and set of initial weights, several subsets of cases were randomly chosen from the learning database. We did not pose any restriction on the class composition of the selected cases.

We considered a network properly trained on a subset of cases from the learning set, when it can classify correctly 96% of the training instances used (*accordance* $\geq$ 96%). Furthermore, we allowed the algorithm to cycle at most 2600

times through the set of selected learning cases. In case the network was unable to classify 96% of the training cases correctly after 2600 cycles, the network was discarded. In each experiment series, we trained 300 networks with an *accordance* $\geq 96\%$, resulting in a total of 1200 trained and converged networks.

These 1200 networks were all tested with the 144 cases of the independent test set. The obtained quality measures were used to test various hypotheses regarding the influence of the design criteria on the performance of the networks. The main conclusions are that

- networks do vary considerably regarding performance and therefore, model selection is *necessary*, and
- the composition of the set of training cases is the only factor from those varied in our experiments that influences performance systematically.

It was also shown that in the selection process, the user has to make a tradeoff between the various aspects of quality. Optimizing one of the quality measures will result in a less optional result for the others.

A few salient results warrant further discussion. First, we analyzed the reason cases could not be classified by an NN. It turned out that in nearly all generated networks the number of cases that could not be classified because of interference (more than one output node had an activation larger than the threshold) exceeded the number of cases due to restrictedness (no output node had an activation larger than the threshold).

Fig. 4 shows the distribution of the number of networks in which a certain number of cases could not be classified due to interference and due to restrictedness. The graph shows the results for the 300 networks that were trained with 50% (98) of the cases in the learning set. In the test set hardly any case could not be classified because it didn't fit the knowledge learned. So the set of training cases seems in general to reflect very well the distribution of the attribute values of the cases in real practice.

The bias/dispersion metrics were used to analyze the extent to which systematic misclassifications were made. It turned out, that both the Hypothyroid and the Hyperthyroid class had only misclassified Euthyroid cases. In almost all networks these two classes are significantly biased towards the Euthyroid class. This was to be expected from the nature of the domain. The three classes (Hypothyroid, Euthyroid and Hyperthyroid) form a kind of continuum, ranging from a decreased functioning to an increased functioning of the thyroid gland. Since the values of the attributes are more or less proportional to the level of functioning of the thyroid gland, misclassifications among the Hypo- and Hyperthyroid classes are very unlikely.

Already in 1977 Devijver showed that within a noisy domain the user must make a tradeoff between correctness and coverage [10]. More recently, a study showed that classifiers derived with various methods from the same database had different properties [35], which justifies the making of a tradeoff between the various quality aspects. For an NN classifier it can be expected that its coverage can be altered by changing the rule that is used to interpret the values of its output nodes. We expect that an increase in a classifier's coverage will lead to a drop in correctness.

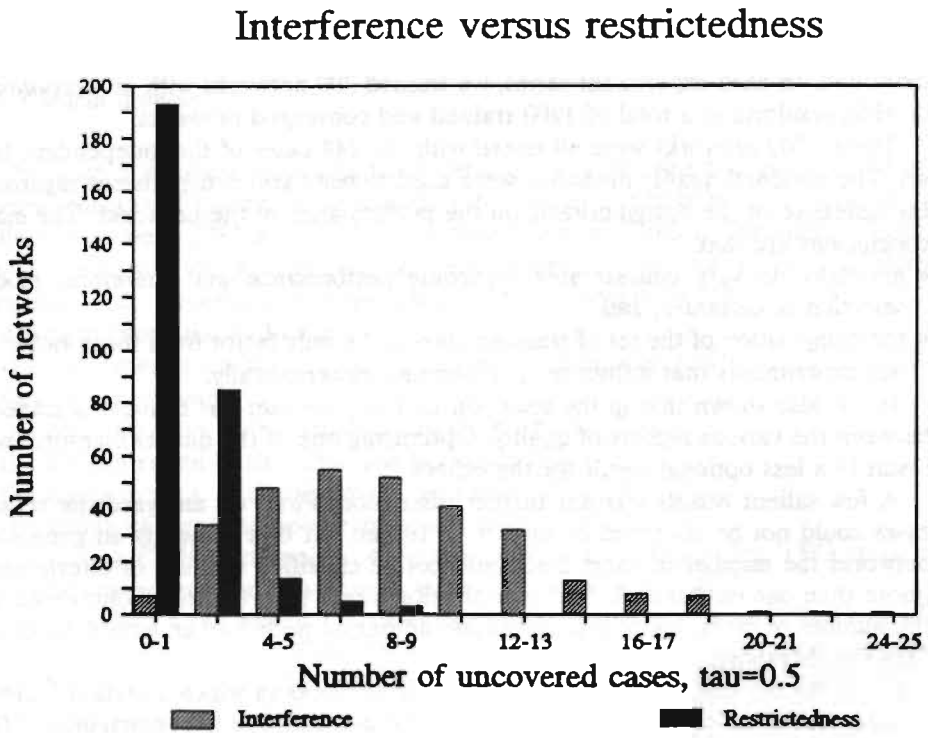


Fig. 4. The distribution of the number of networks that had a certain number of cases not classified because of interference and because of restrictedness. Results are shown for the 300 networks generated from random samples of 50% of the set of training cases. The threshold for the interpretation of the activation of the nodes in the output layer was set to 0.5 for each of the nodes ($\tau_i = 0.5$).

We investigated whether such a tradeoff also existed among a series of networks: do networks with a high coverage have a lower correctness and visa versa. It turned out that this is not generally true (see Fig. 5).

Networks that have a coverage in a certain range can have a large variation in their correctness measures. There is, however, a certain boundary beyond which no network exists. This observation makes it necessary that the users somehow express their preferences with respect to various quality concepts to allow for a multicriteria decision analysis on the quality measures of the generated networks. The users can make statements in terms of the amount of reduction in, for example, correctness they are willing to accept for a certain increase in another quality measure.

That this tradeoff not only can be made for correctness and coverage is shown in Fig. 6. In this scatterplot of correctness versus bias, three different preference profiles are shown. Preference 1 indicates that the user is willing to have a lower correctness, providing that the misclassifications are dispersed (the misclassifications of class A as class B are cancelled by approximately the same amount of

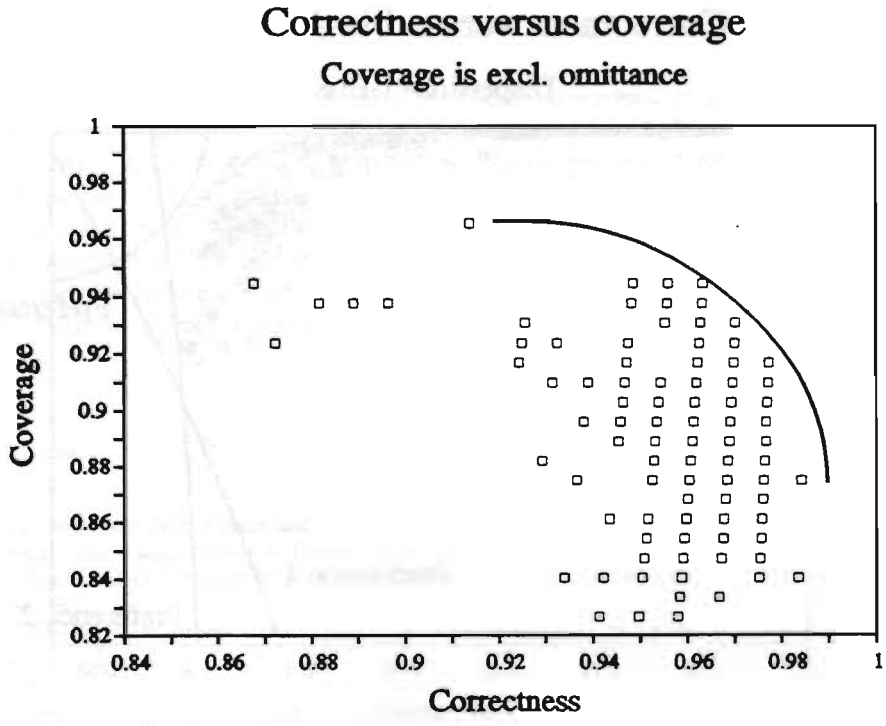


Fig. 5. A scatter plot of correctness versus coverage for a series of networks. There seems to be a boundary on the 'optimal' performance of a network, allowing for a tradeoff between correctness and coverage. Data are shown for the networks trained with 25% of the cases of the learning set. The curve indicates the limit which cannot be exceeded.

cases of class B that are classified as class A). Preference 2 indicates a user who just wants to have the best correctness. He does not mind whether the classifier is biased or dispersed. Preference 3 indicates the profile of a user who prefers a biased classifier and is willing to have a less correct classifier, provided it is more biased, but only to a certain extent.

Clearly the selection of an optimal NN classifier is not a trivial task. In fact, one needs a full multicriteria decision making support for finding the network(s) that best meets user needs. This is particularly true when the users want to express their quality preferences on the basis of the conditional quality metrics. For example, are they willing to have a lower correctness for class A when the coverage for class B increases, and to what extent? A thorough discussion of this issue is out of scope of this paper. It suffices here to indicate that our experimentations with a technique called 'Pareto race' [26] showed promising results. However, more experimentation and comparison with other methods like fuzzy sets [43] is needed, before a generally valid statement about the utility of this approach can be made.

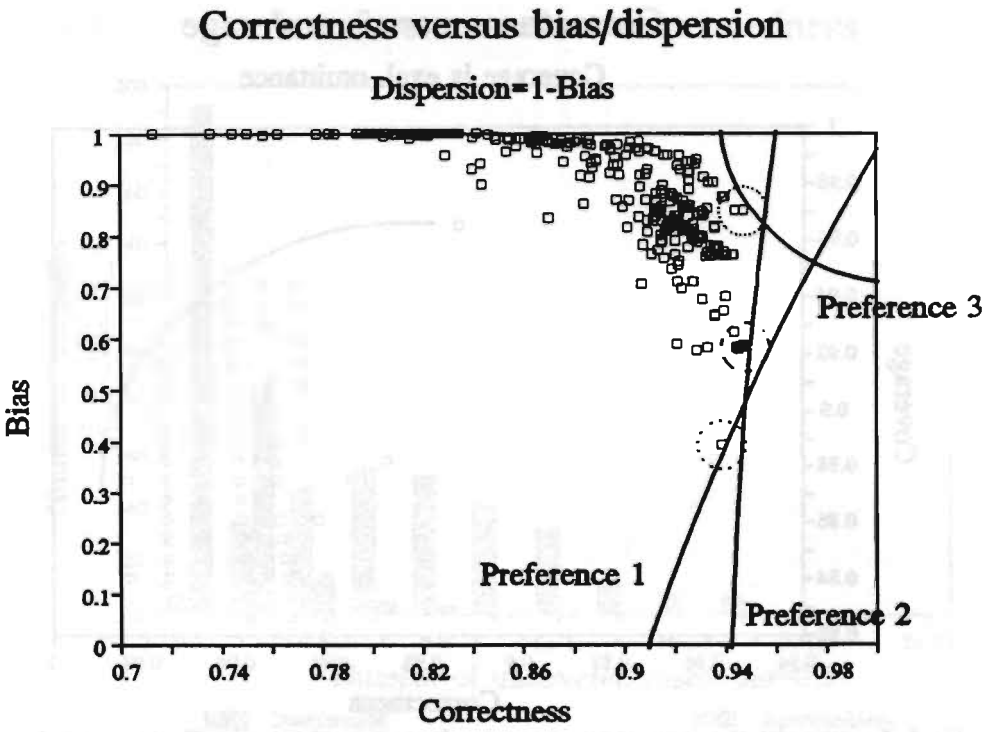


Fig. 6. A scatter plot of correctness versus bias, together with 3 different preference profiles. This figure shows the networks in which only T3 and T4 were utilized. Most of the networks were biased. The three curves illustrate how different users' indifference curves might look like.

6. Conclusions

In this paper we argued for the need of several quality measures to describe the properties of an NN classifier. We introduced a set of quality concepts that is part of a larger framework. We provided a set of metrics covering all quality concepts presented. It was shown that these quality concepts and the corresponding metrics allow users to define criteria that can be used to select a (small) subset of networks that fulfil their needs. The quality concepts presented here are the more basic ones. Among others, one needs to have insight into e.g. the robustness, the credibility, time dependency and ultimately the transferability of a (series of) networks before one will embark on using an NN routinely for a particular task.

Although we focused mainly on the application of the quality concepts and metrics for NNs, the concept of testing and quantifying various quality aspects holds for other types of classifiers as well.

Acknowledgements

The research reported in this paper has been performed in the framework of the KAVAS – Knowledge Acquisition, Visualization and Assessment Study – (A1021) and KAVAS-2 (A2019) projects. These projects have partially been funded by the Commission of the European Communities under the Exploratory phase of AIM and the current AIM Telematics in Health Care programme and by the Academy of the Technical Sciences, Denmark (EF-348). M. Egmont-Petersen performed the research when he was at CRI A/S, Birkerød and DASY, Copenhagen Business School, Copenhagen, Denmark as a PhD student.

References

[1] E.B. Andersen, N.E. Jensen and N. Kougsgaard, *Theoretical Statistics for Economists* (Academic Press, Copenhagen, 2nd ed. in Danish, 1984).

[2] E. Baum and D. Haussler, What size net gives a valid generalization?, *Neural Computat.* 1 (1989) 151–160.

[3] J. Brender, Information systems validation, I: A method for validation of functional aspects, Master thesis, No. 89-1-22, Institute of Computer Science, University of Copenhagen, Denmark, 1989.

[4] J. Brender, P. McNair, H. Raun, J. Nolan and S. Vingtoft, Meta-knowledge as a means for quality management in knowledge-based systems, in: R. O'Moore, S. Bengtsson, J.R. Bryant and J.S. Bryden eds., *Proc. Medical Informatics Europe '90, Lecture Notes in Medical Informatics 40* (Springer Verlag, Berlin, 1990) 360–367.

[5] J. Brender, P. McNair, H. Raun, J. Nolan and S. Vingtoft, Meta-knowledge Concepts, Deliverable 24, Technical report META-1.1 of the KAVAS (A1021) AIM project, 2nd ed., 1990.

[6] J. Brender, J.L. Talmon and P. McNair, Framework for validation of semantic aspects of knowledge, in preparation.

[7] J.A. Cohen, A coefficient of agreement for nominal scales, *Educational and Psychological Measurement* 20 (1960) 37–46.

[8] J. Cunningham and S. Haykin, Neural network detection of small moving radar targets in an ocean environment, in: S.Y. Kung, F. Fallside, J.A. Sorenson and C.A. Kaufmann, eds., *Proc. 1992 IEEE Workshop on Neural Networks for Signal Processing* (IEEE, Piscataway, NJ, 1992) 306–315.

[9] C.-H. de Verdier, T. Aronsson and A. Nyberg, eds., Quality control in clinical chemistry – efforts to find an efficient strategy *Scand. J. Clin. Lab. Invest.* 44, suppl 172 (1984) 1–241.

[10] P.A. Devijver, Reconnaissance des formes par la méthode des plus proches voisins, report R346, Phillips Research Laboratories, Brussels, 1977.

[11] D. Donker, Interobserver variation in the assessment of fetal heart rate recordings, Doctoral dissertation, VU University Press, Amsterdam, 1991.

[12] B.S. Everitt, *Analysis of Contingency Tables* (Chapman&Hall, London, 1977).

[13] J.L. Fleiss, *Statistical Methods for Rates and Proportions* (John Wiley&Sons, New York, 2nd ed. 1981).

[14] J.L. Fleiss, L. Cohen and B.S. Everitt, Large sample standard errors of kappa and weighted kappa, *Psychological Bull.* 72 (1969) 323–327.

[15] W. Gerhardt and H. Keller, Evaluation of test data from clinical studies, I: Terminology, graphic interpretation, diagnostic strategies and selection of sample groups, II: Critical review of the concepts efficiency, receiver operated characteristics (ROC) and likelihood ratios, *Scand. J. Clin. Lab. Invest.* 46, suppl 181 (1986) 1–74.

[16] E.S. Gelsema, Pattern recognition and artificial intelligence in medical research and practice, *Methods of Informat. in Med.* 28 (1989) 63–65.
[17] L.K. Hansen and P. Salamon, Neural Network Ensembles, *IEEE Trans. on Pattern Analysis and Machine Intelligence* 12 (1990) 993–1001.
[18] L.K. Hansen, C. Liisberg and P. Salamon, Ensemble methods for handwritten digit recognition, in: S.Y. Kung, F. Fallside, J.A. Sorenson and C.A. Kaufmann, eds., *Proc. 1992 IEEE Workshop on Neural Networks for Signal Processing* (IEEE, Piscataway, NJ, 1992) 333–342.
[19] S.J. Hanson and D.J. Burr, What connectionist models learn: Learning and representation in connectionist networks, *Behavioral and Brain Sci.* 13 (1990) 471–517.
[20] R.F. Harrison, S.J. Marshall and R.L. Kennedy, A connectionist aid to the early diagnosis of myocardial infarction, in: M. Stefanelli, A. Hasman, M. Fieschi and J. Talmon, eds., *Proc. Third Conf. on Artificial Intelligence in Medicine, Lecture Notes in Medical Informatics 44* (Springer Verlag, Berlin, 1991) 119–128.
[21] A. Hart and J. Wyatt, Connectionist models in medicine: an investigation of their potential, in: J Hunter, J. Cookson and J. Wyatt, eds., *Proc. AIME-89 Conf. Lecture Notes in Medical Informatics 38* (Springer Verlag, Berlin, 1989) 115–124.
[22] G.E. Hinton, Connectionist learning procedures, *Artificial Intelligence* 40 (1989) 185–234.
[23] G. Hripcsak, Using connectionistic modules for decision support, *Methods of Information in Med.* 29 (1990) 167–181.
[24] G.F. Jensen, P. McNair, J. Boesen and V. Hegedüs, Validity in diagnosing Osteoporosis, *Europ. J. Radiol.* 4 (1984) 1–3.
[25] A.N. Komolgorov, On the representation of continuous functions of several variables by superposition of continuous functions of a smaller number of variables, *Dokl. Akad.* 108 (1956) 179–182.
[26] P. Korhonen and J. Wallenius, A Pareto race, *Naval Research Logistics* 35 (1988) 615–623.
[27] S.Y. Kung, F. Fallside, J.A. Sorenson and C.A. Kaufmann, eds., *Neural networks for Signal Processing II, Proc. 1992 IEEE Workshop on Neural Networks for Signal Processing* (IEEE, Piscataway, NJ, 1992).
[28] R.J. Light, Measures of response agreement for qualitative data: Some generalizations and alternatives, *Psychol. Bull.* 76 (1971) 365–377.
[29] F.A. Lodewyk and E. Barnard, Avoiding false local minima by proper initialization of connections, *IEEE Trans. Neural Networks* 3 (1992) 899–905.
[30] M. Minsky and S. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).
[31] R.A. Nadkarni, The quest for quality in the laboratory, *Analytical Chemistry* 63 (1992) 675–682.
[32] P. Nykänen, ed., Issues in evaluation of computer-based support to clinical decision making, Research Report 127, Institute of Informatics, Oslo University, 1989.
[33] T. Read, R.C. Noal and A.C. Cressie, *Goodness-of-fit Statistics for Discrete Multivariate Data* (Springer Verlag, New York, 1988).
[34] D.E. Rumelhart, J.L. McClelland and the PDP Research Group, *Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Vol 1: Foundations* (MIT Press, Cambridge, MA, 1986).
[35] T. Schiøler, W. Grimson, P. Sharpe, M. Egmont-Petersen, G. Momsen, R. O'Moore and P. McNair, Automatic decision support based on voting by independent decision support systems, *Proc. Computing in Clinical Laboratories '92* (1992) 58.
[36] C.N. Schizas, C.S. Pattchis, T.S. Schofield, and P.R. Fawcett, Artificial Neural Nets in computer-aided macro motor unit potential classification, *Trans. IEEE Eng. in Med. and Biol.* (1990) 31–38.
[37] J.W. Shavlik and G.G. Towell, An approach to combining explanation-based and neural learning algorithms, *Connection Sci.* 1 (1989) 231–252.
[38] S. Siegel, *Nonparametric Statistics for the Behavioral Sciences* (McGraw-Hill, Kogakusha, Tokyo, 1956).
[39] P.K. Simpson, *Artificial Neural Systems, Foundations, Paradigms, Applications and Implementations* (Pergamon Press, New York, 1990).
[40] G.G. Towell and J. Shavlik, Extracting refined rules from knowledge-based neural networks, *Machine Learning* 13 (1993) 71–101.
[41] F. Vogelsang, Segmentierung radiologisch dokumentierter fokaler Knochenläsionen auf Basis

kontextbezogener Vektoren mit neuronalen Netzwerken, Diplomarbeit (Master Thesis), Fakultät für Informatik, Medizinische Fakultät, RWTH Aachen, 1993.

[42] J.L. Willems, C. Abreu-Lima, P. Arnaud, C.R. Brohet, B. Denis, J. Gehring, I. Graham, G. van Herpen, H. Machado, J. Michaelis and S.D. Moulopoulos, Evaluation of ECG interpretation results obtained by computer and cardiologists, *Methods of Informat. in Med.* 29 (1990) 308–316.

[43] L.A. Zadeh, Making computers think like people, *IEEE Spectrum* (Aug. 1984) 26–32.

Assessing the discriminative power of attributes for multi-layer perceptrons

3

Authors: M. Egmont-Petersen, J.L. Talmon, A. Hasman

Submitted for publication.

We introduce an approach for assessing the relative importance of attributes for the classification of cases by a trained MLP. The importance of an attribute for the classification of a case – its *discriminative power* – is defined as the probability that the case would obtain *another* class label if the attribute would be observed again while keeping the other attribute values fixed. For each case in a dataset, the overall discriminative power of an attribute is determined by ranking the n attributes according to their discriminative power. These ranks are summed across all cases. Attributes with a low summed rank have a high discriminative power whereas unimportant attributes obtain a high summed rank. The approach is tested on MLPs trained to perform artificial classification tasks and is also used to rank the attributes of an MLP that is trained to segment a radiograph. The approach provides credible results.

1 Introduction

Multi-Layer Perceptrons (MLPs) have been developed for various classification tasks in medicine; e.g. the decision whether to send patients to a cardiac care unit [1], the diagnosis of myocardial infarction [2], classification of patients who are expected to have thyroid disorders [3] and segmentation of radiographs [4,5]. A common problem when developing MLP classifiers is to determine which attributes are important for the performance of the MLP. When many – possibly redundant – attributes are available it is useful to investigate whether the number of attributes can be reduced. Acquisition or computational costs can motivate leaving out specific attributes. One can also prefer some attributes above others based on ethical considerations, a situation not uncommon in medicine.

In this paper we present a novel approach for attribute assessment and selection. The results of such an assessment may be used to prune relatively unimportant attributes such that a simpler classifier is obtained. First, we analyze an approach suggested by Harrison *et al.* for assessing the contribution of attributes to the classification of a single case by an MLP [2]. Based on this analysis, we develop an approach that *ranks* the attributes according to their *discriminative power*: the probability that an attribute may cause a case to be classified differently. We validate our approach with a set of experiments using artificial Gaussian data. Experimentation with MLPs for segmentation of radiographs is used to assess the applicability of the approach for attribute selection in practical classification problems.

2 Background

An MLP performs a mapping from an n-dimensional attribute space onto a c-dimensional class space; $\mathrm{N}(\cdot)$: $\Re^n \rightarrow \Re^c$. We consider MLPs with only one hidden layer consisting of h nodes. Two weight matrices $w^{[1]} \in \Re^{h\times(n+1)}$, $w^{[2]} \in \Re^{c\times(h+1)}$ connect the input with the hidden layer and the hidden layer with the output layer, respectively. The weights $w^{[1]}_{k,n+1}$ and $w^{[2]}_{j,h+1}$ connect a node that always has activation 1 (the bias node) to the kth hidden and jth output node, respectively. The activation of the nodes in the output layer is given by $\vec{o} = \mathrm{N}(\vec{m}) = \mathrm{a}(w^{[2]}\ \mathrm{a}(w^{[1]}\vec{m}))$ where $\mathrm{a}(\cdot)$ denotes the (nonlinear) monotonic activation function that is applied to each element of the vector that is passed as argument. The attribute values of a case form the input vector $\vec{m}$. By interpreting the output vector $\vec{o}$, for example according to the winner-takes-it-all rule, a class label is assigned to the case.

An MLP can be trained to classify cases characterized by real and binary valued attributes. Nominal attributes are usually encoded as a series of binary attributes. The goal is to build a minimal-risk classifier. When the loss function is symmetrical such that all wrong classifications have identical costs and all correct classifications identical gains, this boils down to maximizing the *correctness* (minimizing the error rate). In the following, we consider only the situation with a symmetrical loss function. An MLP with *one* hidden layer with a sufficient number of parameters (weights and bias terms) can implement any arbitrary discriminant function [6]. This makes MLPs generally applicable classifiers. Training an MLP with, for example, back-propagation [7] is tantamount to estimating a statistical discriminant. The training algorithm adjusts the weights such that

the error function – a function of the differences between the expected and observed output – is minimized. The weights implement nonlinear decision boundaries in the attribute space that separate the c classes. Although the available training algorithms do not ensure that the global minimum of the error function is found [8] it has been shown that the output value o_j of an MLP approaches the Bayesian a posteriori probability $P(\omega_j|\vec{m})$ that $\vec{m}$ belongs to class ω_j when the topology of the MLP is complex enough and the training set is sufficiently large [9]. (For a comparison of MLPs with other statistical classifiers see, for example [10]).

Application of MLPs for classification tasks is impeded by a number of unresolved problems:

- It is tedious to determine the performance that can be obtained from different subsets of attributes. The initial weight configuration and topology of the MLP as well as the learning parameters (learning rate, momentum, etc.) may influence the classification result of a particular network. The optimal topology and weight configuration are likely to differ for different subsets of attributes and for different training/test sets [5,11].
- When too many weights and/or attributes are used in relation to the size and composition of the learning set, the learning algorithm will lead to a poor generalization [8].
- The "knowledge" embedded in an MLP is *encoded* as weights on the links between the input, hidden and output nodes. This impedes validation against existing domain knowledge [1].

In this paper, we describe an approach to *attribute* (feature) *assessment* for MLPs.

2.1 Attribute selection

The problem of overgeneralization has been payed much attention to in the literature on neural networks. Especially how to determine the optimal trade-off between the size of the training set and the number of parameters has been investigated (see for example [12]). In a situation where training an MLP causes it to overgeneralize, instead of increasing the size of the training set or reducing the number of hidden nodes, one could try to remove redundant attributes. Leaving out attributes with little or no discriminative power may even cause the performance on a test set to increase; a situation known as *peaking* [13,14].

The problem of selecting attributes for a classification task has been studied extensively in the field of pattern recognition (see e.g. [13–20]). When the distribution of the attributes and its true parameters are known a Bayesian classifier yielding maximal correctness can be developed [16,21]. Such a classifier exhibits no peaking phenomenon as it relies on complete information of the underlying attribute distribution [22]. Although pruning attributes from such a Bayesian classifier cannot increase its performance, one may wish to prune attributes that do not contribute to the discrimination. In practice, however, even when the attribute distribution is known, the parameters have to be estimated from a finite sample of cases and one may be confronted with the peaking phe-

nomenon. In such situations, a search must be made to find the optimal subset of attributes, for example by:

- An exhaustive search in the attribute space in which all combinations of attributes are assessed or by
- The modified Branch And Bound (BAB) search scheme [23]

Exhaustive search entails building $2^n - 1$ different classifiers which is infeasible even for small n. Also the BAB algorithm – although more efficient than exhaustive search – still is an exponential search procedure. When the number of attributes is high, one must take recourse to a suboptimal search method such as *forward* or *backward search* [13,24].

Algorithms for building classification trees – like NPPA [25] and ID3 [26] – and a variant of stepwise discriminant analysis use a *forward search* strategy to select the attributes. In each step, the attribute that provides the best contribution to the discrimination given the attributes that already are selected, is added. Which subset of attributes is optimal depends on the type of classifier used [17], so the subset of attributes used in a discriminant function or classification tree will not necessarily constitute the best subset for an MLP. The forward search process can also be used to build MLP classifiers with an increasing number of attributes. This requires, however, much computation as different MLPs, each with a different subset of attributes, have to be trained. When one wants also to eliminate the effect of choosing a particular topology and initial weight configuration, one needs to train a sample of networks for each subset of attributes, resulting in even more computations.

In backward search one begins with building n classifiers. Each classifier uses a different subset of $n-1$ attributes. The subset used by the classifier with the best performance is retained and used to build $n-1$ new classifiers from the different subsets of $(n-1)-1$ attributes. This search procedure continues as long as the performance of the best classifier exceeds a stop criterion. Nobis used a backward search to reduce the set of attributes in MLPs [11]. His experiments required much computation, especially because also alternative network topologies were tested. Both forward and backward search entail training $\frac{1}{2}(n^2+n)$ different MLPs when the search is performed until reaching the root (forward search) or a leaf of the search tree (backward search). We develop a computationally more attractive approach to reducing the set of attributes. The approach ranks the n attributes based on their discriminative power in a trained MLP. The attribute with the smallest discriminative power is removed from the attribute set and an MLP is trained using the remaining attributes. By leaving out the least important attribute at each step in the backward search procedure, only n networks need to be trained (not taking into account experiments with different topologies, initial weight configurations and training sets).

3 Discriminative power

Harrison *et al.* presented an approach to explain the classification of a case by estimating the contribution of the individual attributes to the classification result [2]. They suggested to compute the partial derivatives of the (winning) output o_j with respect to the n input values. Enbutsu *et al.* suggested to rank the inputs according to the absolute

size of these partial derivatives [27,28]. The rank of the attribute – they use the term *causal index* – is said to express its relative contribution to the classification of the case (for a practical application of the causal index see [28–31]). This approach entails in fact a sensitivity analysis of the classification of a single case as a high value of the partial derivative $\left|\frac{\partial o_j}{\partial m_i}\right|$ implies that the output o_j is sensitive to a change in the input value m_i. The contribution analysis introduced by Harrison *et al.* was extended to assess the discriminative power of attributes using a *set of cases* [5,32]. In this approach, the discriminative power of an attribute was defined as the average over all cases of the absolute value of the partial derivative of the *winning output* with respect to the corresponding input node.

Harrison *et al.* used their contribution analysis to assess the importance of *binary* attributes for discriminating two classes. The neural network they used had only one output node; a low output value was associated with one class and a high value with the other class. The question is whether a partial derivative is a reasonable measure for the sensitivity of the classification from binary inputs as it expresses the change in output for an *infinitesimal* change of a particular input. In fact, Harrison *et al.* could also have 'flipped' each individual attribute value to assess its influence, an approach suggested in [33] and applied in [34] to visualize the knowledge encoded in an MLP. By doing so, the change of the output value can be used as a measure for the sensitivity of the classification to a change in each attribute value.

Whereas only one output node is needed for a dichotomous classification task, it is common to designate an output node for each class when more than two classes are to be discerned. For such networks, flipping a binary input value may influence the activation of more output nodes. Thus, the effect on all output values needs to be considered. To assess the contribution of a binary attribute to the discrimination, one could compute the change in $(o_j - o_v)$ when attribute i is flipped

$$C_{ijv} = \left| \frac{(o_j - o_v) - (o_j^i - o_v^i)}{\Delta_i} \right| \tag{1}$$

where j indicates the winning output and, $v \neq j$, some other output. o_j and o_j^i denote the value of output node j before and after flipping the binary input i, respectively. C_{ijv} is the sensitivity of the difference between the activation of the output nodes j and v to a change $\Delta_i(=1)$ in input i. Figure 1 illustrates for a four class problem the possible effect of flipping an attribute value on the output activations. Some output values increase while others decrease. Another class may even become the 'winner'.

The *overall* sensitivity of the classification (where the case is classified as class j) to a change in attribute i is estimated from C_{ij}:

$$C_{ij} = \max_{v \neq j}(C_{ijv}) \tag{2}$$

For *real valued* attributes, one could in an analogous way use the partial derivatives of the difference in activation between a pair of output nodes. In a method suggested earlier [35], the classification of vector $\vec{m}$ was considered *sensitive* to a change in attribute value

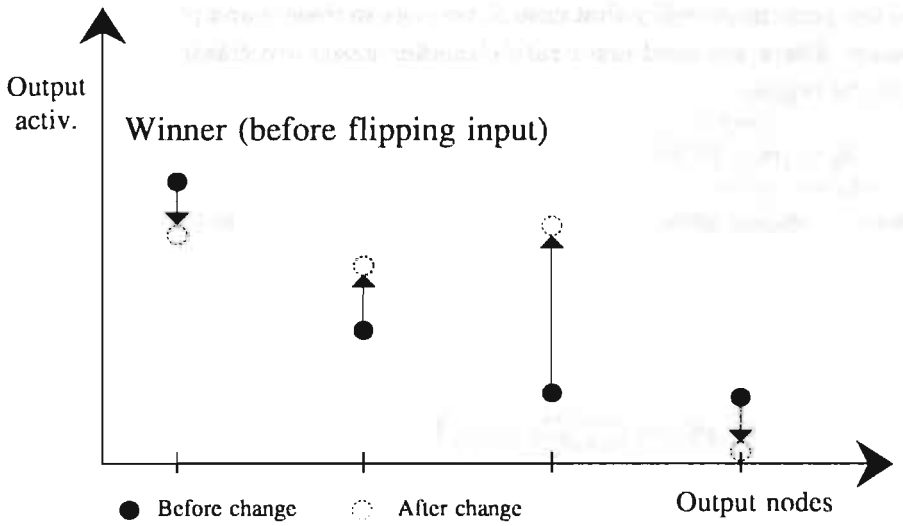


Figure 1. The effect of 'flipping' a binary attribute on the output of an MLP with four output nodes.

m_i when

$$\delta_{ijv} = \left| \frac{\partial(o_j - o_v)}{\partial m_i} \right| \tag{3}$$

is large, where j and v are the indices of the output nodes with the highest and second highest activation, respectively. This approach did not take into account that there might be another output node that had a higher partial derivative. Hence, similar to C_{ij}, $\delta_{ij} = \max_{v \neq j}(\delta_{ijv})$ could measure the overall sensitivity of the classification to a change in attribute i.

Whereas the sensitivity measures C_{ij} and δ_{ij} do somehow comply with our intuitive notion of when an attribute is important for a classification, a measure for the discriminative power of an attribute must take into account two additional aspects:

1. How much an attribute value has to change before the case is classified differently.

2. The probability that values which cause the case to be classified differently are observed.

Take as an example for a particular case two real-valued attributes i and k for which $\delta_{ij} = \delta_{kj}$. The classifier is equally sensitive to a change in both attribute values. If, however, attribute i has a much *wider* conditional distribution than k, attribute i is more likely to change the classification of the case if resampled. A similar situation occurs when for two binary attributes i and k, for which $C_{ij} = C_{kj}$, the probabilities of the '0' event differ, $P(m_i=0) \neq P(m_k=0)$.

3.1 Defining the discriminative power of an attribute

Let $P(\omega_j)$ denote the prior probability that case $\vec{m}$ belongs to class j and $p(\vec{m}|\omega_j)$ its class-conditional density. For a minimal error-rate classifier cases are classified as class j when they fall in the region

$$R_j = \{\vec{m} \in \Re^n | P(\omega_j)p(\vec{m}|\omega_j) > P(\omega_v)p(\vec{m}|\omega_v), \forall v \neq j\} \tag{4}$$

Now, because $p(m_1, \ldots, m_n|\omega_j)$=$p(m_k, \forall k \neq i|\omega_j)\ p(m_i|m_k, \forall k \neq i, \omega_j)$, R_j can be written as

$$R_j = \left\{ \vec{m} \in \Re^n \;\middle|\; \begin{array}{ll} P(\omega_j)\,p(m_k, \forall k \neq i|\omega_j)\ p(m_i|m_k, \forall k \neq i, \omega_j) & > \\ P(\omega_v)p(m_k, \forall k \neq i|\omega_v)\ p(m_i|m_k, \forall k \neq i, \omega_v), & \forall v \neq j \end{array} \right\} \tag{5}$$

Rearranging this expression and keeping constant the values $m_k, \forall k \neq i$, gives

$$R_j^{m_i|m_{k\neq i}} = \left\{ m_i \in \Re \;\middle|\; P(\omega_j)\frac{p(m_i|m_k, \forall k \neq i, \omega_j)}{p(m_i|m_k, \forall k \neq i, \omega_v)} > P(\omega_v)\frac{p(m_k, \forall k \neq i|\omega_v)}{p(m_k, \forall k \neq i|\omega_j)}, \forall v \neq j \right\} \tag{6}$$

$R_j^{m_i|m_{k\neq i}}$ indicates the range of the values of attribute i for which case $\vec{m}$ obtains class label j, given the values of the other $n-1$ attributes. The relation between the two likelihood ratios on each side of the greater than sign determines the extent of the range that encloses class j cases. When for a particular case the attribute-conditional likelihood ratio

$$\frac{p(m_i|m_k, \forall k \neq i, \omega_j)}{p(m_i|m_k, \forall k \neq i, \omega_v)} \approx 1,\ \forall v \neq j,\ \forall m_i \in \Re \tag{7}$$

attribute m_i does not contribute to the discrimination between the classes. In this situation, m_i has no influence on how the case is classified and the region $R_j^{m_i|m_{k\neq i}}$ is either $\Re$ or $\emptyset$. Consequently, the probability $P(m_i \notin R_j^{m_i|m_{k\neq i}}) \in \{0,1\}$ because m_i cannot cause a *change* in classification when the values of the other attributes are kept fixed. A measure for the discriminative power of an attribute can therefore be defined as the probability that the case will be assigned another class label when the attribute measurement is redrawn from the conditional distribution $p(m_i|m_k, \forall k \neq i, \omega_j)$, while keeping the values of the other attributes fixed:

$$PO_{m_i|m_{k\neq i}} \equiv 1 - \int_{R_j^{m_i|m_{k\neq i}}} p(m_i|m_k, \forall k \neq i, \omega_j)\,dm_i \tag{8}$$

This measure still has some problems when operationalizing it:

- Determination of the attribute-conditional range $R_j^{m_i|m_{k\neq i}}$.
- Computing the conditional density $p(m_i|m_k, \forall k \neq i, \omega_j)$.

In the following section, we will suggest a metric, based on this measure, to assess the discriminative power of an attribute taking these problems into account.

3.2 Determining the attribute-conditional range

The partial derivative δ_{ijv} is an indicator for the effect of a small change in m_i on the output of the MLP and we use it to roughly estimate the range $R_j^{m_i|m_{k\neq i}}$. Defining the function $f_{jv}(\vec{m}) = \Delta_{jv} = o_j - o_v$, we may use a Taylor expansion to roughly locate the

intersection point of the decision boundary separating the classes j and v with a line through the point representing the attribute values and parallel to the axis of attribute i; j is the index of the maximal output. The expansion f in variable i becomes:

$$f(m_i^*) = \Delta_{jv} + \frac{\partial f_{jv}(\vec{m})}{\partial m_i}(m_i^* - m_i) + \mathcal{O}(m_i^{*2}). \tag{9}$$

with m_i the observed value of attribute i and m_i^* the running variable. As the partial derivative with respect to attribute i, $\frac{\partial f_{jv}(\vec{m})}{\partial m_i} = \delta_{ijv}$, the value D_{ijv} of m_i for which the activations of the output nodes j and v become equal can be approximated by

$$D_{ijv}(\vec{m}) = m_i^* = -\frac{\Delta_{jv}}{\delta_{ijv}} + m_i \tag{10}$$

A good approximation is probably only obtained for m_i close to the decision boundary. This is precisely the situation where measuring m_i is *relevant* for the classification. When m_i is further away from the boundary the linear approximation to determine the location of the boundary will be inaccurate, but the importance of the attribute is likely to be less for such a case. Figure 2 illustrates how the intersection point D_{ijv} is determined.

If the value D_{ijv} lies within the range of attribute i, this attribute has discriminative power. Hence, it can cause a change of classification. Note that the other $n-1$ attribute values and the classifier together determine the location of the intersection point D_{ijv}.

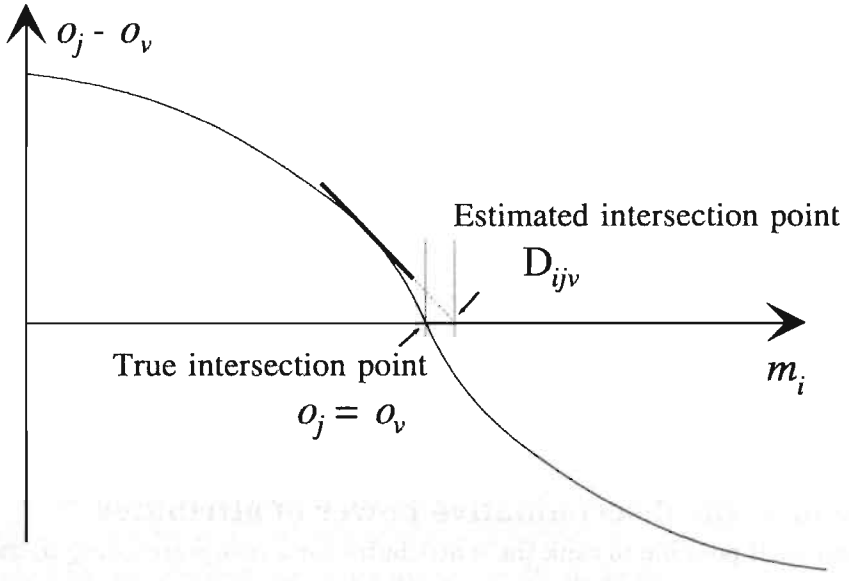


Figure 2. D_{ijv} is the estimated intersection point with the attribute-conditional decision boundary that separates the classes j and v.

3.5 Pairwise comparison of attributes

If the n attributes do not have the same discriminative power and H_0 is *rejected*, the attributes are ordered according to their summed ranks RankO_i. This ranking indicates the relative discriminative power of the attributes. The attribute i with the highest summed rank is regarded as the least powerful attribute for the MLP and is a good candidate to remove. A relevant question is whether this attribute has a *significantly higher* rank than the attribute with the second highest rank.

We can test the hypothesis that two attributes have the same discriminative power: $H_0: \theta_i = \theta_k$ against $H_1: \theta_i \neq \theta_k$ with the significance level α. The difference $|\text{RankO}_i - \text{RankO}_k|$ is used as (two-sided) test statistic and compared with

$$z_{ik} = u_{\alpha/(n(n-1))} \sqrt{\frac{r\,n(n+1)}{6}} \tag{19}$$

with $u_{\alpha/(n(n-1))}$ the abscissa value above which $\alpha/(n(n-1))$ percent of the standard normal distribution is lying [36]. If $|\text{RankO}_i - \text{RankO}_k| > z_{ik}$ we may reject H_0 and conclude that the attribute with the smaller summed rank has a higher discriminative power than the other attribute.

4 Experiments

To assess the value of our heuristic for estimating the discriminative power of attributes, we designed four experiments with artificial Gaussian data. In a fifth experiment, the approach was used to select attributes to be pruned from neural networks for segmentation of a radiograph.

4.1 First experiment with artificial data

We defined two dichotomous classification problems where two attributes were modelled with normal distributions, $N(\vec{m} \mid \vec{\mu}_j, I)$, j=A,B1,B2, with I being the identity matrix. Class A has the centre $\vec{\mu}_A=(0,0)^T$, B1 the centre $\vec{\mu}_{B1}=(2,0)^T$ and B2 the centre $\vec{\mu}_{B2}=\left(\sqrt{2},\sqrt{2}\right)^T$. In problem 1 where classes A and B1 should be discerned, only the first attribute has discriminative power; in the other problem the two attributes have identical discriminative power. We sampled 500 uncorrelated observations from the $N(\vec{m} \mid \vec{\mu}_A, I)$ distribution and 500 observations from each of the two $N(\vec{m} \mid \vec{\mu}_{Bi}, I)$ distributions. The observations were divided into a training set and a test set, each containing 250 vectors of class A and 250 of each class Bi. For each problem, 10 MLPs with a 2-2-2 topology but different initial weights were trained 700 iterations using the 500 training cases (offline learning). Among the 10 MLPs trained for the same classification task, we selected the one with the highest correctness as computed on the training set. The discriminative power of its two attributes was computed on a test set in the way specified above. In figure 3 the estimated intersection point is plotted against the value of attribute 1 for the first classification task. The location of the Bayesian decision boundary is independent of the value of attribute 2 and has the value m_1=1. As can be seen from this plot, the location of the decision boundary is estimated rather well in the neighbourhood of this boundary. The further away the attribute value is from the boundary the less accurate the estimate. In general, the distance to the boundary is overestimated.

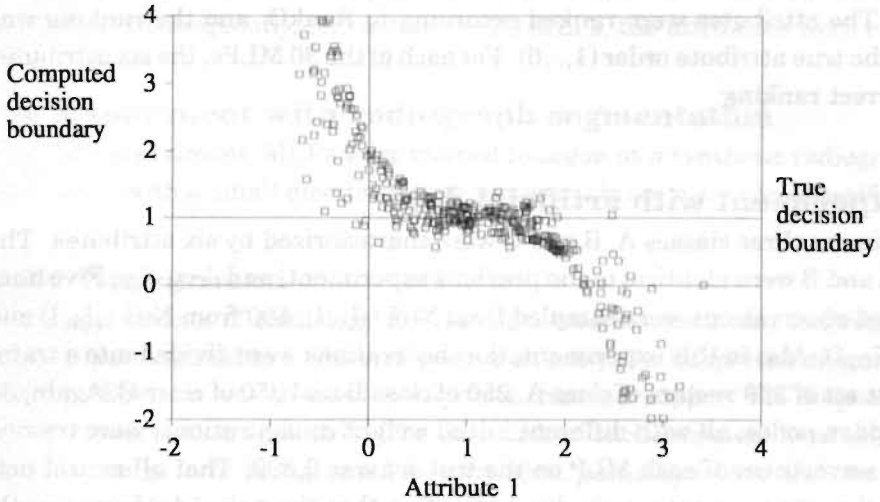


Figure 3. Estimated location of the decision boundary for attribute 1 as a function of the measured value. The Bayesian boundary has the value 1.

Table 1 shows the results of the experiments. Attribute 1 has a significantly higher discriminative power – lower summed rank – than attribute 2 in the first classification problem (A versus B1). In the other classification task, attribute 1 and 2 are equally important as there is no significant difference in the ranks. This is in agreement with the expected outcome.

A versus	B1		B2	
	S.r.	E(S.r.)	S.r.	E(S.r.)
Attribute 1	633	500	739	750
Attribute 2	867	1000	761	750
$P(H_0)$	0.000		0.345	

Table 1. Results from the first experiment, artificial data. S.r. is the summed rank of the attribute, E(S.r.) the expected summed rank and P is the probability that the ranks are equal.

4.2 Second experiment with artificial data

In a second experiment two classes A and B were characterized by six attributes. The class centra were set to $\vec{\mu}_A=(0,0,0,0,0,0)^T$ and $\vec{\mu}_B=(1.75,1.50,1.25,1.00,0.75,0.50)^T$. Five hundred uncorrelated observations were sampled from $N(\vec{m} \mid \vec{\mu}_A, I)$ and 500 from $N(\vec{m} \mid \vec{\mu}_B, I)$. The observations were divided into a training set and a test set of 250 vectors of class A and 250 of class B. Thirty MLPs with two hidden nodes, all with different initial weight configurations, were trained by iterating 700 cycles through the learning set (offline learning). The average correctness of the MLPs on the test set was 0.927 (±

0.028). The correctness of the corresponding Bayesian classifier is 0.929.

For each MLP, the discriminative power of the six attributes was computed from the 500 test cases. The attributes were ranked according to RankO_i and the ranking was compared with the true attribute order (1,..,6). For each of the 30 MLPs, the six attributes obtained the correct ranking.

4.3 Third experiment with artificial data

In a third experiment three classes A, B and C were characterized by six attributes. The class centra of A and B were identical to the previous experiment, and $\vec{\mu}_C = -\vec{\mu}_B$. Five hundred uncorrelated observations were sampled from $N(\vec{m} \mid \vec{\mu}_A, I)$, 500 from $N(\vec{m} \mid \vec{\mu}_B, I)$ and 500 from $N(\vec{m} \mid \vec{\mu}_C, I)$. Also in this experiment, the observations were divided into a training set and a test set of 250 vectors of class A, 250 of class B and 250 of class C. Again, 30 MLPs with 2 hidden nodes, all with different initial weight configurations, were trained 700 cycles. The correctness of each MLP on the test set was 0.899. That all neural networks obtained the same correctness is due to the fact that the networks have exactly the number of degrees of freedom required for this classification task. The expected correctness of the corresponding Bayesian classifier is 0.906.

For each MLP, the discriminative power of the six attributes was computed from the 750 test cases, the attributes were ranked according to RankO_i and the order was compared with the true attribute ranking (1,..,6). We used Kendall's measure T_c for the correlation between n judges and a criterion ranking to compare the attribute ranking from the 30 MLPs with the true ranking. T_c was 0.87, which indicates that our approach ranks the six attributes for each of the 30 MLPs almost correctly. The average ranks were (**1**: 3.319, **2**: 3.228, **3**: 3.478, **4**: 3.480, **5**: 3.714, **6**: 3.780) which indicates that the *2nd* attribute was considered slightly more important than the *1st* attribute.

4.4 Fourth experiment with artificial data

In the fourth experiment three classes A, B and C were characterized by six attributes, two were completely dependent. The class centra were set identical to those in the previous experiment. Each class has the same population covariance matrix $\Sigma_A = \Sigma_B = \Sigma_C$. This covariance matrix Σ is equal to the identity matrix I except for the covariance between attribute 1 and 6 that is set to 1. So the three classes are perfectly separable using the two attributes 1 and 6. Five hundred uncorrelated observations were sampled from $N(\vec{m} \mid \vec{\mu}_A, \Sigma)$, 500 from $N(\vec{m} \mid \vec{\mu}_B, \Sigma)$ and 500 from $N(\vec{m} \mid \vec{\mu}_C, \Sigma)$. As in the previous experiment, the observations were divided into a training set and a test set. Also in this experiment, 30 MLPs with 2 hidden nodes, all with different initial weight configurations, were trained 700 cycles. The correctness of each MLP on the test set was 1.00.

For each MLP, the discriminative power of the six attributes was computed from the 750 test cases. As the three classes are separable by the attributes 1 and 6, the contribution of the other 4 attributes is by definition zero. The statistical test indicated that the four attributes 2–5 did not have a significantly different power, neither did the two attributes 1 and 6. However, attribute 6 had a significantly higher discriminative power

than the four attributes 2–5. Although the two attributes 1 and 6 do not have a significantly different discriminative power attribute 1 had a lower summed rank for all neural networks. Consequently, for each of the 30 MLPs, the attributes were ranked correctly.

4.5 Experiment with radiograph segmentation

In a fifth experiment, MLPs were trained to segment a synthetic radiograph. We pruned attributes with a small discriminative power to obtain a simpler classifier.

4.5.1 Background

The Departments of Radiology and Medical Informatics of the Technical University in Aachen participated in a research project on computer supported diagnosis of focal bone lesions. The aim was to develop a system that can support the diagnosis of focal bone lesions from radiographs [37]. The first task of such a system is to segment the radiographic image into: *background*, *healthy bone*, *pathologic bone* and *tissue* [38]. The Department of Medical Informatics of Maastricht University contributed to this research project. The focus of the cooperation was attribute selection for neural-net classifiers that were trained to perform the segmentation task.

The Department of Radiology provided a sample of 10 radiographs containing different bone lesions. The X-rays were segmented by an experienced radiologist into the four different classes. A mask image that indicates the class membership of the pixels was made. The class membership of only a subset of the pixels in the radiographs was known as not all segments were associated with a specific class. We selected rectangular regions inside the demarcated areas in each of the 10 images. These regions were pasted into 4 images of 492 $\times$ 492 pixels with 256 grey levels. We selected the synthetic radiograph named 'Marvin9' [4] for our experiments.

4.5.2 Attributes used by the MLP

Seven different images were derived from the original synthetic radiograph by various image processing techniques. The following three standard image operations were applied:

- Histogram equalization
- Unsharp masking, 7 $\times$ 7 window
- Median filter, 7 $\times$ 7 window

Haralick defined 14 measures to characterize texture based on a co-occurrence matrix of an image [39]. Weiler *et al.* suggested to compute a co-occurrence matrix for each local neighbourhood in the image to describe local texture properties in an image [40]. We used four of Haralick's measures for each pixel using an 11 $\times$ 11 window which was moved over the whole image:

- Second angular moment of co-occurrence matrix
- Inverse difference moment of co-occurrence matrix

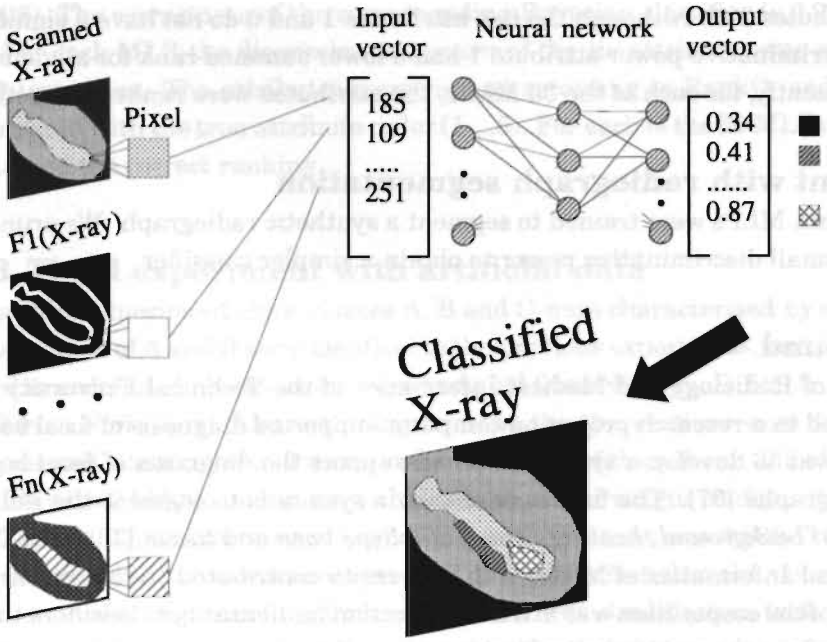


Figure 4. Schematic representation of the segmentation algorithm. The original image is transformed using a number of image operations. These provide the MLP with neighbourhood information of the pixel that is to be classified.

- Contrast according to co-occurrence matrix
- Entropy of co-occurrence matrix

The value of each co-occurrence measure was assigned to the central pixel of the 11×11 window.

The original image together with the 7 derived images provide for each pixel a vector $\vec{m}_p = (\gamma_1, \gamma_2, ..., \gamma_8)^T$, $\gamma_i \in 0,..,255$, where γ_i is the intensity of a pixel in image i. This vector is used as input to the MLP, see figure 4.

4.5.3 Discriminative power

The attribute assessment approach was used to identify attributes with a small discriminative power. First, 7000 pixel vectors were randomly selected and used as training set. Two test sets were randomly selected, one consisting of 996 vectors, the other of 1050 vectors. The first test set is used to compute the performance of the MLPs. The second set is used to compute the discriminative power of the attributes. Therefore, these 1050 vectors in the second set are indirectly a training set as well. A network topology with 6 hidden nodes yielded the best generalization in earlier experiments [4,32]. So we trained 30 MLPs with different initial weight configurations, each with a 8-6-4 topology, for 8000 cycles (offline learning). Figure 5 shows the synthetic radiograph and the segmentation by one of the MLPs using all 8 attributes.

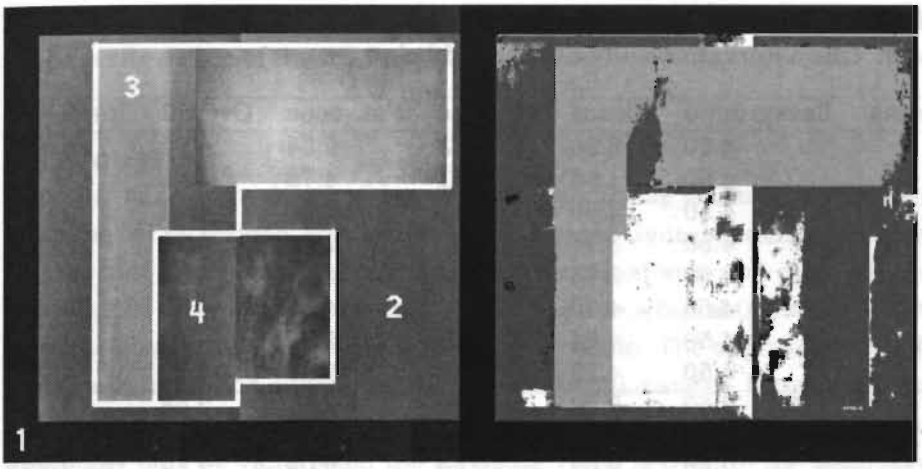


Figure 5. Synthetic radiograph (left) and the segmentation result (right). The numbers 1 to 4 represent the classes background, tissue, healthy and pathologic bone. The right image is classified with an MLP that used 8 attributes. The grey level of a pixel indicates the class membership assigned by the MLP. Black is Background (1), dark grey is tissue (2), light grey is healthy bone (3) and white is pathologic bone (4).

The average correctness of these 30 networks computed with the 996 pixels in the first test set was 0.891. Inspection of the contingency tables indicated that the classes Tissue and Pathologic Bone were the most difficult classes to discern. See table 2.

	Mask			
MLP	Background	Tissue	Healthy bone	Pathologic bone
Background	52955	1540	526	1188
Tissue	10	67679	4979	2493
Healthy bone	0	1210	68184	180
Pathologic bone	0	8750	3731	28639
Total	52965	79179	77420	32500

Table 2. Contingency table of an MLP for pixel classification that uses all 8 attributes.

The discriminative power of the 8 attributes was computed with the second test set (1050 pixels) for each of the 30 MLPs. As discussed in the introduction, the importance of an attribute for a particular neural network may depend on the initial weight configuration. To eliminate the influence of using a specific initial weight configuration, the discriminative power of each attribute was computed by summing $\text{Rank}O_{ij}$ (discriminative power per class) and $\text{Rank}O_i$ (discriminative power across all classes) over all 30 MLPs.

When attributes were compared two by two using the test statistic z_{ik}, α=0.01, Eq. (19), their overall ranks differed significantly, see table 3. All eight attributes had the

same power for discriminating Background from the other classes and Healthy Bone from the other classes. This observation will be discussed below.

Average rank	Background	Tissue	H. Bone	Pat. bone	Overall	Rank
Orig.	4.50	5.30	4.53	5.54	4.95	8
Hist.	4.50	4.68	4.51	4.78	4.61	*5
Ush.	4.50	3.69	4.48	3.30	4.02	*2
Med.	4.50	3.74	4.48	2.98	3.99	1
SAM	4.50	5.13	4.52	5.07	4.82	7
IDM	4.50	4.70	4.50	4.56	4.58	4
Cont.	4.50	4.64	4.50	5.78	4.75	6
Entr.	4.50	4.13	4.49	3.99	4.28	3
Ush.	2.50	2.46	2.49	3.01	2.58	3
Med.	2.50	2.50	2.49	2.99	2.59	*4
IDM	2.50	2.67	2.52	2.06	2.48	2
Entr.	2.50	2.37	2.50	1.93	2.35	1
Ush.	2.00	1.77	1.98	2.24	3.92	*2
IDM	2.00	2.28	2.02	1.96	4.18	3
Entr.	2.00	1.95	2.00	1.79	3.89	1

Table 3. The ranks of the attributes averaged over 30 MLPs computed with 1050 pixels. The first 8 rows contain the average ranks of all attributes for each of the four classes and the average overall rank. The next four rows contain similar ranks for the MLPs with 4 attributes, the last tree rows the ranks of those with 3 attributes. The last column indicates the rank assigned to each attribute according to the overall rank. '*' means that the associated attribute does *not* have a significantly higher average rank than its immediately preceding attribute.

Attributes:	All	4 attributes	3 attributes	2 attributes
Correctness	0.891 ± 0.028	0.903 ± 0.004	0.890 ± 0.007	0.858 ± 0.029

Table 4. Average correctness and 99% confidence intervals computed with the features. The standard errors were estimated among the 30 MLPs.

The 8 attributes are highly dependent and earlier experiments with MLPs for segmentation of similar images had shown that the number of attributes could be reduced to 3 without a significant effect on classifier performance [11]. It was therefore decided to discard the 4 attributes with the smallest discriminative power (Original, Histogram equalization, Second angular moment and Contrast) from the training set and to use the four remaining attributes (Unsharp masking, Median, Inverse difference moment and Entropy) to train a new sample of 30 MLPs. The number of hidden nodes, learning cycles and the learning rate were identical to those used to train the MLPs with all 8 attributes. The correctness of the 30 MLPs was computed with the first test set and the discriminative power of the attributes from the second test set. The attribute with the smallest discriminative power among the four – Median – was discarded and a new set of MLPs was trained using 3 attributes. This procedure was repeated once more, leaving the 2 most discriminating attributes – Unsharp masking and Entropy – for training. The 30 MLPs based on these 2 attributes had a significantly lower performance according to the

χ^2–test, α=0.01, than the 30 MLPs that used 3 or 4 attributes. Table 4 indicates the average performance of the MLPs.

5 Discussion

The heuristic approach to ranking attributes according to their discriminative power was tested on MLPs trained with artificially generated Gaussian data. These experiments indicated that our operationalization of the discriminative power ranks attributes correctly when tested on two and three-class problems, also when attributes are correlated.

Some observations on the approach can be made. The approach uses a *linear* approximation to locate the nearest attribute-conditional decision boundary. For an MLP that implements a nonlinear mapping from the input to the output space, a linear approximation may only be valid when the attribute value is close to the boundary. Figure 3 illustrates the increasing discrepancy between the true and estimated decision boundary when the distance between the attribute value and the true boundary increases.

In general, a small class overlap with a concomitant high correctness entails a small probability that an attribute value is observed close to the boundary. In this case, the linear approximation will be inaccurate, leading to a possibly less precise attribute ranking. An example where two classes can be separated well and where the attribute assessment is less accurate are *Background* and *Healthy bone*. These classes can easily be discriminated using each of the 8 attributes. In this case, the estimated intersection point will overshoot the true boundary and the probability that another class label is assigned will tend to zero. On the other hand, when each of the attributes alone can classify these types of pixels correctly, none of the attributes are likely to be able to alter the assigned class when the values of the other attributes all indicate a specific class. So, none of the attributes has discriminative power given the values of the other attributes. Both reasonings lead to the conclusion that in these cases all attributes should have the same rank. These situations do not form a problem as they are easily detected.

Our approach locates only *one* boundary between the two classes j and v. In the metric for the discriminative power, we made another simplification. Instead of integrating (8) over the whole region $R_j^{m_i|m_{k\neq i}}$, we have replaced it by a one sided measure. The discriminative power of an attribute is computed as the probability that the boundary is exceeded. For many classification problems, more than one attribute-conditional decision boundary may be crossed when the attribute value is varied within its range. Our approach locates only the closest boundary and it is assumed that any value exceeding this boundary results in a different class label. This may lead to an underestimate of the discriminative power of an attribute. When more than two classes are to be discriminated, there may be intersections with class boundaries on both sides of the observed value m_i. When such intersections exist, taking the maximum of the probabilities ignores the possibility that significant parts of the lower *and* upper tail of the distribution of m_i are cut off by different intersections. This problem can be remedied by taking the maximum of the probabilities resulting from intersections at lower values than the observed value and the maximum of the probabilities resulting from intersections at higher values than the observed. Taking the sum of these two maxima gives a better estimate of PO.

The problem of overshooting the nearest intersection point can be overcome using a Newton-Raphson technique that iteratively locates the closest intersection point. In addition, we have developed a method to identify all possible intersection points (see [41]), i.e. all attribute-conditional decision boundaries bounding $R_j^{m_i|m_{k\neq i}}$.

Dependencies between attributes are not considered explicitly in our approach for estimating the discriminative power. Since the decision boundaries implemented by a trained MLP do take dependencies into account the location of the intersection point is influenced by dependencies between attributes (through $R_j^{m_i|m_{k\neq i}}$). Besides, our experiments indeed indicate that correlated attributes were assessed correctly. It is possible to explicitly account for dependencies between attributes if their probability density functions are known or by estimating the conditional densities using advanced nonparametric techniques such as Kernel-functions (see [13]). This approach, which we leave to further research, is not trivial and requires large datasets.

The relative discriminative power of each attribute provides the user of the MLP insight into which attributes enable an MLP to discriminate the different classes, a facility often required when developing MLPs [30]. Besides, the discriminative power of attributes for a single case can be used for explanation purposes as originally suggested by Harrison. Attributes with a high probability that the closest decision boundary is exceeded are more important for assigning class label j than attributes which can be varied without causing the case to obtain a different class label. However, it is not trivial to validate whether such an approach results in adequate explanations.

6 Conclusion

We have presented an approach to estimate the discriminative power of attributes when used by a Multi-Layer Perceptron. We defined the discriminative power of an attribute as the probability that another class label than the assigned one would be obtained when the attribute value is redrawn from the conditional distribution. To determine the discriminative power, the partial derivatives of the activation of the output nodes were used to linearly approximate the input value for which the output values of two output nodes, including the one corresponding to the assigned class, become equal. This value is used in combination with the class-conditional marginal distribution to estimate the discriminative power.

Experiments with artificial two and three class problems based on Gaussian distributions show that our method produces credible results. The developed method was also used in a backward search to prune attributes from MLPs that were trained for segmenting radiographs. These experiments show that even when the attributes are not normally distributed, our method still produces reliable results. Based on the suggested approach, it is possible to develop an explanation facility that provides a user information about which attributes are important for the classification of a specific case.

Acknowledgements

We wish to thank Frank Vogelsang for valuable discussions, the Department of Radiology at the University Hospital of RWTH-Aachen for allowing us to use their DEC-alpha com-

puter for the experiments and Erich Pelikan for kindly providing the image 'Marvin9'.

References

1. A. Hart, J. Wyatt, Connectionist models in medicine: an investigation of their potential, *AIME-89*, Springer Verlag, Heidelberg, 115-124 (1989).
2. R.F. Harrison, S.J. Marshall, R.L. Kennedy, A connectionist aid to the early diagnosis of myocardial infarction, *AIME-91*, Springer Verlag, Heidelberg, 119-128 (1991).
3. M. Egmont-Petersen, J.L. Talmon, J. Brender, P. NcNair, On the quality of neural net classifiers, *Artificial Intelligence in Medicine*, **6**(5), 359-381 (1994).
4. E. Pelikan, F. Vogelsang, B. Schultz, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf, Röntgenbildsegmentierung durch topologische Karten oder Multilayer-Perceptron ein Vergleich, *Proceedings 2'nd workshop on digital image processing in medicine. Freiburg* (1994).
5. F. Vogelsang, E. Pelikan, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf, Segmentierung von Röntgenbildern fokaler Knochenläsionen durch neuronale Netzwerke – Optimierung durch Quality Metrics und modifizierte Contribution Analysis, *Mustererkennung*, 450-459 (1993).
6. J.D. Villiers, B. Bernard, Backprogation neural nets with one and two hidden layers, *IEEE Transactions on Neural Networks*, **4**, 136-141 (1992).
7. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error propagation, *Parallel distributed processing. Exploration into the microstructure of cognition*, D.E. Rumelhart, J.L. McClelland and the PDP research group, MIT Press, Cambridge, Vol. 1, Chap. 8, 318 (1986).
8. J. Hertz, A. Krogh, R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison Westley, Redwood City, CA. (1991).
9. M.D. Richard, R.P. Lippmann, Neural network classifiers estimate Bayesian a posteriori probabilities, *Neural Computation*, **3**, 461-483 (1991).
10. B.D. Ripley, Statistical aspects of neural networks, *Networks and Chaos – Statistical and Probabilistic Aspects*, 40-123 (1993).
11. T. Nobis, *Berücksichtigung lokaler und globaler Textureigenschaften durch Erweiterung des Konzepts der Grauwertübergangsmatrizen auf einen Multi-Skalen Ansatz*, Diplomarbeit (Master Thesis), Institut für Medizinische Informatik und Biometrie, Technical University of Aachen, Aachen (1994).
12. E. Baum, D. Haussler. "What size net gives a valid generalization?". *Neural Computation*, **1**(1), 151 -160 (1989).
13. D.J. Hand, *Discrimination and classification*, John Wiley & Sons, Chichester (1981).
14. W. Schaafsma, Selecting variables in discriminant analysis for improving upon classification procedures, *Handbook of statistics*, **2**, North Holland, 857-881 (1982).
15. S.-T. Bow, *Pattern recognition and image processing*, Marcel Dekker, New York (1992).
16. R.O. Duda, P.E. Hart, *Pattern classification and scene analysis*, John Wiley & Sons, New York (1973).

17. M. Kudo, M. Shimbo, Feature selection based on the structural indicies of categories, *Pattern recognition*, **26**(6), 891-901 (1993).
18. G.J. McLachlan, *Discriminant analysis and statistical pattern analysis*, John Wiley & Sons, N.Y. (1992).
19. P.M. Narenda, K. Fukunaga, A branch and bound algorithm for feature subset selection, *IEEE Transactions on Computers*, **26**, 917-922 (1977).
20. B. Yu, B. Yuan, A more efficient branch and bound algorithm for feature selection, *Pattern Recognition*, **26**(6), 883-889 (1993).
21. P.A. Devijver, J. Kittler, *Pattern recognition: A statistical approach*, Prentice-Hall, Englewood Cliffs, N.J. (1982).
22. W.G. Waller, A.K. Jain, On the monotonicity of the performance of a Bayesian classifier, *IEEE Transactions on Information Theory*, **2**, 392-394 (1978).
23. W. Siedlecki, J. Sklansky, On automatic feature selection, *International Journal of Pattern Recognition and Artificial Intelligence*, **2**(2), 197-220 (1988).
24. J. Kittler, Feature set and search algorithms, *Pattern recognition and signal processing*, Sijthoff & Noordhoff, Alphen a/d Rijn, 41-60 (1978).
25. J.L. Talmon, A multiclass nonparametric partitioning algorithm, *Pattern Recognition Letters*, **4**, 31-38 (1986).
26. J.R. Quinlan, Induction of decision trees, *Machine learning*, **1**(1), 81-106 (1986).
27. I. Enbutsu, K. Baba, N, Hara, Fuzzy rule extraction from a multilayered neural network, *International Joint Conference on Neural Networks*, Vol. 2, 461-465 (1991).
28. M. Yoda, K. Baba, I. Enbutsu, Explicit representation of knowledge acquired from plant historical data using neural networks, *International Joint Conference on Neural Networks*, Vol. 3, 155-160 (1991).
29. C.C. Klimasauskas, Neural nets tell why, *Dr. Dobb's journal*, April, 16-24 (1991).
30. J. Šíma, Neural expert systems, *Neural Networks*, **8**(2), 261-271 (1995).
31. H. Turner, T.D. Gedeon, Extracting Meaning from Neural Networks, *Proceedings 13th International Conference on AI, Avignon*, Vol. 1, 243-252 (1993).
32. F. Vogelsang, E. Pelikan, M. Egmont-Petersen, T. Tolxdorf, K. Bohndorf, Segmentierung von Röntgenbildern fokaler Knochenläsionen durch neuronale Netzwerke. Optimierung durch Quality Metrics und modifizierte Contribution Analysis, *Proceedings of the workshop on neural networks at the RWTH-Aachen*, Aachen, 201-210 (1993).
33. M. Egmont-Petersen, An approach for generating explanations in neural networks, *Proceedings for the KAVAS Workshop on Knowledge Acquisition, Visualization, and Assessment*, Computer Resources International, Copenhagen, 131-136 (1990).
34. J.L. Talmon, W.R.M. Dassen, V. Karthaus, Neural nets and classification trees, *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems*, Elsevier, Amsterdam, 415-423 (1994).
35. M. Egmont-Petersen, J.L. Talmon, E. Pelikan, F. Vogelsang, Contribution analysis of multi-layer perceptrons. Estimation of the input sources' importance for the classification, *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems*, Elsevier, Amsterdam, 347-358 (1994).

36. S. Siegel, N.J. Castellan, *Nonparametric statistics for the behavioral sciences*, 2'nd ed, McGraw Hill, Singapore (1988).
37. E. Pelikan, K. Bohndorf, T. Tolxdorff, D. Zarrinnam, B. Wein, Computer-unterstützte Diagnose von fokalen Knochenläsionen, *Radiologia diagnostica*, *35*(1), 34-37 (1994).
38. E. Pelikan, *Texturorientierte Segmentierungsmethoden in der medizinischen Bildverarbeitung*, Ph.D. thesis, Faculty of Mathematics and natural sciences of the Technical University of Aachen, Aachen (1994).
39. R.M. Haralick, R. Shanmugan, I. Dinstein, Textual features for image classification, *IEEE Transactions on Man Systems and Cybernetics*, **3**(6), 610-621 (1973).
40. F. Weiler, E. Pelikan, T. Nobis, T. Tolxdorff, K. Bohndorf, Texturbasierte Extraktion medizinischer Merkmale aus Filmröntgenbildern, *Mustererkennung*, 460-467 (1993).
41. M. Egmont-Petersen, J.L. Talmon, A. Hasman, A.W. Ambergen, Assessing the importance of features for Multi Layer Perceptrons, *In preparation*.

Assessing the importance of features for Multi-Layer Perceptrons

4

Authors: M. Egmont-Petersen, J.L. Talmon, A. Hasman, A.W. Ambergen

Submitted for publication.

Abstract

In this paper, we establish a mathematical framework in which we develop measures for determining the contribution of features to the performance of a classifier. Corresponding to these measures, we design metrics that allow estimation of the importance of features for a specific multi-layer perceptron neural network. We also present a method for pruning input nodes from the network such that most of the knowledge encoded in its weights is retained. The proposed metrics and the pruning method are validated with a number of experiments with artificial classification tasks.

4

Assessing the Importance of Features for Multi-Layer Perceptrons

1 Introduction

Multi-layer perceptrons (MLPs) have been trained to perform various classification tasks [6,10,11,12,14,20,25,32,33,40]. An MLP performs a mapping from an input (feature) space onto an output (class) space. Cases are represented in the input space by a vector of feature values. The output vector is used to classify a case; e.g. by means of the winner-takes-it-all rule. An MLP is an interesting alternative to other classifiers: Even when the type of distribution of the features is unknown an MLP with the optimal number of hidden nodes approaches a Bayesian classifier and hence its error rate will be close to the minimal error rate [30].

For many classification tasks a large number of potentially useful features can be defined. Acquisition costs or the computational effort needed to provide and process all features can be a motivation to reduce the number of features used by an MLP [15,16]. It can also be valuable to know which features influence the classification of cases. Such information can be used for verification against available domain knowledge or it may yield new insight in a domain.

Ideally, one wants to rank the available features according to the change in performance that results from removing or adding the respective feature from the feature set [34]. One criterion for ranking the features could be the change in correctness that results when a feature is removed. We define the *marginal contribution* of a feature k in relation to a set of $(n-1)$ features as the difference in error rate of a classifier based on n features and a classifier based on all but the kth feature. When the distribution of the features is unknown, it is computationally complex to estimate the minimal error rates needed to compute the marginal contribution. Another possibility is to rank features using heuristic criteria such as the discriminative power (see [8]). Also classifier independent probabilistic distance and dependence measures can be used as ranking criteria. However, these measures do not take the assumptions and properties of a particular classifier into account.

We will define probabilistic measures that establish different upper bounds for the marginal contribution of a feature. These measures are made operational with metrics that make it possible to estimate bounds on the marginal contribution of features such that the ones with the smallest contribution can be pruned from a particular neural network.

First, we discuss different criteria for feature assessment that have been proposed in the literature. Also different approaches to searching for the best subset of features are discussed. We then define a number of probabilistic measures to estimate the marginal contribution of a feature for a classifier. The metrics can be used as ranking criteria to decide which feature to prune from the input space. Our goal is to identify superfluous/inferior features and to prune such features without having to train an MLP from scratch based on the remaining features. A pruning method with these properties is also presented. The metrics and the pruning method are embedded in a backward search procedure. They are evaluated in a number of experiments with different datasets.

2 Background

Classification is assigning a class label to a case based on an n-dimensional feature vector $\boldsymbol{x}$[1]. Let $p(\boldsymbol{x}|\omega_j)$ denote the n-dimensional class-conditional probability density function (PDF) of the n features for class j, j=1,..,c. When the PDFs of the features are unknown, it is difficult to identify the subset of features that results in the best possible trade-off between classifier performance and the size and composition of the feature set. In the simple case where the features are associated with equal acquisition costs, all types of misclassifications are equally bad and all correct classifications have the same benefit, the best assessment criterion is the estimated error rate that may be obtained from each subset of features.

2.1 Assessment criteria

In the general case where the type of distribution of the features is unknown, it is computationally complex to estimate the minimal error rate that can be obtained from a (sub)set of features using a specific classifier. Different assessment criteria have been suggested in the literature on statistical pattern recognition. Among these, probabilistic distance measures, dependence measures and entropy measures have been proposed (for an overview see [17,34]). An example is the Mahalanobis distance measure that is based on the assumption that the c class-conditional PDFs of the features are Gaussian. With some distance measures, bounds of the error rate for the assessed feature subset can be determined. One disadvantage of some probabilistic measures is that the PDFs of the features should be known. Another disadvantage is that the relationship between the distance measures and the error rate is often very loose [17].

Battiti defines a metric to rank different subsets of features using the *mutual information*, an entropy based distance measure [2]. The mutual information is defined as the difference between the total entropy $H(\Omega)$ and the conditional entropy $H(\Omega|\boldsymbol{x})$, $H(\Omega)-H(\Omega|\boldsymbol{x})$, with Ω being the set of classes and $\boldsymbol{x}$ the feature vector that is assessed. Battiti compares $H(\Omega|\boldsymbol{x})$ with $H(\Omega|\boldsymbol{x}')$, where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are vectors with and without the feature that is assessed. He also introduces a *penalty* that is large if the assessed feature is highly dependent on other features in $\boldsymbol{x}$ (also computed using the mutual information). So his metric takes two aspects into account: the ability to discriminate the classes and the degree to which the feature may be predicted from the other features. Battiti's metric linearly weighs these two aspects. A drawback of Battiti's metric – and also of probabilistic distance and dependency measures – is that they do not take into account the properties of a particular classifier, nor that they assess the contribution of each feature to the *performance* of the classifier [9,34].

---

[1]Henceforward, a capital letter X denotes a matrix, a bold letter $\boldsymbol{y}$ a column vector. $\boldsymbol{x}_i \in X$ denotes column i in X, $\boldsymbol{x}^{<k>}$ denotes row k in X, and $x_{k,i}$ the kth element in column i in X. The ith element in vector $\boldsymbol{y}$ is denoted by y_i. A function is in the main text rendered by f(·). In general, $P(E)$ denotes the probability that the event E is observed, $p(x)$ the probability density function of variable x.

A similar problem exists when a set of features that is optimal for one type of classifier is used as feature set for another type of classifier. There is no guarantee that this feature set is also optimal for the other classifier. This is particularly true if the underlying assumptions for the classifiers are very different, for example that one classifier is parametric and the other is not.

2.2 Search schemes

A number of approaches for feature selection have been developed [2,9,13,16,18,34, 35]. Such approaches combine a feature assessment criterion with a search scheme to identify the best subset of features. Some approaches select features by a *forward search* scheme. The feature that discriminates the most (according to a selection criterion, e.g. its marginal contribution) is used to build a classifier. In the next step, the (remaining) features are evaluated. The feature that performs best in combination with the already selected feature(s) is added to the feature set, the classifier is rebuilt and the procedure is repeated. When to stop depends on the type of classifier used and the size and composition of the learning set. Examples of learning algorithms that select features this way are NPPA [38] and ID3 [27]. Also in a variant of stepwise discriminant analysis features are selected by forward search [5].

Other approaches select features by a *backward search* scheme. One starts building n classifiers, one for each different subset of $n-1$ features. The classifier with the maximal performance is assumed to contain the best subset of $n-1$ features. Based on this subset, the procedure is repeated by building a classifier with each subset consisting of $((n-1)-1)$ features. Usually, one stops when the performance becomes unacceptably low. Backward search has been used to select subsets of features for MLPs and Kohonen's feature maps [23,40].

Even when the error rate is used as assessment criterion, forward and backward search do not necessarily lead to the optimal subset of features for a particular type of classifier. If the performance (on a test set) of an MLP is incidentally largest because of statistical fluctuations, one ends up exploring an inferior subset of features [9]. However, Fouroutan and Sklansky have demonstrated that a backward search procedure yields 'close to' optimal subsets of features when used as a selection procedure for their piecewise linear classifier [9].

Also the Branch-and-Bound algorithm [22] may fail to find the optimal subset of features when pruning a feature leads to a performance increase on a test set. Siedlecky and Sklansky have extended the Branch-and-Bound algorithm to better cope with small increases in performance [34]. The error rates estimated for each investigated subset of features are compared using a tolerance Δ. The feature subsets are ranked according to their performance and the best subsets with a performance within the range Δ are all explored further by the algorithm.

Siedlecky and Sklansky also developed a *genetic algorithm* for feature selection [35]. They designed an experiment with 24 features, some of which did not provide any discriminative information. They compared their genetic algorithm with forward and backward search and with their modified Branch-and-Bound algorithm. They

used exhaustive search to identify the subset of features among the $2^{24}-1$ possible subsets that resulted in the best classifier performance. In the experiments where different parameters of the genetic algorithm were varied, 1000 to 3000 classifiers were built before the stop criterion was met. The subset of features that was identified resulted in an error rate of 12.3%. Backward search gave a subset with an error rate of 12.4% by building only 301 classifiers. Siedlecky and Sklansky reported the minimal error rate to be 12.2% based on exhaustive search. Although the genetic algorithm outperformed both the forward and backward search schemes as well as the Branch-and-Bound algorithm, it is computational more complex than the latter three approaches. Using a genetic approach to select features for MLPs would imply building a large population of networks. In each "generation", new MLPs would have to be trained while others are discarded from the population[2]. So, one can question whether in a practical application of MLPs, the extra computations – a factor ten – justify the 0.1% decrease of the error rate to 12.3%.

Backward search has a drawback when used to find the best subset of features for MLPs. The first step of a backward search results in n MLPs, each with a different subset of $n-1$ features. However, learning algorithms as for example Back-Propagation or Conjugate Marquart do not guarantee convergence to the global minimum of the error function. When the MLP that is trained with the optimal subset of features ended up in a local minimum and its error rate exceeds the error rate of one of the other MLPs, the optimal subset of features is not explored further. In this case, backward search results in a suboptimal set of features. The problem of local minima is usually remedied by training several MLPs, each with different initial weights and topologies. However, this is a toilsome procedure.

To overcome this problem we propose a method to pruning an input node from a trained MLP. The pruning method removes the input node from the MLP by discarding the weights that connect to this node and by adapting the weights that connect the remaining input nodes with the hidden nodes. Thereby, most of the knowledge embedded in the MLP is retained and retraining may not be necessary. The measures and metrics we propose can be used to guide this process and to obtain estimates of the performance of the pruned MLP.

3 Four feature measures

We define a set of probability measures to estimate bounds of the marginal contribution of a feature to the performance of a statistical classifier. Each probability measure is later made operational by a metric.

[2]It is possible to use a genetic algorithm also to construct the MLPs. One can create a new generation of MLPs by combining weights from well performing MLPs in the previous generation. The probability that a new MLP inherits weight(s) from an MLP in the previous generation is related to the performance of this ancestor.

3.1 Classification

In general, classifiers partition the feature space into disjoint regions R_j, $j=1,..,c$. Cases that occur in R_j are assigned class label j. For a minimal error-rate classifier based on n features, R_j is given by

$$R_j^n = \{\boldsymbol{x} \in \mathbb{R}^n \mid P(\omega_j)\,p(\boldsymbol{x} \mid \omega_j) > P(\omega_l)\,p(\boldsymbol{x} \mid \omega_l), \forall l \neq j\} \tag{1}$$

with $P(\omega_j)$ the prior probability of class j. The probability of classifying a class j case correctly is

$$P(\boldsymbol{x} \in R_j^n \mid \omega_j) = \int_{R_j^n} p(\boldsymbol{x} \mid \omega_j)\,d\boldsymbol{x} \tag{2}$$

The *correctness* ρ^n of the classifier using n features is

$$\rho^n = \sum_{j=1}^{c} P(\omega_j)\,P(\boldsymbol{x} \in R_j^n \mid \omega_j) \tag{3}$$

For a minimal error-rate classifier, the marginal contribution of a feature – the decrease in correctness that results when feature k is removed – is

$$\Delta\rho^{\backslash k} = \sum_{j=1}^{c} P(\omega_j)\left(P(\boldsymbol{x} \in R_j^n \mid \omega_j) - P(\boldsymbol{x}^{\backslash k} \in R_j^{\backslash k} \mid \omega_j)\right) \tag{4}$$

with $\boldsymbol{x}^{\backslash k}$ an $n-1$ dimensional vector that is equal to $\boldsymbol{x}$ except for feature k that has been removed, the probability

$$P(\boldsymbol{x}^{\backslash k} \in R_j^{\backslash k} \mid \omega_j) = \int_{R_j^{\backslash k}} p(\boldsymbol{x}^{\backslash k} \mid \omega_j)\,d\boldsymbol{x}^{\backslash k} \tag{5}$$

and the region

$$R_j^{\backslash k} = \{\boldsymbol{x}^{\backslash k} \in \mathbb{R}^{n-1} \mid P(\omega_j)\,p(\boldsymbol{x}^{\backslash k} \mid \omega_j) > P(\omega_l)\,p(\boldsymbol{x}^{\backslash k} \mid \omega_l), \forall l \neq j\} \tag{6}$$

3.2 Feature measures

The right hand side of (2) can be rewritten as

$$\int_{R_j^{n\backslash k}} \left[\int_{S_j(\boldsymbol{x}^{\backslash k})} p(\boldsymbol{x} \mid \omega_j)\,dx_k \right] d\boldsymbol{x}^{\backslash k} \tag{7}$$

with $R_j^{n\backslash k}$ denoting the projection of the region R_j^n onto the $n-1$ dimensions excluding dimension k:

$$R_j^{n\setminus k} = \left\{ x^{\neq k} \in \mathbb{R}^{n-1} \mid \exists x_k \in \mathbb{R}: P(\omega_j)\, p(x^{\neq k}, x_k \mid \omega_j) > P(\omega_l)\, p(x^{\neq k}, x_k \mid \omega_l), \forall l \neq j \right\} \tag{8}$$

The range $S_j(x^{\neq k})$ is a function of the values of all features except k

$$S_j(x^{\neq k}) = \{ x_k \in \mathbb{R} \mid P(\omega_j)\, p(x^{\neq k}, x_k \mid \omega_j) > P(\omega_l)\, p(x^{\neq k}, x_k \mid \omega_l), \forall l \neq j \} \tag{9}$$

The larger the range $S_j(x^{\neq k})$ and the larger the probability that the value will fall within this range the less feature k influences the classifier. Using the fact that $p(x^{\neq k}, x_k \mid \omega_j) = p(x^{\neq k} \mid \omega_j)\, p(x_k \mid x^{\neq k}, \omega_j)$, (7) can be rewritten as the integral over the product

$$\int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j) \left[\int_{S_j(x^{\neq k})} p(x_k \mid x^{\neq k}, \omega_j)\, dx_k \right] dx^{\neq k} \tag{10}$$

Rewriting (9) gives

$$S_j(x^{\neq k}) = \left\{ x_k \in \mathbb{R} \;\middle|\; \frac{p(x_k \mid x^{\neq k}, \omega_j)}{p(x_k \mid x^{\neq k}, \omega_l)} > \frac{P(\omega_l)}{P(\omega_j)} \frac{p(x^{\neq k} \mid \omega_l)}{p(x^{\neq k} \mid \omega_j)}, \forall l \neq j \right\} \tag{11}$$

It is clear that $S_j(x^{\neq k})$ is determined by the relation between the attribute-conditional likelihood ratio (left), the likelihood ratio of the $n-1$ other features (right) and the prior probabilities. When for a case x, $S_j(x^{\neq k}) \in \{\varnothing, \mathbb{R}\}$, the other $n-1$ features determine exclusively its class label.

The correctness ρ^n can be rewritten as

$$\rho^n = \sum_{j=1}^{c} P(\omega_j) \int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j) \left[\int_{S_j(x^{\neq k})} p(x_k \mid x^{\neq k}, \omega_j)\, dx_k \right] dx^{\neq k} \tag{12}$$

Equation (12) can be used to obtain some insight in the marginal contribution of feature k. An overly optimistic estimation of the marginal contribution of a feature can be obtained when we assume that whenever feature k can influence the classification it will do so in a negative way. The resulting correctness ρ^{n1} will be smaller than the correctness ρ^n and the difference is due to feature k. The assumption has as a consequence that the integral in (12)

$$\int_{S_j(x^{\neq k})} p(x_k \mid x^{\neq k}, \omega_j)\, dx_k \tag{13}$$

is equal to zero, whenever $S_j(x^{\neq k})$ is not equal to $\mathbf{R}$. We will call $\rho^n - \rho^{n\setminus}$ the *potential influence* (ϕ_k) of feature k. The potential influence is equal to

$$\phi_k \equiv \rho^n - \sum_{j=1}^{c} P(\omega_j) \int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j)\, g(S_j(x^{\neq k}) = \mathbf{R})\, dx^{\neq k} \tag{14}$$

with

$$g(e) = \begin{cases} 1: & e = \text{TRUE} \\ 0: & e = \text{FALSE} \end{cases} \tag{15}$$

We can prove:

Theorem 3.2.1 The potential influence is always larger than or equal to the decrease in correctness that results when feature k is removed from the feature set, $\phi_k \geq \Delta\rho^{\neq k}$.

Proof

The inequality can be written as $\rho^n - \phi_k \leq \rho^{n\neq k}$ (as $\Delta\rho^{\neq k} = \rho^n - \rho^{n\neq k}$, $\rho^{n\neq k}$ is the maximal correctness that can be obtained using all n features but k) or

$$\sum_{j=1}^{c} P(\omega_j) \int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j)\, g(S_j(x^{\neq k}) = \mathbf{R})\, dx^{\neq k} \leq \sum_{j=1}^{c} P(\omega_j) \int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j)\, dx^{\neq k} \tag{16}$$

For class j, we can prove

$$\int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j)\, g(S_j(x^{\neq k}) = \mathbf{R})\, dx^{\neq k} \leq \int_{R_j^{n\setminus k}} p(x^{\neq k} \mid \omega_j)\, dx^{\neq k} \tag{17}$$

as $S_j(x^{\neq k}) = \mathbf{R}$ if and only if $x^{\neq k} \in R_j^{n\setminus \forall k}$ with

$$R_j^{n\ \forall k} = \left\{ \boldsymbol{x}'^k \in \mathbb{R}^{n-1} \mid S_j(\boldsymbol{x}'^k) = \mathbb{R} \right\} \tag{18}$$

So we may write the left hand integral of (17) as

$$\int_{R_j'^{\setminus \forall k}} p(\boldsymbol{x}'^k \mid \omega_j)\, d\boldsymbol{x}'^k \tag{19}$$

As (6) can be written as

$$R_j'^k = \{ \boldsymbol{x}'^k \in \mathbb{R}^{n-1} \mid S_j(\boldsymbol{x}'^k) \neq \varnothing \} \tag{20}$$

$R_j^{n\ \forall k} \subseteq R_j'^k$. So (17) is proven and thereby (16) holds. Therefore, $\phi_k \geq \Delta\rho^{*k}$ ■

The potential influence of a feature ϕ_k therefore is a bound for the maximal decrease in correctness that can occur when feature x_k is removed.

Instead of the potential influence that clearly overestimates the contribution of feature k to ρ^n one can also try to estimate the contribution to ρ^n of that part of feature k that is independent of the other $n-1$ attributes. The part of feature k that is dependent on the other $n-1$ features is computed by its expected value given the values of the other features

$$E(x_k \mid \boldsymbol{x}'^k) = \int_{-\infty}^{\infty} x_k\ p(x_k \mid \boldsymbol{x}'^k)\, dx_k \tag{21}$$

The difference between ρ^n and the resulting ρ^{n2}, which we will call the *replaceability* of feature k, ι_k, is the contribution of the independent part of feature k

$$\iota_k \equiv \sum_{j=1}^{c} P(\omega_j) \int_{R_j'^{\setminus k}} p(\boldsymbol{x}'^k \mid \omega_j) \times \left[\int_{S_j(\boldsymbol{x}'^k)} p(x_k \mid \boldsymbol{x}'^k, \omega_j)\, dx_k - g\{E(x_k \mid \boldsymbol{x}'^k) \in S_j(\boldsymbol{x}'^k)\} \right] d\boldsymbol{x}'^k \tag{22}$$

The replaceability[3] is another and probably better estimate of the marginal contribution of feature k.

In practice, $E(x_k|\boldsymbol{x}'^k)$ will be an estimate of the population expected value. We suggest another measure that takes into account the variability of $E(x_k|\boldsymbol{x}'^k)$ for a particular value of $\boldsymbol{x}'^k$. So we replace $g\{E(x_k|\boldsymbol{x}'^k) \in S_j(\boldsymbol{x}'^k)\}$ by a probability distribution $p(E(x_k|\boldsymbol{x}'^k)|\boldsymbol{x}'^k)$. The difference between ρ^n and the resulting ρ^{n3} we call the

[3]Note that a *high* replaceability is associated with a *small* value of ι_k and vice versa.

predicted influence of feature k (the measure resembles replaceability) and is defined as

$$\zeta_k \equiv \sum_{j=1}^{c} P(\omega_j) \int_{R_j^{\setminus k}} p(x^{\setminus k} \mid \omega_j) \times \left[\int_{S_j(x^{\setminus k})} p(x_k \mid x^{\setminus k}, \omega_j) - p(E(x_k \mid x^{\setminus k}) \mid x^{\setminus k}) \, dx_k \right] dx^{\setminus k} \tag{23}$$

The potential influence was defined to identify poor features. However, as we have seen this measure overestimates the marginal contribution of feature k, because both the extent of $S_j(x^{\setminus k})$ (when $S_j(x^{\setminus k}) \neq \mathbb{R}$) and the probability of observing values in $S_j(x^{\setminus k})$ are not taken into account: the more values observed in $S_j(x^{\setminus k})$ the less the feature can influence the classification result. For a poor feature, moreover, one may expect that the difference between the marginal distribution $p(x_k)$ and the conditional distribution $p(x_k|x^{\setminus k},\omega_j)$ will be relatively small. If we therefore replace $p(x_k|x^{\setminus k},\omega_j)$ by $p(x_k)$, we obtain again an estimate of the influence of the feature. The difference in correctness between ρ^n and the resulting ρ^{n4} we call the *expected influence* of feature k

$$\varrho_k \equiv \sum_{j=1}^{c} P(\omega_j) \int_{R_j^{\setminus k}} p(x^{\setminus k} \mid \omega_j) \left[\int_{S_j(x^{\setminus k})} p(x_k \mid x^{\setminus k}, \omega_j) - p(x_k) \, dx_k \right] dx^{\setminus k} \tag{24}$$

Although ϕ_k, ι_k, ζ_k, $\varrho_k \in (0,\rho^n)$, in practice, one would never use a statistical classifier with a correctness smaller than

$$\sum_{j=1}^{c} (P(\omega_j))^2 \tag{25}$$

the correctness of a classifier that assigns the class labels at random, taking the prior distribution into account.

4 Four feature metrics

In practice, we have to estimate ϕ_k, ϱ_k, ζ_k and ι_k from a set of cases. For each case, the range $S_j(x^{\setminus k})$ is obtained from a trained MLP. In the appendices A and B it is shown how this range can be found using a Taylor expansion. The numerical precision of the polynomial approximation is determined by ε_{max} which is derived in appendix B.

4.1 Definition of an MLP

Let $X=(\boldsymbol{x}_1,\boldsymbol{x}_2,...,\boldsymbol{x}_r)$ denote a data matrix. A vector $\boldsymbol{x}_i$, which belongs to one of c classes, represents the n feature values of case i. Let α_k and β_k denote the lower and upper limits of feature k, respectively. α_k and β_k should be the minimal and maximal values that can *possibly* be observed for feature k.

Let $\boldsymbol{o}$ denote the output vector of the MLP, $\boldsymbol{o}=N(\boldsymbol{x})$. Each element in $\boldsymbol{o}$ represents the output activation of a node in the output layer. Define a feed-forward MLP with one hidden layer with h hidden nodes as a mapping $N: \{[\alpha_1,\beta_1],..,[\alpha_n,\beta_n]\} \rightarrow [\gamma,\eta]^c$:

$$N(\boldsymbol{x}) = f(W^2 f(W^1 \boldsymbol{x} - \boldsymbol{q}^1) - \boldsymbol{q}^2) \tag{26}$$

where W^1 is the weight matrix that connects the n input nodes with the h hidden nodes and W^2 the weight matrix connecting the h hidden with the c output nodes. $\boldsymbol{q}^1$ and $\boldsymbol{q}^2$ are the bias vectors of the hidden and output nodes, respectively. $f(\boldsymbol{a})$ is the nonlinear, bounded activation function applied to each element in the vector $\boldsymbol{a}$. γ and η are the limits of the function $f(\cdot)$.

For MLPs that are used for classification tasks, each dimension in the output space generally represents a class. Usually, a case is assigned the class label that corresponds to the node with the highest output value. This winner-takes-it-all rule is defined as

$$\text{class}(\boldsymbol{o}) = \begin{cases} j \ : \ \forall l \neq j : o_l < o_j \\ \varnothing : \ else \end{cases} \tag{27}$$

In the latter case no decision can be made. $E=(\boldsymbol{e}_1,\boldsymbol{e}_2,...,\boldsymbol{e}_r)$ is a matrix that specifies the correct class labels of the corresponding vectors in X. Vector $\boldsymbol{e}_i$ contains c elements, one of which has the value 1 the other elements the value 0. The class label of $\boldsymbol{x}_i$ is j when $e_{j,i}=1$. $\boldsymbol{x}^{<k>}=[x_{k,i}]_{i=1}^{r}$ denotes a row vector that contains the observations of feature k for all cases.

4.2 Metrics

We first define a function change(·) that is used in the computation of three of the feature metrics. This function takes as arguments an input vector $\boldsymbol{x}$, its correct class label specified by $\boldsymbol{e}$ and the feature index k. The function change(·) returns the set of values of x_k given the values of $\boldsymbol{x}^{\neq k}$ for which two or more output values of $\boldsymbol{o}=N(\boldsymbol{x})$ are maximal, one of them being o_j, where j is the correct class label of vector $\boldsymbol{x}$. When x_k equals one of the values returned by change(·), the value x_k lies on the decision boundary that separates the *correct* class from a wrong class. Using

$$N'(\boldsymbol{x},k,y) = N((x_1,x_2,...,x_{k-1},y,x_{k+1},..,x_n)^T) \tag{28}$$

change(·) is defined as

$$\text{change}(\boldsymbol{x},\boldsymbol{e},k) = \left\{ y \;\middle|\; \begin{array}{l} \forall y \in [\alpha_k,\beta_k],\ j=\text{class}(\boldsymbol{e}), \\ \boldsymbol{o}=\text{N}^*(\boldsymbol{x},k,y), \\ \forall h\neq j:\ o_h \leq o_j\ , \\ \exists i\neq j:\ o_i = o_j \end{array} \right\} \tag{29}$$

Note that change(·) returns the empty set ∅ when $\forall y \in [\alpha_k,\beta_k]$: the case $\boldsymbol{x}$ always obtains the same class label or always an incorrect class label.

The *potential influence* of feature k is estimated from

$$\hat{\phi}_k = \hat{\rho} - \sum_{i=1}^{r} \frac{\text{cor}(\boldsymbol{x}_i,\boldsymbol{e}_i) \times (1 - \min\{\text{card}(\text{change}(\boldsymbol{x}_i,\boldsymbol{e}_i,k)),1\})}{r} \tag{30}$$

which may be simplified to

$$\hat{\phi}_k = \sum_{i=1}^{r} \frac{\text{cor}(\boldsymbol{x}_i,\boldsymbol{e}_i) \times \min\{\text{card}(\text{change}(\boldsymbol{x}_i,\boldsymbol{e}_i,k)),1\}}{r} \tag{31}$$

the fraction of correctly classified cases for which feature k can influence the classification. The function card(S) returns the number of elements in the set S. The function cor(·) is defined as

$$\text{cor}(\boldsymbol{x},\boldsymbol{e}) = \text{g}\{\text{class}(\text{N}(\boldsymbol{x})) = \text{class}(\boldsymbol{e})\} \tag{32}$$

and is 1 when the vector $\boldsymbol{x}$ is classified correctly, otherwise cor(·) is 0. $\hat{\rho}$ is the estimated correctness using all n features. The metric $\hat{\phi}_k$ estimates ϕ_k from the sample X using the MLP $\text{N}(\boldsymbol{x})$. $\hat{\phi}_k$ is binomially distributed.

Let's define the function z(·) that takes as input vector $\boldsymbol{x}$, the vector $\boldsymbol{e}$ that specifies the correct class label of $\boldsymbol{x}$ and feature index k

$$z(\boldsymbol{x},\boldsymbol{e},k) = \begin{cases} \varnothing & : \text{change}(\boldsymbol{x},\boldsymbol{e},k)=\varnothing,\ \text{class}(\text{N}(\boldsymbol{x}))\neq\text{class}(\boldsymbol{e}) \\ \{[\alpha_k,\beta_k]\} & : \text{change}(\boldsymbol{x},\boldsymbol{e},k)=\varnothing,\ \text{class}(\text{N}(\boldsymbol{x}))=\text{class}(\boldsymbol{e}) \\ \{\boldsymbol{s}(d)\} & : \forall d:\ \boldsymbol{s}(d)=[l_d,h_d] \mid l_d,h_d \in \{\alpha_k,\beta_k\} \cup \text{change}(\boldsymbol{x},\boldsymbol{e},k), \\ & \quad \forall y \in [l_d,h_d]:\ \text{class}(\text{N}^*(\boldsymbol{x},k,y)) = \text{class}(\boldsymbol{e}) \end{cases} \tag{33}$$

z(·) returns an ordered set of intervals $S=\{\boldsymbol{s}(1),\boldsymbol{s}(2),..,\boldsymbol{s}(t)\}$, $\boldsymbol{s}(d)=[l_d,h_d]$, $l_d<h_d$, $l_d,h_d \in [\alpha_k,\beta_k]$, $d=1,..,t$, and $h_d<l_{d+1}$, $d=1,..,t-1$. S is an estimate of the range $S_j(x^{*k})$ defined in Eq. (9). The vector $\boldsymbol{x}$ obtains the correct class label as specified by $\boldsymbol{e}$ for any value of $x_{k,i}$ that occurs in one of the intervals $[l_d,h_d]$ in S. If the vector $\boldsymbol{x}$ is assigned the correct class label for all valid values of x_k z(·) returns $\{[\alpha_k,\beta_k]\}$. If $\boldsymbol{x}$ is assigned the wrong class label for all valid values of x_k, z(·) returns the empty set.

The metric for the *expected influence* ϱ_k is computed from

The parameters of the regression $\boldsymbol{b}^k$ used to compute the replaceability ι_k have a stochastic component. The matrix for the predicted influence takes this into account. For normally distributed features $p(\boldsymbol{x})$, the predicted values $\hat{\boldsymbol{x}}^{<k>} = \boldsymbol{b}^{k\,T}\boldsymbol{g}_i^k$ (the conditional means) are t-distributed around their true mean $\bar{\boldsymbol{x}}^{<k>} = \beta^{k\,T}\boldsymbol{g}_i^k$ with the variance [21]

$$\hat{V} = \hat{\sigma}^2_{res,k}\,\boldsymbol{g}^k(G_k G_k^T)^{-1}\boldsymbol{g}^{k\,T} \tag{41}$$

and $r-n$ degrees of freedom. $\hat{\sigma}^2_{res,k}$ is the residual variance of the regression estimated from

$$\hat{\sigma}^2_{res,k} = \frac{\|\hat{\boldsymbol{x}}^{<k>} - \boldsymbol{x}^{<k>}\|^2}{r-n} \tag{42}$$

and $\|\cdot\|$ the Euclidian vector norm.

The probability of observing $\hat{x}_{k,i}$ in the range S can be estimated as

$$pc(S,\hat{x}_k,r-n) = \sum_{x(d)\in S} T\left(\frac{l_d-\hat{x}_k}{\hat{V}}, \frac{h_d-\hat{x}_k}{\hat{V}}, r-n\right) \tag{43}$$

with T the cumulative t-distribution [24]

$$T(a,b,df) = \int_a^b \frac{1}{\sqrt{\pi df}}\,\frac{\Gamma((df+1)/2)}{\Gamma(df/2)}\left(1+\frac{x^2}{df}\right)^{-(df+1)/2} dx \tag{44}$$

where df is the number of the degrees of freedom and $\Gamma(\cdot)$ is the Gamma function.

The *predicted influence* of feature k can now be estimated from

$$\hat{\zeta}_k = \hat{\rho} - \sum_{i=1}^{r} \frac{pc(z(\boldsymbol{x}_i,\boldsymbol{e}_i,k),\boldsymbol{b}^{k\,T}\boldsymbol{g}_i^k,r-n)}{r} \tag{45}$$

$\hat{\zeta}_k$ is an unbiased estimator of ζ_k when the n features are multivariate normally distributed. Of course in practical situations this is hardly ever the case and the practical value of the metric has to be established empirically.

4.3 Properties of the feature metrics

Two of the metrics, $\hat{\iota}_k$ and $\hat{\zeta}_k$, are based on the assumption that the features are normally distributed. Often this is not the case, so we briefly investigate their relation to the (nonparametric) metric $\hat{\phi}_k$.

Theorem 4.3.1. For an MLP, N, the potential influence is greater than or equal to the replaceability $\hat{\phi}_k \geq \hat{\iota}_k$.

Proof

The inequality $\hat{\phi}_k \geq \hat{\iota}_k$ may be written as

$$\hat{\rho} - \sum_{i=1}^{r} \frac{\text{cor}(\boldsymbol{x}_i,\boldsymbol{e}_i) \times (1 - \min\{\text{card}(\text{change}(\boldsymbol{x}_i,\boldsymbol{e}_i,k)),1\})}{r} \geq \hat{\rho} - \sum_{i=1}^{r} \frac{g\{\text{class}(N^*(\boldsymbol{x}_i,k,\boldsymbol{b}^{k\,T}\boldsymbol{g}_i^k)) = \text{class}(N^*(\boldsymbol{e}_i)\}}{r} \tag{46}$$

It can be proven that for case i

$$\frac{\text{cor}(\boldsymbol{x}_i,\boldsymbol{e}_i) \times (1 - \min\{\text{card}(\text{change}(\boldsymbol{x}_i,\boldsymbol{e}_i,k)),1\})}{r} \leq \frac{g\{\text{class}(N^*(\boldsymbol{x}_i,k,\boldsymbol{b}^{k\,T}\boldsymbol{g}_i^k)) = \text{class}(N^*(\boldsymbol{e}_i)\}}{r} \tag{47}$$

as the first fraction is 1 if and only if the case $\boldsymbol{x}_i$ is classified correctly and feature k has no influence on its classification. Consequently, feature $x_{k,i}$ can be replaced by any value $\hat{x}_{k,i} \in [\alpha_k,\beta_k]$. In this case the second fraction is also one. This second fraction can also be 1 when the first fraction is 0. This happens when feature k has potential influence on the classification of the case (change(·) returns a nonempty set) and the prediction of $x_{k,i}$ lies in the corresponding range S. Hence

$$\sum_{i=1}^{r} \frac{\text{cor}(\boldsymbol{x}_i,\boldsymbol{e}_i) \times (1 - \min\{\text{card}(\text{change}(\boldsymbol{x}_i,\boldsymbol{e}_i,k)),1\})}{r} \leq \sum_{i=1}^{r} \frac{g\{\text{class}(N^*(\boldsymbol{x}_i,k,\boldsymbol{b}^{k\,T}\boldsymbol{g}_i^k)) = \text{class}(N^*(\boldsymbol{e}_i)\}}{r} \tag{48}$$

and $\hat{\phi}_k \geq \hat{\iota}_k$ ■

Theorem 4.3.2. The metrics for the replaceability and the likely influence $\hat{\iota}_k$ and $\hat{\zeta}_k$ approach each other when the variance of the prediction $\hat{V} \to 0$. In this situation, both $\hat{\iota}_k$ and $\hat{\zeta}_k$ approach 0.

Proof

The probability that the true conditional mean is observed within the interval $[l_d,h_d]$ given by S is computed from

$$\mathrm{T}\left(\frac{l_d-\hat{x}_k}{\hat{\mathrm{V}}},\frac{h_d-\hat{x}_k}{\hat{\mathrm{V}}},df\right),\quad [l_d,h_d]=s(d)\in \mathrm{S} \tag{49}$$

As $\hat{\mathrm{V}} \to 0$, the integral over the t-distribution approaches 1 when $\hat{x}_k \in [l_d,h_d]$ for a $d \in \{1,..,t\}$ and 0 otherwise.
The probability pc(S,$\hat{x}_k$,*df*), approaches g{class(N'($\boldsymbol{x}_i$,k,$\boldsymbol{b}^{k\,\mathrm{T}}\boldsymbol{g}_i^k$))=class($\boldsymbol{e}_i$)}, which is 1 if $\hat{x}_k \in [l_d,h_d]$ and 0 otherwise, and hence $\mathfrak{t}_k \to \zeta_k$.

$\hat{\mathrm{V}}$ can only be zero in the situation where $\boldsymbol{g}^k=\boldsymbol{0}$ or when $\hat{\sigma}^2_{res,k}=0$, the predicted feature is linearly dependent of the other $n-1$ features. In this situation, $\mathfrak{t}_k$ and ζ_k are both zero. ■

Theorem 4.3.1 derives a bound for the replaceability in relation to the potential influence $\hat{\phi}_k \geq \mathfrak{t}_k$. Theorem 4.3.2 shows that as the prediction variance decreases, the two metrics $\mathfrak{t}_k$ and ζ_k will become alike. How close the two feature metrics are, is among other things investigated in the experiments to follow.

5 Feature pruning

The four feature metrics can be used as criteria to select features to be pruned from an MLP. Each of the four feature metrics defined in the previous section estimates a bound for the marginal contribution of a feature. The ability of the metrics to rank features is investigated in the experiments that follow.

We developed a technique for pruning an input node from a trained MLP which in many situations makes retraining superfluous.

Let us define LMS pruning:

Definition 5.1. Least Mean Square (LMS-) pruning of a feature k from an MLP N consists of creating an MLP N′ identical to N but without input node k. The weights of N′ that connect the n−1 input nodes with the h hidden nodes as well as their bias terms obtain values such that N′ classifies a set of cases identically as N does when feature k is replaced by $\hat{\boldsymbol{x}}^{<k>}$, its LMS-predicted value.

LMS-pruning is obtained as follows:
Assume for simplicity that input n is to be pruned from N. Define $\boldsymbol{a}$ as the input vector to the hidden nodes before the activation function f(·) is applied, see Eq. (26):

$$\boldsymbol{a}=\mathrm{W}^1\boldsymbol{x}-\boldsymbol{q}^1 \tag{50}$$

From X, compute the regression parameter vector $\boldsymbol{b}^n$ – see Eq. (39) – and split it into the coefficient vector $\boldsymbol{b}^{nc}$ and the constant term $b_n{}^n$, $\boldsymbol{b}^n=(\boldsymbol{b}^{nc\,\mathrm{T}},b_n{}^n)^{\mathrm{T}}$. We replace x_n in (50) by its predicted value $\hat{x}_{n,i}$ (for case i) computed from

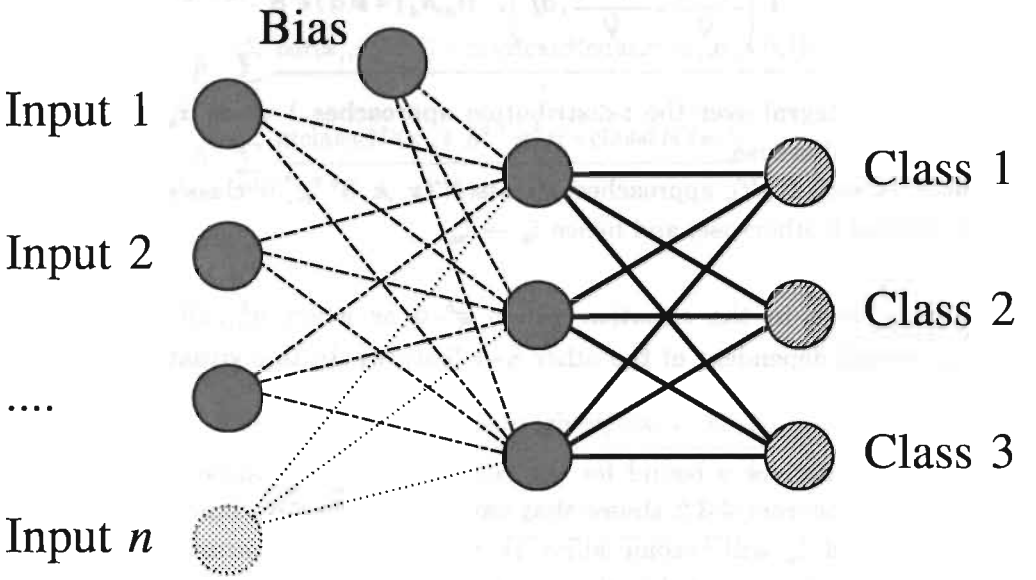


Figure 1. The weights on the dotted connections are removed by LMS-pruning. The dashed weights are modified.

$$\hat{x}_{n,i} = b_n^n + \sum_{j=1}^{n-1} b_j^{nc} x_{j,i} \tag{51}$$

Combining (50) and (51) gives for hidden node u

$$a_u = \sum_{j=1}^{n-1} w_{u,j}^1 x_{j,i} + q_u^1 + w_{u,n}^1 \left(b_n^n + \sum_{j=1}^{n-1} b_j^{nc} x_{j,i} \right) \tag{52}$$

which simplifies to

$$a_u = \sum_{j=1}^{n-1} (w_{u,j}^1 + w_{u,n}^1 b_j^{nc}) x_{j,i} + (q_u^1 + w_{u,n}^1 b_n^n) \tag{53}$$

Define $\Xi_{u,j}^1 = (w_{u,j}^1 + w_{u,n}^1 b_j^{nc})$, $\gamma_u^1 = q_u^1 + w_{u,n}^1 b_n^n$ and construct a new MLP N′ that has $n-1$ input nodes and the same number of hidden nodes as N with the weight matrix W^2, the bias vector $\boldsymbol{q}^2$, and the new weight matrix Ξ^1 and bias vector γ^1. Now feature $\boldsymbol{x}^{<n>}$ has been LMS-pruned from N.

Figure 1 illustrates which weights are updated (dashed) and which are pruned (dotted) when feature n is pruned. The pruning operation turns out to be useful because the following holds:

Corollary 5.2. A network N with a correctness $\hat{\rho}$ will, when feature k is LMS-pruned, have a correctness $\rho' = \hat{\rho} - \hat{\iota}_k$.

This corollary specifies a *lower bound* for the correctness of a network from which feature k has been pruned as it is possible to retrain the LMS-pruned network using the weights of N′ as initial weight configuration and thereby possibly improve its correctness.

6 Experiments

We conducted a set of experiments to assess the developed metrics and the pruning method introduced in section 5. We constructed two artificial classification problems to investigate whether the features were ranked correctly by each of the feature metrics.

First experiment

In the first problem two classes A and B were characterized by 6 features with the centres $\boldsymbol{\mu}_A=(0,0,0,0,0,0)^T$ and $\boldsymbol{\mu}_B=(1.75,1.50,1.25,1.00,0.75,0.50)^T$, respectively. Feature 1 has the largest discriminative power, feature 6 the smallest. We sampled 500 uncorrelated observations from $N(\boldsymbol{x}|\boldsymbol{\mu}_A,I)$ and 500 from $N(\boldsymbol{x}|\boldsymbol{\mu}_B,I)$ with I the identity matrix. The observations were divided into a training set and a test set each containing 250 vectors from class A and 250 from class B.

In total 30 MLPs with 2 hidden nodes, all with different initial weight configurations, were trained for 700 cycles with Back-Propagation in offline mode. The average correctness of the MLPs for the test set was $\rho_{avg}=0.9274$ (±0.0027). This correctness is very close to the Bayesian correctness, $\rho_{bayes}=0.9292$.

We used Kendall's measure T_c for the correlation between several judges and a criterion ranking [36] to compare the true (criterion) ranking of the 6 features with the ranking obtained from each feature metric. T_c in table 1 provides the average rank order correlation between the 30 MLPs and the true ranking (1,..,6) which follows directly from the population parameters $\boldsymbol{\mu}_A$ and $\boldsymbol{\mu}_B$.

	T_c(pot. infl.)	T_c(exp. infl.)	T_c(pred. infl.)	T_c(repl.)
$\varepsilon_{max}=0.01$	0.813	1.000	0.884	0.920
$\varepsilon_{max}=0.0001$	0.778	1.000	0.924	

Table 1. The rank correlations T_c between the feature raking of the 30 MLPs and the true ranking. These are computed for two numerical precision levels ε_{max} of the polynomial approximation.

The first row in table 1 shows that potential influence is the poorest ranking criterion whereas the expected influence resulted in an optimal ranking. The latter is to be expected as the features are independent (within the two classes). The predict-

ed influence and the replaceability are slightly worse ranking criteria. An analysis of the weights of the MLPs indicated that feature 6 was given a larger weight than feature 5 in most of the 30 MLPs.

Second experiment

In the second experiment, we investigated the influence of the numerical precision ε_{max} on the feature metrics.

We recomputed all feature metrics except the replaceability ι_k (because ι_k does not depend on ε_{max}) using the 30 MLPs from the first experiment with a higher precision level for the polynomial approximation, ε_{max}=0.0001. The second row in table 1 shows the coefficient of agreement T_c between the true raking and the average ranking assigned by each metric to the features in the 30 MLPs with the increased precision level. Only the agreement between the predicted influence and the true rank slightly improves.

The feature metric that was influenced most by the level of precision is the potential influence. Figure 2 shows the relative discrepancies between the potential influence computed for the six features, for both prediction levels, ε_{max}=0.01 and ε_{max}=0.0001, $[\phi_k(\varepsilon_{max}=0.01)-\phi_k(\varepsilon_{max}=0.0001)]/\phi_k(\varepsilon_{max}=0.01)$. Table 2 shows the potential influence of the six features.

Precision	1	2	3	4	5	6
0.01	0.865	0.795	0.799	0.519	0.117	0.135
0.0001	0.727	0.647	0.663	0.462	0.111	0.134

Table 2. Potential influence computed for the two different levels of precision for each of the six features.

The discrepancies between $\phi_k(\varepsilon_{max}=0.01)$ and $\phi_k(\varepsilon_{max}=0.0001)$ become small when the features are unimportant. This is also to be expected. For unimportant features, small fluctuations of the polynomial approximation round the true difference in output o_j-o_l are unlikely to lead to false zero crossings, because o_j-o_l is in more cases unequal to zero when feature x_k is varied within its range.

We investigated the correlation between some of the feature metrics. The correlations in table 3 indicate that the replaceability and the predicted influence metrics are closely related, which is also to be expected from their definition. Also the expected and the predicted influence are correlated. The potential influence is almost independent of the two other influence metrics.

Third experiment

A third experiment was designed to investigate how effective LMS-pruning is and to compare the ranking of each metric with the true ranking when the variables

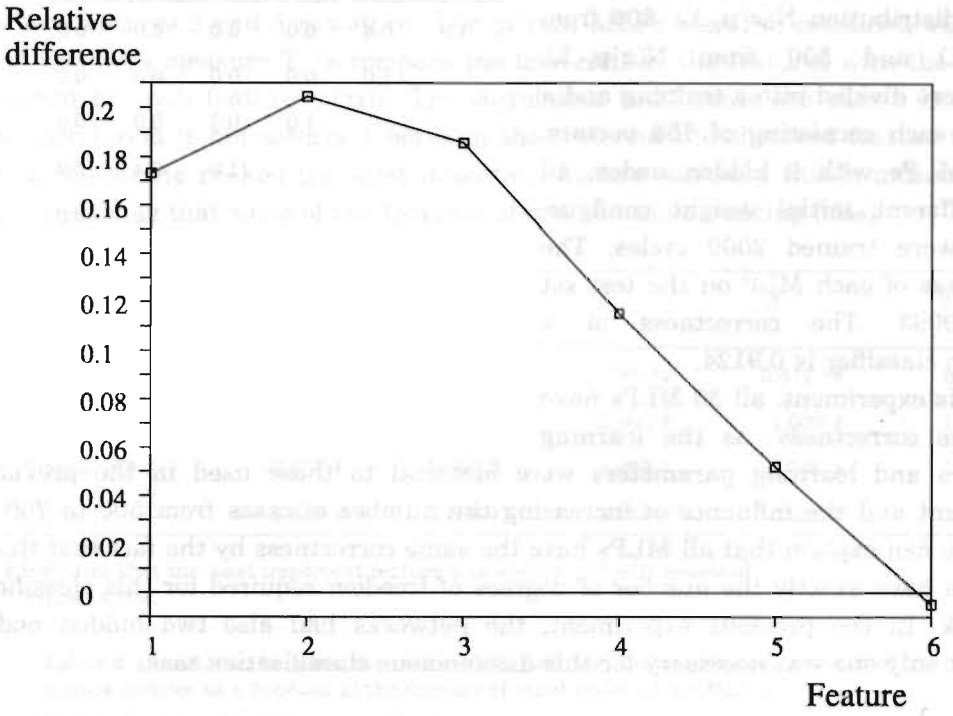


Figure 2. Average relative difference between the potential influence as computed with the precision level ε_{max}=0.01 and ε_{max}=0.0001.

Feature	Pot. vs. exp. influence	Exp. vs. pred. infl.	Pred. infl. vs. repl.	Pot. vs. pred. infl.
1	0.000	0.117	0.247	0.006
2	0.280	0.295	0.515	0.123
3	0.003	0.565	0.295	0.004
4	0.380	0.945	0.966	0.364
5	0.045	0.806	0.919	0.000
6	0.001	0.301	0.640	0.102
Avg.	0.118	0.505	0.597	0.100

Table 3. Correlations between the four feature metrics computed among the 30 MLPs with the precision level used in the second experiment.

contain dependencies. We designed a classification problem with three classes A, B and C that are characterized by six attributes. The centra of A and B were identical to the previous experiments and $\mu_C=-\mu_B$. The three classes have identical covariance matrices $\Sigma_A=\Sigma_B=\Sigma_C$, see table 4.

We sampled 500 vectors from the normal distribution $N(\boldsymbol{x}|\boldsymbol{\mu}_A,\Sigma)$, 500 from $N(\boldsymbol{x}|\boldsymbol{\mu}_B,\Sigma)$ and 500 from $N(\boldsymbol{x}|\boldsymbol{\mu}_C,\Sigma)$. These were divided into a training and a test set each consisting of 750 vectors. Thirty MLPs with 2 hidden nodes, all with different initial weight configurations, were trained 2000 cycles. The correctness of each MLP on the test set was 0.9093. The correctness of a Bayesian classifier is 0.9124.

1.0	0.4	0.0	0.0	0.0	0.0
	1.0	0.0	0.0	-0.3	0.0
		1.0	0.3	0.0	0.0
			1.0	-0.4	0.7
				1.0	0.0
					1.0

Table 4. The correlation matrix used in the third experiment.

In this experiment, all 30 MLPs have the same correctness. As the learning algorithm and learning parameters were identical to those used in the previous experiment and the influence of increasing the number of cases from 500 to 750 is small, we can explain that all MLPs have the same correctness by the fact that these networks have exactly the number of degrees of freedom required for this classification task. In the previous experiment, the networks had also two hidden nodes although only one was necessary for this dichotomous classification task.

Feature:	1	2	3	4	5	6
M. contribution	0.0125	0.0426	0.0011	0.0398	0.0643	0.0137
Ranking	5	2	6	3	1	4
M. contribution	0.0123	0.0473		0.0785	0.0910	0.0227
Ranking	5	3		2	1	4
M. contribution		0.1371		0.1073	0.1401	0.0302
Ranking		2		3	1	4
M. contribution		0.1504		0.0872	0.1137	
Ranking		1		3	2	
M. contribution		0.2534			0.0838	
Ranking		1			2	

Table 5. The true marginal contributions and feature rankings for a Bayesian classifier after successively removing the least contributing feature.

Table 5 contains the marginal contribution of each feature for a Bayesian classifier and the true ranking of the attributes. The feature with the smallest marginal contribution is the correct feature to prune.

The four feature metrics were used to estimate the importance of each feature among the 30 MLPs using the set of training vectors. The most replaceable feature (smallest ι_k) was LMS-pruned and the importance of the five remaining features was

estimated among the 30 LMS-pruned MLPs. This procedure was continued until only the two features 2 and 5 remained. The pruned MLPs were not retrained. Again, we used Kendall's measure T_c to compare the true rank of the features with the ranking obtained by each feature metric. The correlation coefficients are shown in table 6. The correlation is not always 1 between the true and the observed feature ranking. When the metric ranked the least important feature correctly, this is indicated with '#'; '*' indicates that some of the features obtain the same ranking (ties).

Features contained	6	5	4	3	2
T_c(pot. infl.)	0.867 #	0.738 *	0.707 *#	0.817 *#	0.000 *
T_c(exp. infl.)	0.733 #	0.600	0.000 #	1.000 #	1.000 #
T_c(pred. infl.)	0.867 #	1.000 #	1.000 #	1.000 #	1.000 #
T_c(repl.)	0.867 #	1.000 #	1.000 #	1.000 #	1.000 #

# indicates that the least important feature was always correctly assessed.
* indicates ties.

Table 6. Correlation between the true ranking and the ranking obtained by each of the four feature metrics as a function of the number of input nodes of the MLPs.

Table 6 indicates that the metrics for the predicted influence and the replaceability are superior to the other two metrics. Only in the situation with six features, not all features are ranked correctly. In this case, the features 4 and 5 are ranked as (5,4) leading to a T_c of 0.867. Another observation is that the potential influence produces ties when the number of features is below 6. So when the classification relies on a few features, their contribution can only be assessed by taking the probability density function of the features into account. The expected influence only performed well when the number of features was reduced to 3. For the configuration with 5 features, the expected influence metric does not identify the feature with the smallest marginal contribution. We conclude that in this experiment where the features contain dependencies, the predicted influence and replaceability are the best ranking criteria.

The replaceability of a feature is an estimate of its marginal contribution. Figure 3 is a plot with as abscissa the marginal contribution of a feature and as ordinate the relative deviation in percent between the marginal contribution of the feature (=100%) and the corresponding replaceability estimated on the training set. The deviation decreases as the marginal contribution becomes large. The average deviation is 18%.

Figure 4 shows the decrease in the average correctness among the 30 MLPs when features are LMS-pruned. The correctness is estimated with a test set that also contains 750 cases. The pruning method is effective as the difference between the observed and theoretical correctness remains small.

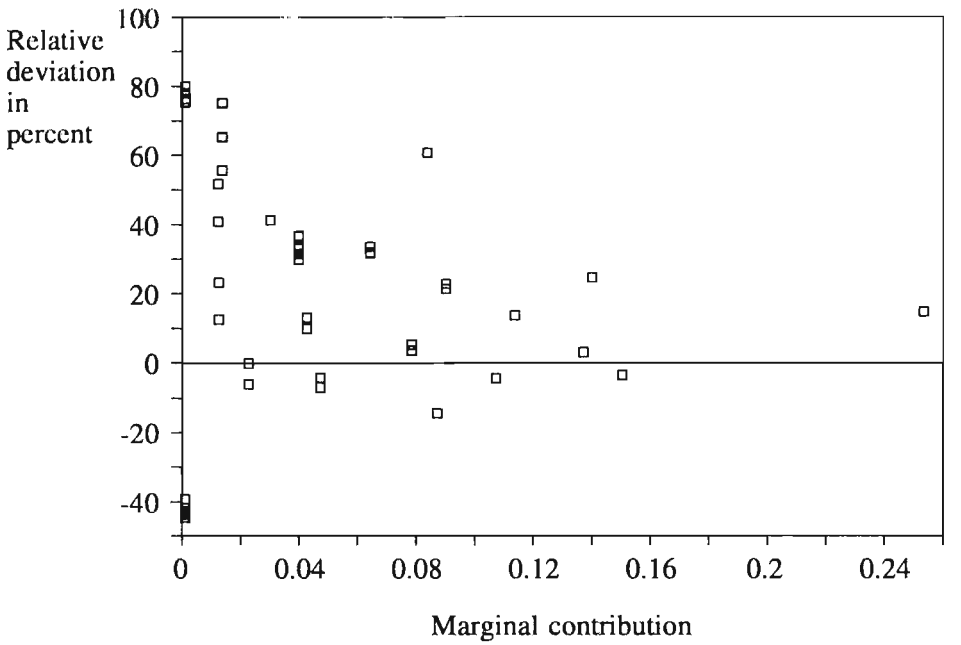


Figure 3. Relative deviation between true marginal contribution and the replaceability computed on the training set.

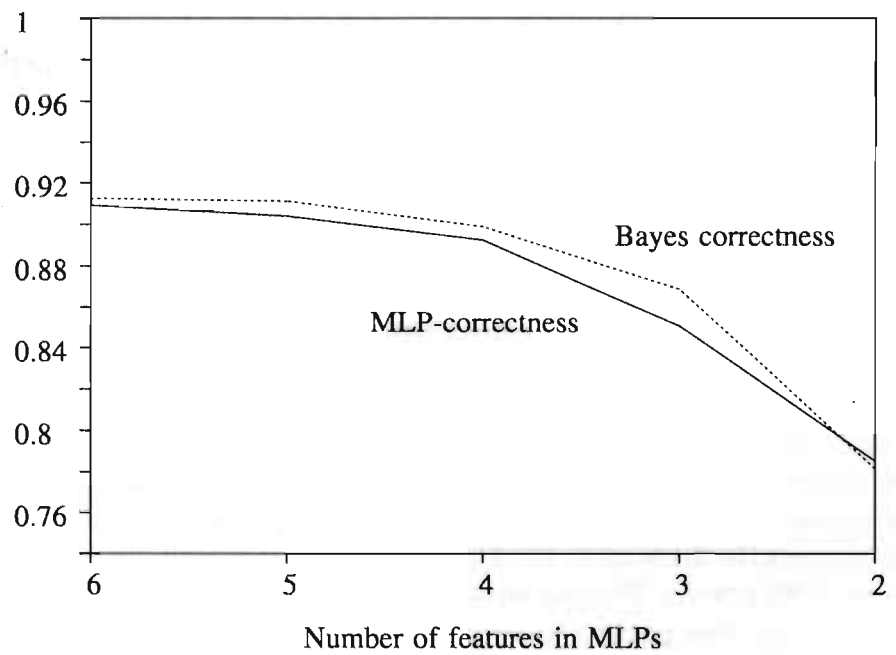


Figure 4. The average decrease in correctness among the 30 MLPs lies close to the Bayes-optimal correctness.

7 Discussion

Four measures were defined to assess the importance of a feature for a classifier that uses n features. The four measures were made operational by metrics. Each metric can be used to estimate the importance of a feature for an MLP and can be used in e.g. a backward search procedure for pruning features.

One could ask whether all four metrics are needed to assess the importance of features. In our experiments, the replaceability and the predicted influence are the best ranking criteria when the features contain dependencies and the expected influence the best criterion when the features are uncorrelated.

The estimates computed with the four metrics all have a certain variance. Whether the difference in importance between two features is significant or not, should therefore be tested. The metrics for potential influence and replaceability are binomially distributed and a χ^2-test can be used to compare the importance of two features.

The potential influence metric can be valuable as it makes it possible to identify subsets of cases for which a feature has no influence on their classification. So this metric may enrich the understanding of the classification task. The metric can also aid the construction of classifiers for sequential classification tasks. Quinlan distinguishes between *sequential* and *parallel* classification tasks [28]. In parallel classification tasks all features are relevant for the classification of each case. In sequential classification tasks only a few of the available features determine the class label for a specific case. Whether a feature is relevant when classifying a specific case, depends on the value of one (or more) of the other features. When an MLP has been trained for a classification task, the potential influence metric can be used to identify features that are only (potentially) relevant for a small subset of cases. The least important of the n features can then be LMS-pruned. The procedure can be repeated for $n-2$ features, etc. Thereby, the potential influence metric helps to establish the order in which features can be used by a sequential classifier, e.g. a cascade of MLPs. Building such a cascaded MLP classifier is, however, not trivial as the networks that are based on only a subset of features should be able to leave cases unclassified that can only be classified correctly using additional features.

The major advantage of LMS-pruning is that one can prune a feature from a good MLP without having to train its weights from scratch. The amount of computation needed by a backward search is reduced as one needs not to train a set of networks with different initial weight configurations for each combination of $n-1$ features. We conclude that LMS-pruning is a convenient and computationally simple procedure to remove input nodes from an MLP. When a good subset of features has been identified, one can always try to retrain the MLP and possibly improve its performance. Our approach does not take into account that the number of hidden nodes that is optimal when using n features may not be optimal for $n-1$ features. How to prune hidden nodes, is left as a topic for further research.

The overall correctness of a classifier is one of many possible yardsticks that can be used to measure the importance of a feature. If one wants an assessment that is

independent of the prior probability of each class the class-conditional correctness can be used as criterion [7]. Class-conditional variants of our feature metrics can easily be computed by summing only over cases that belong to a given class.

We developed a numerical approach based on Taylor expansions to solve the $c-1$ equations that determine the values of each feature for which two outputs of the MLP are equal. The polynomial approach solves the equations with sufficient accuracy but is computationally heavy as a different set of polynomial coefficients has to be computed for each feature k in each feature vector $\boldsymbol{x}_i$. Laguerre's method, which is used to find all roots in each polynomial, is also computationally complex. For one MLP with six input nodes, two hidden and three output nodes, the computation time for 750 vectors was 18 minutes on a Pentium-133 PC.

When one wants to prune features, we suggest to use the replaceability as a raking criterion. This metric gives directly the correctness of the pruned MLP and the Taylor expansion is not needed to compute it.

8 Conclusion

We derived a mathematical framework in which four measures for the importance of a feature for a classifier are developed. These measures are related to the marginal contribution of a feature. For each measure, we defined a metric to assess the importance of features for a multi-layer perceptron. It was suggested to use the metrics as ranking criteria to identify features from a trained MLP in a backward search. The feature with the smallest contribution is removed from the MLP using LMS-pruning.

A polynomial approximation was developed to identify the values of feature k for which two outputs of the network were maximal. These values together specify the range of feature k, given the values of the other $n-1$ features, that results in a correct classification of a case. The polynomial approximation was used to compute three of the feature metrics.

Experiments illustrated that using LMS-pruning in combination with a backward search strategy enabled us to prune features from an MLP in an efficient way. The error rate obtained after a feature was LMS-pruned deviated only little from the Bayesian error rate. So in our experiments retraining the pruned network from scratch could be avoided. The expected influence resulted in the best ranking of features when they are class-conditionally independent whereas the metrics for replaceability and predicted influence were superior when the features are dependent.

Appendix A

In section 4 we defined the function change(·). For a specific vector $\boldsymbol{x}$, this function returns values of feature k for which more than one element of the output $\boldsymbol{o}=\mathrm{N}(\boldsymbol{x})$ of the MLP N has the maximal value.

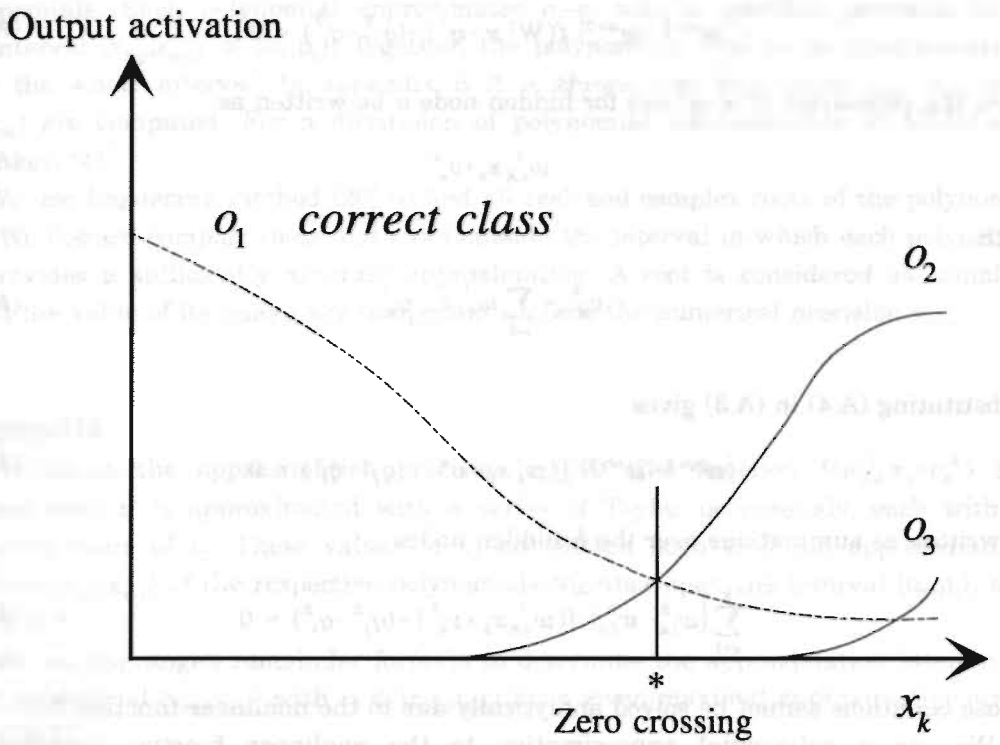


Figure A.1. A case can be classified as class 1 or 2 depending on x_k. The function change(•) returns the value of x_k for which $o_1=o_2$.

To evaluate the function change(·), we need to identify the values of feature k in $[\alpha_k,\beta_k]$ that cause two output nodes to be maximal including the node that represents the correct class of the case. Given the vector $\boldsymbol{x}$, all values except x_k are kept fixed which allows us to write o_j-o_l as a function of x_k. The roots of this equation comprise the values of feature k we seek. As all nodes different to j can be maximal, in total $c-1$ equations need to be solved $o_j-o_l=0$, $\forall l\neq j$. The subset of roots occurring in the interval $[\alpha_k,\beta_k]$ for which $o_j=\max(\boldsymbol{o})$, $j=\text{class}(\boldsymbol{e})$, $\forall l\neq j$, constitute the set of values to be returned by change(·).

As output value o_j is computed from

$$o_j = \mathrm{f}\left(\boldsymbol{w}^{<j>2}\,\mathrm{f}(\mathrm{W}^1\boldsymbol{x}-\boldsymbol{q}^1)-q_j^{\,2}\right) \tag{A.1}$$

the $c-1$ equations can be written as

$$\mathrm{f}\left(\boldsymbol{w}^{<j>2}\,\mathrm{f}(\mathrm{W}^1\boldsymbol{x}-\boldsymbol{q}^1)-q_j^{\,2}\right)-\mathrm{f}\left(\boldsymbol{w}^{<l>2}\,\mathrm{f}(\mathrm{W}^1\boldsymbol{x}-\boldsymbol{q}^1)-q_l^{\,2}\right)=0 \tag{A.2}$$

for $l\neq j$. Solutions to these equations are called *zero crossings* (see figure A.1). As f(·) is a monotonous transformation and f(0)=0, (A.2) simplifies to

$$\left(\boldsymbol{w}^{<j>2} - \boldsymbol{w}^{<l>2}\right) \mathrm{f}(\mathrm{W}^1 \boldsymbol{x} \quad \boldsymbol{q}^1) - (q_j^{\,2} - q_l^{\,2}) = 0 \tag{A.3}$$

Now, the expression $\mathrm{W}^1\boldsymbol{x}-\boldsymbol{q}^1$ can for hidden node u be written as

$$w_{u,k}^1 x_k + v_u^{\,k} \tag{A.4}$$

with

$$v_u^{\,k} = \sum_{i \neq k} w_{u,i}^1 x_i - q_u^1 \tag{A.5}$$

Substituting (A.4) in (A.3) gives

$$\left(\boldsymbol{w}^{<j>2} - \boldsymbol{w}^{<l>2}\right) \mathrm{f}(\boldsymbol{w}_k^1 x_k + \boldsymbol{v}^k) - (q_j^{\,2} - q_l^{\,2}) = 0 \tag{A.6}$$

or written as summations over the h hidden nodes

$$\sum_{u=1}^{h} \left(w_{j,u}^2 - w_{l,u}^2\right) \mathrm{f}(w_{u,k}^1 x_k + v_u^{\,k}) - (q_j^{\,2} - q_l^{\,2}) = 0 \tag{A.7}$$

These equations cannot be solved analytically due to the nonlinear function f(·).

We use a polynomial approximation to the nonlinear function specified as $\mathrm{f}(x_k) = \tanh(w_{u,k}^1 x_k + v_u^{\,k})$. Its Taylor expansion is given by

$$\mathrm{P}_u(x_k) = \sum_{n=0}^{\infty} \frac{(x_k - x_{0k})^n}{n!} \mathrm{f}^{(n)}(x_{0k}) \tag{A.8}$$

We incorporate the constant x_{0k} into the coefficients of the polynomial using the Binomial theorem

$$(a-b)^n = \sum_{t=0}^{n} \binom{n}{t} a^t (-b)^{n-t} \tag{A.9}$$

The tth coefficient (coefficient of $(x_k)^t$) of the polynomial $\mathrm{P}_u(x_k)$ becomes

$$\psi_{u,t} = \sum_{n=t}^{\infty} \frac{\binom{n}{t} (-x_{0k})^{n-t}}{n!} \mathrm{f}^{(n)}(x_{0k}) \tag{A.10}$$

The coefficients of the polynomial expansions are summed over the hidden nodes to obtain one polynomial that approximates the $c-1$ equations $o_j-o_l=0$, $l \neq k$

$$\sum_{t=0}^{\infty} \left(\sum_{u=1}^{h} (w_{j,u}^2 - w_{l,u}^2) \psi_{u,t} \right) (x_k)^t - (q_j^{\,2} - q_l^{\,2}) = 0 \tag{A.11}$$

In practice, the degree of the polynomial expansion has to be limited. We have set the maximal degree to 4 and approximate o_j-o_l with a number of concatenated

polynomials. Each polynomial approximates o_j–o_l with a specified precision in a subinterval $[x_{beg},x_{end}]$ of $[\alpha_k,\beta_k]$. Together, the polynomials provide an approximation over the whole interval. In appendix B it is shown how the values x_{beg}, x_{end} and $f^{(n)}(x_{0k})$ are computed. For a discussion of polynomial approximation of MLPs see reference [41].

We use Laguerre's method [26] to find all real and complex roots of the polynomials. We discard complex roots and roots outside the interval in which each polynomial provides a sufficiently accurate approximation. A root is considered as complex when the value of its imaginary component exceeds the numerical precision ε_{max}.

Appendix B

We introduce the approximation precision $\varepsilon_{max}>0$. The function $f(w_{u,k}^{1} x_k + v_u^{k})$ for hidden node u is approximated with a series of Taylor polynomials, each with a different value of x_0. These values of x_0 are chosen such that the approximation interval $[x_{beg},x_{end}]$ of the respective polynomials together span the interval $[\alpha_k,\beta_k]$, see figure B.1.

We use Lagrange's remainder formula to determine the approximation interval of each polynomial $[x_{beg},x_{end}]$ with $x_0 \in [x_{beg},x_{end}]$ for a given maximal approximative error ε_{max} [29,37]:

$$\varepsilon_{max} \le \frac{|x - x_0|^{n+1}}{(n+1)!} M \qquad \text{(B.1)}$$

where M is the maximal absolute value of $f^{(n+1)}(x)$ $\forall x \in I$, (for simplicity $I=\mathbb{R}$). For a fixed x_0, when ε_{max} is specified, we may determine x_{beg} and x_{end} of the polynomial that guarantees an error smaller than ε_{max} by rearranging (B.1)

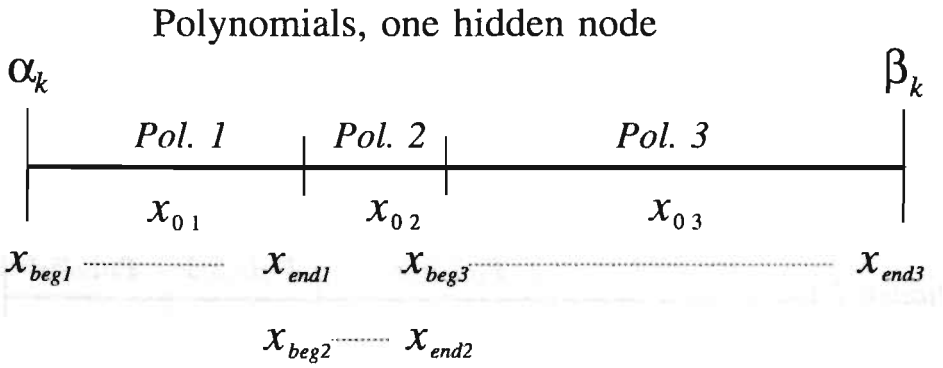


Figure B.1. The coefficients of the consecutive polynomials are chosen such that $\forall x \in [\alpha_k,\beta_k]$: $\varepsilon \le \varepsilon_{max}$.

$$\left(\frac{\varepsilon_{max}(n+1)!}{M}\right)^{(n+1)} \geq |x \quad x_0| \tag{B.2}$$

Now solving for x_{beg}

$$x_0 - \left(\frac{\varepsilon_{max}(n+1)!}{M}\right)^{(n+1)^{-1}} = x_{beg} \tag{B.3}$$

and x_{end}

$$x_0 + \left(\frac{\varepsilon_{max}(n+1)!}{M}\right)^{(n+1)^{-1}} = x_{end} \tag{B.4}$$

The extreme value M can be found by solving $f^{(n+2)}(x)=0$ and choosing the root that maximizes $|f^{(n+1)}(x)|$. The nth derivative $f^{(n)}(x)$ is defined as

$$f^{(n)}(x_0) = \frac{d^n}{dx^n} \tanh(x_0) \tag{B.5}$$

with x_0 a value in tanh(x)'s domain.

We limit the number of the coefficients in a polynomial to 5. This allows us use the roots of $f^{(6)}(x)=0$ to determine the begin and end points of a polynomial $P_u(x_k)$ with the degree 4. The fifth and sixth derivatives of the function $f(x)=\tanh(wx+v)$ with respect to x are

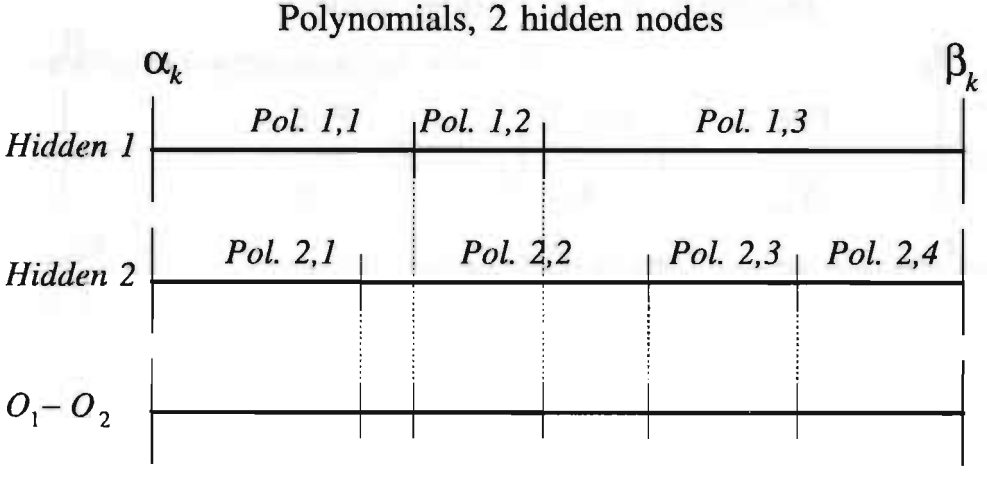


Figure B.2. The coefficients of the polynomials of the two hidden nodes are added resulting in one polynomial that pertains to each intersecting interval.

$$f^{(5)}(x) = 8\,w^5\,\frac{2\cosh(wx+v)^4 - 15\cosh(wx+v)^2 + 15}{\cosh(wx+v)^6} \tag{B.6}$$

$$f^{(6)}(x) = -16\,w^6\sinh(wx+v)\,\frac{2\cosh(wx+v)^4 - 30\cosh(wx+v)^2 + 45}{\cosh(wx+v)^7} \tag{B.7}$$

The roots $f^{(6)}(x)=0$ are

$$-\frac{v}{w} \tag{B.8}$$

$$\pm\frac{\ln\left(\frac{1}{2}\sqrt{3}\sqrt{5+\sqrt{15}}\sqrt{2}+\sqrt{\frac{13}{2}+\frac{3}{2}\sqrt{15}}\right) - v}{w} \tag{B.9}$$

and

$$\pm\frac{\ln\left(\frac{1}{2}\sqrt{3}\sqrt{5-\sqrt{15}}\sqrt{2}+\sqrt{\frac{13}{2}-\frac{3}{2}\sqrt{15}}\right) - v}{w} \tag{B.10}$$

For each hidden node, the origin x_{0k} of the *first* polynomial is computed from (B.3) where the begin point $x_{beg1}=\alpha_k$, the smallest value feature k can possibly take. Then x_{end1} is computed from Eq. (B.4). The point x_{beg2} of the second polynomial is set equal to x_{end1}. This procedure is continued until x_{end} of a polynomial exceeds the limit β_k.

For an MLP with a number of hidden nodes, the polynomials specified in Eq. (A.11) have to be added taking into account the approximation interval of each polynomial. So, for example, for an MLP with 2 hidden nodes, the polynomial $P_{1,1}$ approximates hidden node 1 in the interval $[x_{beg1,1},x_{end1,1}]$ and $P_{2,1}$ hidden node 2 in the interval $[x_{beg2,1},x_{end2,1}]$, see figure B.2. Now, we construct a polynomial that approximates o_j-o_l by adding the coefficients of the two polynomials $P_{1,1}$ and $P_{2,1}$ of which the approximation intervals $[x_{beg1,1},x_{end1,1}]$ and $[x_{beg2,1},x_{end2,1}]$ overlap.

References

[1] S. Ahmad, V. Tresp. "Some solutions to the missing feature problem in vision", In: S.J. Hanson, J.D. Cowlan, C.L. Giles (Ed.), *Neural Information Processing Systems*, Vol. 5, Morgan Kaufmann Publishers, San Mateo, pp. 393-400, 1993.

[2] R. Batitti. "Using mutual information for selecting features in supervised neural net learning", *IEEE Transactions on neural networks*, Vol. 5, No. 4, pp. 537-550, 1994.

[3] C.M. Bishop. *Mixture density networks*, Technical report NCRG/4288, Department of computer science, Aston University, UK, 1994.

[4] T.M. Cover. "The best two independent measurements are not the two best", *IEEE Transactions on man, systems, and cybernetics*, Jan., pp. 116-117, 1974.

[5] W.W. Cooley, P.R. Lohnes. *Multivariate data analysis*, John Wiley & Sons, New York, 1971.

[6] J. Cunningham and S. Haykin. "Neural network detection of small moving radar targets in an ocean environment", In: S.Y. Kung, F. Fallside, J.A. Sorenson and C.A. Kaufmann (Ed.), *Proceedings of the 1992 IEEE workshop on neural networks for signal processing*, IEEE, Piscataway, NJ, pp. 306-315, 1992.

[7] M. Egmont-Petersen, J.L. Talmon, J. Brender, P. NcNair. "On the quality of neural net classifiers", *Artificial Intelligence in Medicine*, Vol. 6, No. 5, pp. 359-381, 1994.

[8] M. Egmont-Petersen, J.L. Talmon, A. Hasman. "Assessing the discriminative power of attributes for multi-layer perceptrons", *Submitted*, 1996.

[9] I. Foroutan, J. Sklansky. "Feature selection for automatic classification of non-gaussian data", *IEEE Transactions on systems, man, and cybernetics*, Vol. 17, No. 2, 1987.

[10] L.K. Hansen, C. Liisberg and P. Salamon. "Ensemble methods for handwritten digit recognition", In: S.Y. Kung, F. Fallside, J.A. Sorenson, C.A. Kaufmann (Eds.), *Proceedings of the 1992 IEEE workshop on neural networks for signal processing*, IEEE, Piscataway, NJ, pp. 333-342, 1992.

[11] R.F. Harrison, S.J. Marshall, R.L. Kennedy. "A connectionist aid to the early diagnosis of myocardial infarction", In: M. Stefanelli, A. Hasman, M. Fieschi and J. Talmon (Eds.), *AIME-91*, Lecture Notes in Medical Informatics 44, Springer Verlag, Berlin, pp. 119-128. 1991.

[12] A. Hart and J. Wyatt. "Connectionist models in medicine: an investigation of their potential", In: J. Hunter, J. Cookson, J. Wyatt (Eds.), *AIME-89*, Lecture Notes in Medical Informatics 38, Springer Verlag, Berlin, pp. 115-124, 1989.

[13] H.J. Holz, M.H. Loew. "Relative feature importance: A classifier-independent approach to feature selection", In: E.S. Gelsema, L.N. Kanal, *Proceedings of Pattern recognition in practice IV: Multiple paradigms, comparative studies and hybrid systems*, Elsevier, pp. 473-487, 1994.

[14] G. Hripcsak. "Using connectionistic modules for decision support", *Methods of Information in Medicine*, Vol. 29, pp. 167-181, 1990.

[15] V. Karthaus, H. Thygesen, M. Egmont-Petersen, J. Talmon, J. Brender, P. McNair. "User-requirements driven learning", *Computer Methods and Programs in Biomedicine*, Vol. 48, No. 1-2, pp. 39-44, 1995.

[16] J. Kittler. "Computational problems of feature selection pertaining to large data sets", In: E.S. Gelsema and L.N. Kanal (Eds.), *Proceedings of Pattern Recognition In Practice*, pp. 405-414, 1980.

[17] J. Kittler. "Feature Selection and Extraction", In: T.Y. Young, K.-S. Fu (Eds.), *Handbook of Pattern Recognition and Image Processing*, Academic Press, Orlando, 1986.

[18] M. Kudo, M. Shimbo. "Feature selection based on the structural indicies of categories", *Pattern recognition*, Vol. 26, No. 6, pp. 891-901, 1993.

[19] P. McNair, J. Brender, S. Ladefoged. "Impact on resource consumption from application of a sequential selection strategy", In: R. O'Moore, S. Bengtsson, J.R. Bryant, J.S. Bryden (Eds.), *Proceedings from Medical Informatics in Europe 1990*, Lecture Notes in Medical Informatics 40, Springer Verlag, Berlin, pp. 381-387, 1990.

[20] C. Moallemi. "Classifying cells for cancer diagnosis using neural networks", *IEEE Expert*, No. 12, pp. 8-12, 1991.

[21] D.C. Montgomery, E.A. Peck. *Introduction to regression analysis*, 2'nd ed., John Wiley & Sons, New York, 1992.

[22] P.M. Narenda, K. Fukunaga. "A branch and bound algorithm for feature subset selection", *IEEE Transactions on computers*, Vol. C-26, No. 9, 1977.

[23] T. Nobis. *Berücksichtigung lokaler und globaler Textureigenschaften durch Erweiterung des Konzepts der Grauwertübergangsmatrizen auf einen Multi skalen ansatz*, Diplomarbeit (Master Thesis), Institut für Medizinische Informatik und Biometrie, Medizinische Fakultät, RWTH-Aachen, Aachen, 1994.

[24] E. Parzen. *Modern probability theory and its applications*, John Wiley & Sons, New York, 1960.

[25] R. Poli, S. Cagnoni, R. Livi, G. Coppini, G. Valli. "A neural network expert system for diagnosing and treating hypertension", *IEEE Computer*, No. 3, pp. 64-71, 1991.

[26] W.H. Press, B.P. Flannery, S.A. Teukolsky, V.T. Vetterling. *Numerical recipes in C*, Cambridge university press, New York, 1988.

[27] J.R. Quinlan. "Learning from noisy data", In: *Proceedings of the third International Machine Learning Workshop*, pp. 58-64, 1983.

[28] J.R. Quinlan. "Comparing connectionist and symbolic learning methods", In: S. Hanson, G. Drastal, R. Rivest (Eds.), *Computational learning theory and natural learning systems: constraints and prospects*, MIT Press, Cambridge MA, 1993.

[29] A. Ralston. *A first course in numerical analysis*, McGraw-Hill, Tokyo, 1965.

[30] M.D. Richard, R.P. Lippmann. "Neural network classifiers estimate bayesian a posteriori probabilities", *Neural computation*, Vol. 3, pp. 461-483, 1991.

[31] D.E. Rumelhart, G.E. Hinton, R.J. Williams. "Learning internal representations by error propagation", In: *Parallel distributed processing. Exploration into the microstructure of cognition*, D.E. Rumelhart, J.L. McClelland and the PDP research group, MIT Press, Cambridge, Vol. 1, Chap. 8, p. 318, 1986.

[32] T. Schiøler, W. Grimson, P. Sharpe, M. Egmont-Petersen, G. Momsen, R. O'Moore and P. McNair. "Automatic decision support based on voting by independent decision support systems", In: *Proceedings Computing in Clinical Laboratories '92*, p. 58, 1992.

[33] C. N. Schizas, C.S. Pattchis, I.S. Schofield, and P.R. Fawcett. "Artificial Neural Nets in Computer-Aided Macro Motor Unit Potential Classification", *Trans. IEEE Engineering in Medicine and Biology*, pp. 31-38, 1990.

[34] W. Siedlecki, J. Sklansky. "On automatic feature selection", *International journal of pattern recognition and artificial intelligence*, Vol. 2, No. 2, pp. 197-220, 1988.

[35] W. Siedlecki, J. Sklansky. "A note on genetic algorithms for large-scale feature selection", *Pattern Recognition Letters*, Vol. 10, No. 5, pp. 335-347, 1989.

[36] S. Siegel, N.J. Castellan. *Nonparametric statistics for the behavioral sciences*, 2'nd ed, McGraw Hill, Singapore, 1988.

[37] K. Sydsæter. *Matematisk analyse. Bind 1*, 5'th ed, Universitetsforlaget, Oslo, 1993.

[38] J.L. Talmon. "A multiclass nonparametric partitioning algorithm". *Pattern Recognition Letters*, Vol. 4, pp. 31-38, 1986.

[39] J.L. Talmon, P. Braspenning, J. Brender, P. McNair, "Machine learning in data rich domains: some experiences from the KAVAS project, In: M. Stefanelli, A. Hasman, M. Fieschi and J. Talmon (Eds.), *AIME-91*, Lecture Notes in Medical Informatics 44, Springer Verlag, Berlin, pp. 283-293. 1991.

[40] F. Vogelsang. *Segmentierung radiologisch dokumentierter fokaler Knochenläsionen auf Basis kontextbezogener Vektoren mit neuronalen Netzwerken*, Diplomarbeit (Master Thesis), Institut für Medizinische Informatik und Biometrie, Medizinische Fakultät, RWTH-Aachen, Aachen, 1993.

[41] R.C. Williamson, U. Helmke. "Existence and uniqueness results for neural network approximations", *IEEE Transactions on neural networks*, Vol. 6, No. 1, pp. 2-13, 1995.

Estimation of missing values with a recurrent MLP The REM-algorithm

5

Abstract

In this paper, a method is developed for the estimation of missing data and of the covariance matrix based on the observed data. The method uses an auto associator network, which is first trained to predict the values in complete cases. For incomplete cases, the auto associator is provided with the observed values and reasonable initial estimates of the missing values in the incomplete case. The auto associator is then used in recurrent mode by computing new estimates of the missing values and feeding these estimates back to the input nodes. The initial estimates of the missing values are replaced by new ones and the missing values are reestimated. This is repeated until a stable state is reached. It is proven under which conditions the recurrent auto associator converges. The recurrent auto associator is embedded in an EM-like algorithm that iteratively estimates the missing values in incomplete cases as well as the covariance matrix of the completed dataset. We call our approach the Recurrent Expectation Maximization algorithm (REM). The recurrent auto associator is compared with multivariate regression and the REM-algorithm is compared with the EM-algorithm in a number of experiments. The major advantages of REM are that it is requires less computation than the EM-algorithm and that cases in which too much information is missing can explicitly be identified.

1 Introduction

Incomplete cases form a frequently occurring problem in empirical research as it impedes statistical analysis of the data. In some situations, one wants to draw conclusions about an individual case, in other situations the goal is to make statistical inferences based on the whole sample. In medicine, for example, physicians often have to establish a diagnosis based on incomplete information [58,59]. Some decision aids cannot cope with incomplete information which make them unapplicable in situations where missing data frequently occur.

Incomplete observations is also a problem in large studies where retrospective collection of the missing data is usually unfeasible and one must somehow cope with the incomplete cases. Among the approaches that have been suggested for handling missing data, the simplest is to abandon the incomplete cases. Many statistical packages have built in this option which is acceptable only when the fraction of incomplete cases is small. The problem of missing data can also be approached by imputation where the empty slots are filled with pseudo values. One possibility is to impute the mean of each variable as computed from all observed values. However, in a situation where for example all values of a variable below a certain threshold are missing, the mean determined from the observed data will obviously be a biased estimate. Whether an approach can provide unbiased estimates of the missing data depends on the 'mechanism' that leads to missing data [29]. It turns out that when the probability that a variable is missing depends on its value, most approaches will impute biased estimates of the missing data.

More advanced approaches than leaving-out incomplete cases and mean imputation are the *Internal Consistency (IC) approach* [8], the *Iterative Lower Rank (ILR) algorithm* [37], the *EM-algorithm* [9,11,28,29,32] and *Multiple Imputation* [48,49]. The IC approach is developed for imputation of categorical data and will not be considered further here as we restrict this paper to imputation of variables with a continuous range. The ILR-algorithm uses correlations both between variables (columns) and cases (rows) to estimate the missing data. Although the ILR-algorithm performed well when used for extrapolation of incomplete marker tracks [37], it is unknown how well the algorithm can perform in other types of completion tasks.

Another method for estimating missing data is the EM-algorithm [4,11]. The EM-algorithm is based on the sample covariance matrix $\hat{\Sigma}$, which is iteratively estimated from the observed data. In each incomplete case, the missing values are predicted from the observed ones with multivariate linear regression (the **E**stimation step). When the missing values of all incomplete cases have been estimated, the covariance matrix $\hat{\Sigma}$ and regression parameters are reestimated from the completed database (the **M**aximization step) and used in the following E-step to compute new estimates of all missing data. The EM-algorithm iterates until the estimate of Σ has stabilized.

One drawback of the EM-algorithm is its computational complexity. If a database contains many variables and numerous patterns of missing and observed variables are present, many different regression parameter sets have to be estimated. Simulation studies have been conducted to analyze the applicability of the EM-algorithm

[49]. They show that when a database that has been completed with the EM-algorithm, is used for hypothesis testing, the p-values are typically too small and the confidence intervals too narrow when compared to the p-values and confidence intervals that are obtained from the complete database. Rubin therefore introduced the multiple imputation approach. The objective is to model the underlying complete data distribution while preserving as much variance as possible in the estimated data. The idea behind multiple imputation is to generate a number of different datasets, each with different estimates of the missing values [48]. The generated datasets are then combined into a large dataset that can be analyzed further.

To our knowledge, there have not been conducted studies to investigate the impact imputation of missing values has on the performance of a classifier. Multiple imputation is not so suited in combination with a classifier as more classifications might be obtained. We propose a computationally tractable method that can estimate the missing data in incomplete cases for the various combinations of missing and observed variables. A subject that has received considerable attention in the connectionist field is *pattern completion* [19]. The so-called *content addressable memories* such as the bidirectional Associative Memory [25] and the Hopfield network [20] are examples of neural networks that can restore deteriorated patterns. A content addressable memory is first trained with a number of 'reference patterns'. After training, when the network is provided with a deteriorated pattern, for example contaminated by noise or only partly available, it recalls the reference pattern that resembles the deteriorated pattern most. The bidirectional associative memory and the Hopfield Network both operate on a limited number of patterns.

One type of network capable of estimating values of continuous variables is the auto associator as introduced by McClelland and Rumelhart [31,50]. The auto associator is a multi-layer perceptron (MLP) neural network with one hidden layer. The network has g input and g output nodes and projects the input vector from the g-dimensional input space onto an h-dimensional space ($h<g$) spanned by the h hidden nodes and back to the g-dimensional output space. An auto associator is normally trained with a gradient descent method, e.g. back-propagation, to predict its input vector such that the output nodes contain unbiased estimates of the input values and the residual variance on the training set is minimized. Baldi and Hornik have shown that a linear auto associator trained this way results in a linear prediction model and that its parameters can be derived from the eigenvectors of the sample covariance matrix $\hat{\Sigma}$ [3]. This technique is known as the Karhunen Loève Transform in signal theory. Such auto associator networks have been used for example for pattern compression [30,51], pattern synthesis [18] and cluster analysis [55].

In this article, we will use an auto associator that is trained to predict the values in complete cases for estimation of missing values. The auto associator is provided with the observed values and reasonable initial estimates of the missing values in an incomplete case. The output vector is computed and the missing values in the input vector are replaced by the corresponding (predicted) output values. This recurrency

enables the auto associator to iterate until the output on the nodes that correspond to the missing values stabilizes.

Such an approach has already been introduced by Gleason and Staelen who proposed to estimate missing data from the observed ones using the principal components of $\hat{\Sigma}$ [14]. Gleason and Staelen briefly considered convergence of a recurrent auto associator but we derive the exact conditions that ensure convergence. A formula is derived for the variance of the predictions provided by the auto associator. The auto associator is embedded in an EM-like algorithm, which we call the Recurrent Expectation Maximization (REM) algorithm, that can iteratively estimate missing values. It will be shown that the REM-approach can overcome some of the shortcomings of the EM-algorithm: it offers the possibility to identify incomplete cases in which too much information is missing. Secondly, the REM-algorithm is computationally less complex than the EM-algorithm.

In a set of experiments, the recurrent auto associator is compared with multivariate regression and the REM-algorithm is compared with the EM-algorithm.

2 The auto associator network

The auto associator is a symmetric linear neural network with one hidden layer. The network has two weight matrices, one that connects the input with the hidden layer, another that connects the hidden with the output layer (see figure 1).

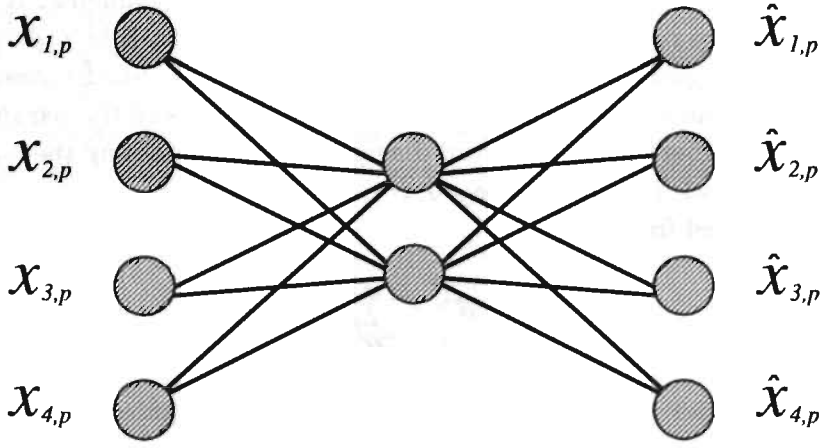


Figure 1. Auto associator network.

2.1 The auto associator – estimation with complete cases

Let us denote case p by $\boldsymbol{x}_p$, a vector with g variables[1]. A data matrix X consists of n vectors, $X=(\boldsymbol{x}_1,..\boldsymbol{x}_n)$, $\dim(X)=g\times n$.

An auto associator performs the mapping

$$\hat{\boldsymbol{x}}_p = C A \boldsymbol{x}_p \tag{1}$$

with the assumption of a zero mean $\boldsymbol{\mu}=\mathbf{0}$ of $\boldsymbol{x}_p$. The matrix A projects the input vector on the subspace spanned by the h hidden nodes. The matrix C transforms the (compressed) vector back to the g-dimensional space [19]. This mapping can be written as

$$\hat{\boldsymbol{x}}_p = W^{(h)} \boldsymbol{x}_p \tag{2}$$

with $W^{(h)}=CA$, the product of the two weight matrices C and A, and h their rank.

The weights of an auto associator can be trained with a gradient descent algorithm to minimize the residual variance summed over all cases $\varepsilon^{(h)^2}$

$$\varepsilon^{(h)^2} = \sum_{p=1}^{n} \| \boldsymbol{x}_p - C A \boldsymbol{x}_p \|_2^2 \tag{3}$$

where $\|\cdot\|_2$ denotes the Euclidian vector norm.

Baldi and Hornik have shown that the summed residual variance $\varepsilon^{(h)^2}$ of a linear auto associator has one uniquely defined minimum for a specific number of hidden nodes h [3]. Training the auto associator by minimizing the prediction variance results in a weight matrix which can also be obtained from the last h eigenvectors of $\hat{\Sigma}$, and a possibly time consuming training process is made superfluous.

Formally, let Λ be the diagonal matrix in which element j,j is the jth eigenvalue of the sample covariance matrix $\hat{\Sigma}$ and let $B=(\boldsymbol{b}_1,...,\boldsymbol{b}_g)$ be the g orthonormal eigenvectors such that $\hat{\Sigma}=B\Lambda B^T$, $B^TB=BB^T=I$. As $\hat{\Sigma}$ is real and symmetric, it follows that its eigenvalues are real and positive. Let the eigenvectors in B be ordered according to the size of the corresponding eigenvalues $\lambda_g(\hat{\Sigma})>\lambda_{g-1}(\hat{\Sigma})>...>\lambda_1(\hat{\Sigma})$. Assume that no eigenvalues are the same, $\lambda_i(\hat{\Sigma})\neq\lambda_j(\hat{\Sigma})$, $i\neq j$. We can now rewrite the parameter matrix as $W^{(h)}=BE^{(h)}B^T$, where $E^{(h)}$ is a partial diagonal matrix containing the eigenvalues of $W^{(h)}$, $e_{j,i}=0$, $j\neq i$; $e_{i,i}=1$, $i=g-(h-1),..g$; $e_{i,i}=0$, $i=1,..,h$.

$W^{(h)}$ is computed from

$$W^{(h)} = \sum_{j=g-h+1}^{g} W^j \tag{4}$$

with

[1]Henceforward, a capital letter X denotes a matrix, a bold letter $\boldsymbol{y}$ a vector. $\boldsymbol{x}_p \in X$ denotes column p in X, $\boldsymbol{x}^{<i>}$ denotes row i in X, and $x_{i,p}$ the ith element in column p in the matrix X. Denote the ith element in vector $\boldsymbol{y}$ by y_i. $E(x)$ denotes the expected value of variable x.

$$W^j = \boldsymbol{b}_j \boldsymbol{b}_j^{\mathrm{T}} = \begin{bmatrix} b_{1,j}b_{1,j} & \cdots & b_{1,j}b_{g,j} \\ \cdots & \cdots & \cdots \\ b_{g,j}b_{1,j} & \cdots & b_{g,j}b_{g,j} \end{bmatrix} \tag{5}$$

Each matrix W^j is the identity mapping for the eigenvector $\boldsymbol{b}_j$, $W^j\boldsymbol{b}_j=\boldsymbol{b}_j$, with $|\boldsymbol{b}_j|_2=1$. We may say that matrix W^j "stores" $\boldsymbol{b}_j$ in $W^{(h)}$. Maximally g orthogonal unit vectors can be stored in a matrix W of rank g. In fact, $W^{(h)}\boldsymbol{x}$ performs an orthogonal projection of $\mathbf{R}^g$ onto a subspace $\mathbf{R}^h$ and back to $\mathbf{R}^g$.

2.2 Estimating missing values

Without loss of generality, we assume that for a vector $\boldsymbol{x}_p$ the last m elements are missing and the number of *observed* elements is k, $g=k+m$. Vector $\boldsymbol{x}_p$ is divided into $\boldsymbol{x}_p^k$ and $\boldsymbol{x}_p^m$, $\boldsymbol{x}_p{}^{\mathrm{T}}=(\boldsymbol{x}_p^{k\,\mathrm{T}},\boldsymbol{x}_p^{m\,\mathrm{T}})$. Let's assume that the weight matrix $W^{(h)}$ has been computed from Eq. (4) and that the eigenvectors are based on $\hat{\Sigma}$ that is estimated from a subset of complete cases. In this section, we consider estimation of missing values in only one vector $\boldsymbol{x}_p$ so the index p is omitted.

Now, given an incomplete case $\boldsymbol{x}$, we will estimate the missing values $\boldsymbol{x}^m$ from the observed $\boldsymbol{x}^k$ using the auto associator. Construct an input vector $\hat{\boldsymbol{x}}(0)$ that contains all observed values $\boldsymbol{x}^k$ and initial estimates for $\boldsymbol{x}^m$. Define a recurrent mapping F: $\hat{\boldsymbol{x}}(t) \rightarrow \hat{\boldsymbol{x}}(t+1)$, where $\hat{\boldsymbol{x}}(t+1)$ is computed from

$$\hat{\boldsymbol{x}}(t+1) = W^{(h)}\hat{\boldsymbol{x}}(t) \tag{6}$$

In each iteration, the computed values in $\hat{\boldsymbol{x}}(t)$ are used to construct the input vector in the following iteration: all observed values $\boldsymbol{x}^k$ remain fixed; the values in $\hat{\boldsymbol{x}}^m$ are replaced by the respective estimated values $\hat{\boldsymbol{x}}(t+1)$: $\hat{\boldsymbol{x}}(t+1)^{\mathrm{T}}=(\boldsymbol{x}^{k\,\mathrm{T}},\hat{\boldsymbol{x}}^m(t+1)^{\mathrm{T}})$. Iteration with the recurrent auto associator continues until the estimates of the missing values stabilize: $\Sigma_{i=k+1,\ldots g}\,(\hat{x}_i(t+1)-\hat{x}_i(t))^2<\alpha$, with α being a small positive number.

The recurrent mapping F can be simplified. Define $W_{mk}^{(h)}$ and $W_m^{(h)}$ to be respectively the lower left and lower right submatrices of $W^{(h)}$, $\dim(W_{mk}^{(h)})=m\times k$ and $\dim(W_m^{(h)})=m\times m$

$$W^{(h)} = \begin{bmatrix} W_k^{(h)} & W_{km}^{(h)} \\ W_{mk}^{(h)} & W_m^{(h)} \end{bmatrix} \tag{7}$$

Estimates of the missing values in $\boldsymbol{x}$, $\hat{\boldsymbol{x}}^m(t+1)$, can now be computed with the iterative formula

$$F(\boldsymbol{x}^k, \hat{\boldsymbol{x}}^m(t)): \quad \hat{\boldsymbol{x}}^m(t+1) = W_{mk}^{(h)} \boldsymbol{x}^k + W_m^{(h)} \hat{\boldsymbol{x}}^m(t) \tag{8}$$

The vector $W_{mk}^{(h)} \boldsymbol{x}^k$ is constant because both $W_{mk}^{(h)}$ and $\boldsymbol{x}^k$ remain fixed.

2.2.1 Convergence of the recurrent auto associator for a single case

Convergence of a recurrent neural network to an attracting point depends in principle on the initial values of the input vector and on the weight matrix [5]. We will prove that under certain conditions the recurrent auto associator $F(\boldsymbol{x}^k, \hat{\boldsymbol{x}}^m(t))$ has *one* attracting point $\hat{\boldsymbol{x}}^{m\bullet}$ for which $\hat{\boldsymbol{x}}^{m\bullet} = W_{mk}^{(h)} \boldsymbol{x}^k + W_m^{(h)} \hat{\boldsymbol{x}}^{m\bullet}$.

Definition 2.2.1. The spectral radius $\rho(A)$ of any symmetric matrix A is equal to $\lambda_{max}(A)$, where λ_{max} denotes the largest eigenvalue of A.

The spectral radius of the projection matrix $W^{(h)}$, $\rho(W^{(h)})=1$.

Lemma 2.2.2. For a real symmetric matrix A, $dim(A)=g\times g$, and any symmetric submatrix A_m, $dim(A_m)=m\times m$, $m<g$, holds for each $1\le j\le m$: $\lambda_{j+g-m}(A)\ge\lambda_j(A_m)\ge\lambda_j(A)$, where the eigenvalues of A and A_m are ordered such that $\lambda_{j+1}(A)\ge\lambda_j(A)$, $j=1,..,g-1$, and $\lambda_{j+1}(A_m)\ge\lambda_j(A_m)$, $j=1,..,m-1$, respectively.

Proof
See Theorem 4.3.15 in [21] ■

From Lemma 2.2.2 follows that the spectral radius of any symmetric submatrix $W_m^{(h)}$ of $W^{(h)}$ fulfils the following inequality: $\rho(W_m^{(h)})\le 1$.

Lemma 2.2.3. A matrix (I-A) can be inverted when $\rho(A)<1$.

Proof
See the von Neumann Lemma in [39] ■

Lemma 2.2.4. An iterative mapping of the form $\boldsymbol{x}(t+1)=\boldsymbol{b}+A\boldsymbol{x}(t)$, where $\boldsymbol{b}$ is a constant vector and A a symmetric matrix, has one attracting point to which it converges when $\rho(A)<1$.

Proof
See Proposition 10.1.5 in [39] ■

Corollary 2.2.5. Given a vector $\hat{x}^m(t)$ *and the mapping F:* $\hat{x}^m(t+1) = W_{mk}^{(h)} x^k + W_m^{(h)} \hat{x}^m(t)$, *where* $W_{mk}^{(h)}$ *and* $W_m^{(h)}$ *are submatrices of* $W^{(h)}$ *as defined in Equation (7),* $F(x^k, \hat{x}^m(t))$ *will converge to an attracting point* $\hat{x}^{m\bullet}$ *when* $\rho(W_m^{(h)})<1$.

Notice that Lemma 2.2.2 ensures only that $\rho(W_m^{(h)})\leq 1$ and not that $\rho(W_m^{(h)})<1$.

Theorem 2.2.6. The recurrent auto associator F does **not** *converge when the number of principal components h in* $W^{(h)}$ *is chosen such that* $h>g-m$, *where g is the dimension of vector* x *and m the number of missing values in* x.

Proof

According to Lemma 2.2.2 $\lambda_{max}(W_m^{(h)})$ is bounded by: $\lambda_g(W^{(h)})\geq\lambda_m(W_m^{(h)})\geq\lambda_m(W^{(h)})$. If $h>g-m$ then $\rho(W_m^{(h)})=1$ because $\lambda_m(W^{(h)})=1$ and $\lambda_m(W_m^{(h)})\geq\lambda_m(W^{(h)})=1$. Then, following from Corollary 2.2.5, there will be no convergence.

If $h\leq g-m$ then $\lambda_m(W^{(h)})=0$ and $\lambda_m(W_m^{(h)}) \in (0,1)$ ■

Whether for a specific choice of h the mapping F converges, depends on $\rho(W_m^{(h)})$. To check for convergence, one can compute the eigenvalues of $W_m^{(h)}$ for different choices of $h=1,..,g-m$.

Theorem 2.2.7. Given a vector $x(t)$ *and a recurrent mapping* $F(x^k, \hat{x}^m(t))$: $\hat{x}^m(t+1)=(W_{mk}^{(h)} x^k)+W_m^{(h)} \hat{x}^m(t)$, *by which the m unknown values* $\hat{x}^m$ *are updated, F has the attracting point* $\hat{x}^{m\bullet}$ *that is the solution to the linear equation* $\hat{x}^{m\bullet} = (I_m - W_m^{(h)})^{-1} W_{mk}^{(h)} x^k$ *s.t.* $\rho(W_m^{(h)})<1$.

Proof.

Following from Corollary 2.2.5, the mapping $\hat{x}^m(t+1)=(W_{mk}^{(h)} x^k)+W_m^{(h)} \hat{x}^m(t)$ converges to an attracting point $\hat{x}^{m\bullet}$ when $\rho(W_m^{(h)})<1$. For the attracting point $\hat{x}^{m\bullet}$ holds

$$\hat{x}^{m\bullet} = W_{mk}^{(h)} x^k + W_m^{(h)} \hat{x}^{m\bullet} \tag{9}$$

Let I_m be the $m\times m$ identity matrix in

$$\hat{x}^{m\bullet} - W_m^{(h)} \hat{x}^{m\bullet} = W_{mk}^{(h)} x^k \Leftrightarrow (I_m - W_m^{(h)})\hat{x}^{m\bullet} = W_{mk}^{(h)} x^k \tag{10}$$

As $(I_m - W_m^{(h)})$ is a square matrix,

$$\hat{x}^{m\bullet} = (I_m - W_m^{(h)})^{-1} W_{mk}^{(h)} x^k \tag{11}$$

From Lemma 2.2.3 follows that the matrix $(I_m - W_m^{(h)})$ can be inverted when $\rho(W_m^{(h)})<1$. So convergence is guaranteed when the inverse $(I_m - W_m^{(h)})^{-1}$ exists. ■

These proofs give occasion to some remarks. The submatrix $W_{mk}^{(h)}$ captures the correlations between the missing and observed data. It is closely related with the

submatrix Σ_{mk} that is used to compute the regression coefficients between the missing and observed variables. The matrix $W_m^{(h)}$, on the other hand, captures the part of the variance of the missing variables that cannot be predicted by the observed variables. From the matrix $W_m^{(h)}$ follows whether the recurrent auto associator converges. In general, when the missing variables depend only little on the observed variables $\lambda_{max}(W_m^{(h)})$ is large.

In a situation where a missing variable is independent, there will be an eigenvalue λ_k for which $\boldsymbol{b}_k\boldsymbol{b}_k^T$ is a matrix with all values zero except for element k,k which is 1. If the variance of the missing variable is large enough compared with the variance of the other variables, the eigenvector $\boldsymbol{b}_k$ is used by the recurrent auto associator, in $W^{(h)}$. The following equality holds

$$I_m = W_m^{j>h} + W_m^{j\leq h} \tag{12}$$

with

$$W_m^{j>h} = \sum_{j=g-(h-1)}^{g} W_m^j \tag{13}$$

and

$$W_m^{j\leq h} = \sum_{j=1}^{g-h} W_m^j \tag{14}$$

W_m^j is the $m\times m$ submatrix of (5) and $W_m^{j>h}$ denotes the sum from $j=g-(h-1),..,g$ of the submatrices of W_j that correspond to W_m. Under these circumstances, the maximal eigenvalue of $W_m^{(h)}$ is 1 and the determinant $|I_m-W_m^{(h)}|$ is 0. Consequently, the recurrent auto associator will not converge.

Another situation is when an independent missing variable has a small variance relatively to each of the observed variables. Also in this situation, $\boldsymbol{b}_k\boldsymbol{b}_k^T$ is a matrix with all values zero except for element k,k which is 1. As the corresponding eigenvalue is small, this eigenvector is not used to compute $W^{(h)}$. Now, following from the equality (12), the maximal eigenvalue of $W_m^{(h)}$ is 0 and the determinant $|I_m-W_m^{(h)}|$ is 1. So the recurrent auto associator converges. However, due to the fact that the last principal component captures all variation of the missing variable, $W_{mk}^{(h)}$ will contain zeros and the output of the auto associator will be zero (the mean of the variable). An important conclusion is that variables can best be predicted by the recurrent auto associator when they have the same variance. Thereby, the ranking of the g eigenvalues reflects the redundancy among the g variables and the eigenvectors that capture most of the covariance are used in the computation of $W^{(h)}$.

Another remark regards the restriction on h which depends on the number of missing variables, $h\leq g-m$. For a specific data matrix, the more variables are missing, the larger the maximal eigenvalue of $W_m^{(h)}$ will, in general, be. This is so because the variance explained by the last m components increases. Therefore, when more variables are missing, the predictions of the recurrent auto associator will be poorer.

How much poorer, depends on the relation between the g–h first and the h last eigenvalues of Λ, in other words on the redundancy in the data.

2.3 Residual variance of estimation

The missing values in each incomplete case are predicted using the auto associator in recurrent mode. Its weights $W^{(h)}$ are derived from the eigenvectors of the sample covariance matrix which has been estimated with a (sub)set of complete cases. For an incomplete case with a particular combination of missing and observed variables, the matrix $P^{(h)}=(I_m-W_m^{(h)})^{-1}W_{mk}^{(h)}$ is computed. Theorem 2.2.6 indicates that the maximal number of eigenvectors h that will result in convergence depends on the number of missing values m in a case. The residual variance per case is estimated from the subset of *complete* cases, denoted COM, by

$$n^{-1}\,\breve{\varepsilon}^{(h)^2} = n^{-1}\sum_{p\in \mathrm{COM}} \| x_p^m - P^{(h)} x_p^k \|_2^2 \qquad (15)$$

with m and k denoting the combination of missing and known variables and n the number of complete cases. Thereby, h is chosen such that the residual variance estimated on the *complete* cases is minimal.

For variable i the residual variance is computed from

$$n^{-1}\,\breve{\varepsilon}_i^{(h)^2} = n^{-1}\sum_{p\in \mathrm{COM}} (x_{i,p}^m - \boldsymbol{p}^{<i>} x_p^k)^2 \qquad (16)$$

with $\breve{}$ denoting that the auto associator is used in *recurrent* mode and n the number of complete cases. From [12] it follows that the residual $\breve{\varepsilon}^{(h)^2}$ is computed from

$$\breve{\varepsilon}^{(h)^2} = \varepsilon_r^2 + \sum_{i=h+1}^{\min(m,k)} \tilde{\lambda}_i^2 \qquad (17)$$

with ε_r^2 the (minimal) residual variance obtained when the missing values X_m are predicted with linear regression. The values $\tilde{\lambda}_i^2$, i=1,..,min(m,k) are called the generalized eigenvalues of the SVD of the two submatrices $\hat{\Sigma}_k$ and $\hat{\Sigma}_m$ of $\hat{\Sigma}$ [12]. The values $\tilde{\lambda}_i^2$ express the excess error due to the prediction with the recurrent auto associator.

It is possible that for one or more of the variables to be predicted the residual variance exceeds the estimated variance of the variable, $\exists i: n^{-1}\,\breve{\varepsilon}_i^{(h^*)^2} > \hat{\sigma}_{i,i}^2$, when the variables correlate poorly with the observed variables. It is necessary to detect such situations to avoid predicting a variable with a variance that is larger than its variance. When the missing values cannot be predicted 'accurately' by the auto associator, we suggest to remove the case from the dataset.

We suggest to choose the optimal number of components h^* such that residual variance is minimal: $\min_{h=1,..g-m}(\breve{\varepsilon}^{(h)^2})$, s.t. $n^{-1}\,\breve{\varepsilon}_i^{(h)^2} < \sigma_{i,i}^2$, i=1,..,m, $\lambda_{max}(W_m^{(h)})$<1.

3 The REM-algorithm

In the previous section, we developed the recurrent auto associator for prediction of missing values. To obtain the best estimates of the missing values in a scattered datamatrix X, we suggest to iteratively estimate the covariance matrix Σ and the missing values in X. We propose therefore to embed the auto associator in an iterative EM-like algorithm to estimate the missing values in the incomplete dataset and subsequently to reestimate the parameters $(\boldsymbol{\mu},\Sigma)$, an idea originally proposed by Gleason and Staelen [14]. We will define the Recurrent Expectation Maximization (REM) algorithm for this purpose.

3.1 Definition

Let's assume that the missing values X_m are initialized with some reasonable values and that the initial parameter estimates $\hat{\boldsymbol{\mu}}^{(0)}$ and $\hat{\Sigma}^{(0)}$ are computed from the subset of complete cases. Let c denote the REM-cycle number.

The E-step:

The eigenvalues and eigenvectors B of the sample covariance matrix $\hat{\Sigma}^{(c)}$ are computed. In the data matrix, for each subset of cases with the same combination of missing and observed variables, the optimal number of hidden nodes h in the auto associator and the parameter matrix of the recurrent auto associator $P^{(h)}$ are determined. The missing values are estimated for those incomplete cases $\boldsymbol{x}_p$ which have the same combination of missing and observed values.

The M-step:

The covariance matrix $\hat{\Sigma}^{(c+1)}$ is reestimated from the updated data matrix X. If the difference between $\hat{\Sigma}^{(c)}$ and $\hat{\Sigma}^{(c+1)}$ as measured with Z_α (to be defined below) exceeds some threshold, go to the E-step else stop.

One should iterate the E- and M-steps until the parameter estimates stabilize. Note that whereas the covariance matrix $\hat{\Sigma}^{(0)}$ is initially estimated from the subset of complete cases, the successive estimates $\hat{\Sigma}^{(c)}$, c=1,..., are estimated from all cases.

We suggest to stop iteration when the *correlation matrices* $\hat{S}^{(c+1)}$ and $\hat{S}^{(c)}$ computed from $\hat{\Sigma}^{(c)}$ and $\hat{\Sigma}^{(c+1)}$ differ less than a threshold α. This way, the influence of all variables is equally considered in the stop criterion. Like Gleason and Staelen we use Z_α, the root-mean-square difference of the $g(g-1)$ *off-diagonal* correlations in the two matrices [14]

$$Z_\alpha = \sqrt{\frac{\|\hat{S}^{(c+1)} - \hat{S}^{(c)}\|_2^2}{g(g-1)}} \qquad (18)$$

to compare the two correlation matrices. The REM-algorithm is formally specified in appendix A.

3.2 Prediction of missing values with REM

The EM-algorithm provides a maximum likelihood estimate of Σ based on the observed data. The REM-algorithm, based on the auto associator, predicts missing values with a larger residual variance than multivariate regression. An advantage of the REM-algorithm is that it requires less computation than the EM-algorithm (see discussion). Another difference between the two algorithms is that in the M-step of the EM-algorithm, a correction term is added to the sample covariance matrix $\hat{\Sigma}$ to correct for the covariance between the regression parameters that were estimated in the previous E-step [4]. For the EM-algorithm, the correction term can be computed from the partial covariance matrix $\hat{\Sigma}_{m \cdot k}$ (the covariance matrix of the conditional distribution of $\boldsymbol{x}^m$ given $\boldsymbol{x}^k$) for each incomplete case. Estimating a similar correction term for the REM-algorithm would entail computation of the prediction variance of the parameters $P^{(h)}$. We derived this variance elsewhere and it turns out that estimation of this variance is computationally complex. So such a correction term would slow the REM-algorithm considerably and was not implemented.

4 Experiments

We designed an experiment to compare the predictions of the recurrent auto associator with the predictions of multivariate regression which is used to estimate missing values in the EM-algorithm. As in this experiment the parameters of the two predictors are estimated from complete datasets it is suited as a benchmark to compare the two prediction methods. In two other experiments, we compared how well the EM- and REM-algorithms could predict the covariance matrix and missing values. In the fourth experiment, REM is used to estimate missing values in cases that are offered to a classifier.

Several data sets were artificially generated. All attributes have a zero mean. The different covariance matrices are shown in table 1.

First experiment

The first experiment is designed to investigate how well the two predictors – the recurrent auto associator and multivariate regression – can estimate values. It should be noted that as the data come from a normal distribution linear regression provides the best possible estimates. In this experiment, all cases were complete.

The covariance matrices in table 1 can be divided into two categories: In the covariance matrices 1, 4 and 6, all four variables have the same variance whereas in the covariance matrices 2, 3 and 5, either one or three of the variables have higher variance than the other variables.

The experiment was performed as follows. All true means were set to $\boldsymbol{\mu}=\mathbf{0}$. For each covariance matrix, 1000 samples, each consisting of 100 cases were drawn from a normal distribution. For each sample, the sample mean was computed and subtracted from the 100 cases. Both multivariate linear regression and the recurrent auto associator were used to predict all $2^4-2=14$ combinations of observed and

One

x_1	x_2	x_3	x_4
1	0.8	0.5	0.55
	1	0.7	0.6
		1	0.7
			1

Two

x_1	x_2	x_3	x_4
3	0.8	0.5	0.55
	1	0.7	0.6
		1	0.7
			1

Three

x_1	x_2	x_3	x_4
1	0.8	0.5	0.55
	3	0.7	0.6
		3	0.7
			3

Four

x_1	x_2	x_3	x_4
1	0.8	0.2	0.1
	1	0.2	0.3
		1	0.9
			1

Five

x_1	x_2	x_3	x_4
3	0.8	0.2	0.1
	1	0.2	0.3
		1	0.9
			1

Six

x_1	x_2	x_3	x_4
1	0.7	0.6	0.5
	1	0.5	0.4
		1	0.3
			1

Table 1. The covariance matrices used.

missing variables for each sample. For each combination, the parameters were estimated from the 100 cases in the sample and the residual variances of the m missing values were computed. Also when the residual variance of the recurrent auto associator exceeded the variance of one or more of the missing variables, $\exists i: n^{-1} \breve{\varepsilon}_i^{(h)^2} > \hat{\sigma}_i^2$, the values were predicted by the recurrent auto associator.

We used the root-mean-square standardized residual Q_α to compare the residual variance obtained

$$Q_\alpha = \sqrt{\sum_{p=1}^{n} \sum_{i \in predicted} \frac{(\hat{x}_{i,p} - x_{i,p})^2}{\sigma_i^2 m}} \tag{19}$$

with $\hat{x}_{i,p}$ the estimated value, σ_i^2 the population variance of variable i, n the number of cases, m the number of variables predicted and *predicted* a set containing the variables that are predicted in the investigated combination. We compared the relative deviation Δ between Q_{RAA} and Q_{MRV}, the root-mean-square standardized residuals of the variables that were predicted with the recurrent auto associator and with multivariate regression, respectively:

Δ	Cov - 1	Cov - 4	Cov - 6	Cov - 2	Cov - 3	Cov - 5
One predicted	13.0%	12.4%	6.6%	15.0%	76.8%	23.9%
Two predicted	8.7%	34.9%	8.6%	19.3%	87.5%	177.6%
Three predicted	11.7%	844.1%	20.5%	25.5%	116.7%	2082.3%

Table 2. Difference in prediction error Δ for values predicted with the recurrent auto associator and multivariate regression.

$$\Delta = \frac{Q_{RAA} - Q_{MVR}}{Q_{MVR}} \qquad (20)$$

Δ measures the relative excess error due to prediction with the recurrent auto associator. The deviations for the six covariance matrices are presented in table 2.

Some combinations of observed and predicted values in samples from the first category of covariance matrices (1, 4 and 6, not shaded), are predicted with up to 20% higher root-mean-square residual by the recurrent auto associator than by regression. However, one can see that the relative error Δ increases dramatically for the fourth covariance matrix when two or three variables are to be predicted. Missing values predicted from samples based on the other three covariance matrices are much poorer.

Table 3 shows for each combination of two observed and two predicted variables the values of Q_{MRV} and Q_{RAA} for data generated using covariance matrix 4. Only the two last combinations of predicted/observed cause problems for the recurrent auto associator. This can be explained from covariance matrix 4, which contains basically two principal components: the variables 1 and 2, and the variables 3 and 4. When either 1 and 2 or 3 and 4 are "missing", the recurrent auto associator results in poor predictions. For the patterns of predicted and observed data 0011 and 1100 ('0' means predicted, '1' observed) the criterion $n^{-1}\,\varepsilon_i^{(h\bullet)^2} < \sigma_{i,i}^2$ is not fulfilled.

	Regression		Recurr. A. A.	
Comb.	Q_{MVR}	Q_{MVR}	Q_{RAA}	Q_{RAA}
1001	0.508	0.385	0.544	0.408
0101	0.531	0.393	0.562	0.409
1010	0.546	0.392	0.580	0.411
0110	0.546	0.383	0.600	0.414
0011	0.881	0.860	1.612	1.603
1100	0.892	0.846	1.744	1.712

Table 3. Root-mean-square standardized residuals for 2 predicted values, covariance matrix 4.

For the second category of covariance matrices in table 2, the results are acceptable only when one value is to be predicted in the matrices 2 and 5. However, as more values are to be predicted, the predictions of become very poor.

Second experiment

In the second experiment, we compare the performance of the EM- and the REM-algorithms. A number of randomly selected observations in the datamatrix were considered missing and had to be predicted by the REM and EM-algorithms. We compare both the residual variance of missing values predicted by the two algorithms and the difference between the population correlation matrix and the estimated correlation matrices. Although it followed from the previous experiment that missing values are predicted poorly by the recurrent auto associator when their variances differ, we included the three covariance matrices 2, 3 and 5 to study whether leaving out cases that can only be predicted poorly will improve the prediction of the correlation matrix.

Based on each of the six covariance matrices in table 1, 100 samples each consisting of 100 cases were drawn from the normal distribution. From each sample, a fraction $\pi \in \{0.1,0.2,0.4\}$ of values in the datamatrix were randomly considered missing which means that they were not used to estimate the sample parameters. Instead, the missing values and the correlation matrix were iteratively estimated from the remaining (observed) values by the EM- and REM-algorithms. Iteration was continued until the difference between the estimated correlation matrices in two successive REM-cycles $Z_\alpha<0.01$ (see Eq. (18)).

In this experiment, we used a modified root-mean-square standardized residual to compare the missing values in a data matrix

$$Q_\beta = \sqrt{\sum_{p=1}^{n} \sum_{i=1}^{g} \frac{(\hat{x}_{i,p} - x_{i,p})^2}{\sigma_i^2 n g \pi}} \qquad (21)$$

with $\hat{x}_{i,p}$ the estimated value, σ_i^2 the variance of variable i, n the number of cases, g the number of variables and π the fraction of entries in the data matrix that are missing. Q_β is a weighted average of the residual standard deviation among the missing values in a sample. After the EM/REM-algorithm had stopped iteration, the estimated correlation matrix was compared with the population correlation matrix (see table 1) using the criterion Z_α. In the current experiment, the recurrent auto associator is used to predict the missing value also when the criterion $\forall i: n^{-1} \breve{\varepsilon}_i^{(h)^2} \leq \hat{\sigma}_i^2$ was *not* fulfilled. The number of principal components h was chosen as to minimize the residual variance as specified in section 2.3.

In this experiment, the root-mean square difference Z_α is used as a stop criterion for the REM-algorithm as well as to compute the deviation between the population correlation matrix and the correlation matrix estimated using REM. Table 4 contains the average of Z_α and Q_β over the 100 samples. The measures Z_α indicate that the EM-algorithm provides equally good or better estimates of the correlation matrix S

Covariance matrix	Fraction of missing data	Z_α		Q_β	
		EM-all	REM-all	EM-all	REM-all
Cov - 1	10%	0.054	0.050	0.621	0.703
	20%	0.056	0.065	0.652	0.725
	40%	0.098	0.099	0.750	0.778
Cov - 2	10%	0.117	0.098	0.659	0.828
	20%	0.090	0.152	0.658	0.832
	40%	0.144	0.222	0.717	0.870
Cov - 3	10%	0.301	0.248	0.627	0.704
	20%	0.397	0.724	0.630	0.909
	40%	0.312	0.320	0.659	1.883
Cov - 4	10%	0.381	0.326	0.487	1.977
	20%	0.564	0.364	0.637	2.532
	40%	0.381	0.461	1.200	3.239
Cov - 5	10%	0.341	0.427	0.602	1.258
	20%	0.392	0.532	0.686	2.399
	40%	0.427	0.719	0.730	5.730
Cov - 6	10%	0.156	0.104	0.764	0.820
	20%	0.127	0.121	0.792	0.858
	40%	0.195	0.174	0.819	0.972

Table 4. Results of the second experiment.

than the REM-algorithm. When the variables were highly correlated and the data contain a high amount of redundancy, both algorithms performed good even when 40% of the data were missing (the covariance matrices 1 and 6). As the redundancy in the data decreases, the discrepancy between $Z_{EM\text{-}all}$ and $Z_{REM\text{-}all}$ increases. When, however, the fraction of missing data is only 10% both methods provide good estimates of the correlation matrices. The discrepancies between the two algorithms increase when the redundancy decreases because multivariate regression is a better predictor than the recurrent auto associator. This is clearly indicated by the values of Q_β.

The values for $Z_{EM\text{-}all}$ and $Z_{REM\text{-}all}$ in table 4 are averages of the 100 samples. An analysis indicated that the coefficients of variation of $Z_{EM\text{-}all}$ and $Z_{REM\text{-}all}$ differed. $Z_{REM\text{-}all}$ had always a smaller coefficient of variation than $Z_{EM\text{-}all}$. The measures $Q_{EM\text{-}all}$ and $Q_{REM\text{-}all}$ indicate that the recurrent auto associator predicts the missing values less well than multivariate regression. Table 4 indicates that the recurrent auto associa-

tor estimates the missing values best in datasets with high redundancy, i.e. when the variables are fairly correlated.

Third experiment

The former experiment indicated that the performance of the REM-algorithm deteriorates when the recurrent auto associator provides poor predictions. In datasets with small redundancy, the predictions of the REM-algorithm had a much higher residual variance than the predictions of the EM-algorithm. We therefore designed an experiment to investigate whether leaving out incomplete cases that are predicted poorly by the recurrent auto associator improves the correlation matrix estimates of REM.

This experiment is identical to the previous one with the exception that the REM-algorithm is initially used to select the subset of cases that could be predicted with the recurrent auto associator with a residual variance smaller than the estimated

Covariance matrix	Fraction of missing data	REM-subset	
		Z_α	Q_β
Cov - 1	10%	0.050	0.695
	20%	0.060	0.708
	40%	0.091	0.761
Cov - 2	10%	0.089	0.842
	20%	0.151	0.851
	40%	0.163	0.858
Cov - 3	10%	0.551	0.855
	20%	0.395	0.860
	40%	0.508	0.912
Cov - 4	10%	0.383	0.817
	20%	0.358	0.788
	40%	0.373	0.755
Cov - 5	10%	0.434	0.954
	20%	0.441	0.895
	40%	0.359	0.903
Cov - 6	10%	0.107	0.806
	20%	0.133	0.836
	40%	0.169	0.867

Table 5. Results of the third experiment.

variance of that variable. For each sample of cases, the covariance matrix is first estimated from the subset of complete cases. Subsequently for each incomplete case with a specific combination of missing and observed variables, the parameters $P^{(h)}$ of the recurrent auto associator are computed from the estimated covariance matrix $\hat{\Sigma}^{(0)}$. As in the previous experiment, h is chosen as to minimize total residual variance $n^{-1}\,\varepsilon^{(h)^2}$. When however, for a specific combination of missing and observed variables for all values of $h \leq m$, one or more variables could not be predicted better than its estimated variance, $n^{-1}\,\varepsilon_i^{(h)^2} > \hat{\sigma}_i^2$, $i \in$ *missing*, all cases with this specific combination were removed from the sample. When all cases that could not be predicted well enough are deleted, the REM-algorithm is used to estimate the missing values of the remaining incomplete cases and the correlation matrix as in the previous experiment.

Table 5 contains results of the third experiment. Leaving out the cases that cannot be predicted well improves the estimate with the REM-algorithm of the correlation matrices ($Z_{\text{REM-all}}$ compared to $Z_{\text{REM-subset}}$, table 4 and 5). In some datasets, the REM-algorithm even results in a better estimate of the correlation matrix than the EM-algorithm did on the whole sample. These results are also displayed in figure 2 which contains also the value of Z_α that results when the correlation matrices are estimated from a complete data set. So leaving out cases that can only be predicted poorly results in better estimates of the correlation matrix.

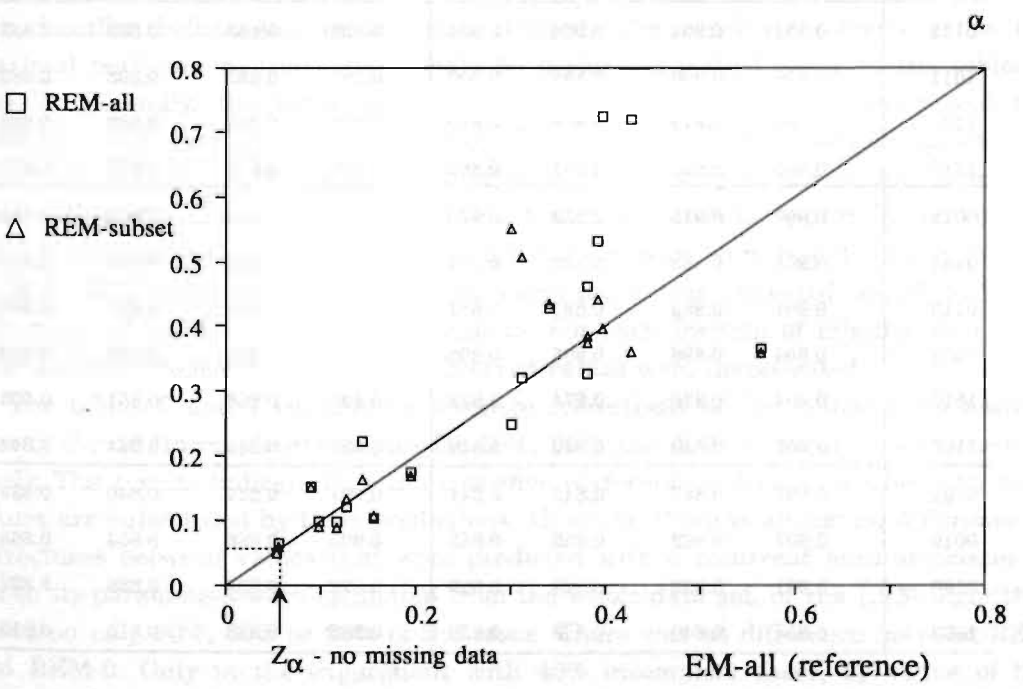


Figure 2. Scatter plot of deviations Z_α between population and estimated correlation matrices resulting from the EM-algorithm based on all cases and the REM-algorithm based on all and a subset of cases.

We conclude that for datasets that contain much redundancy (covariance matrices 1 and 6), the REM-algorithm results in estimates of the correlation matrix with the same deviation from the population matrices as those obtained from the EM-algorithm. For these datasets, the recurrent auto associator estimated the missing values with a slightly higher residual variance than the EM-algorithm.

Fourth experiment

We designed a fourth experiment to investigate the influence of imputing missing values with the REM-algorithm on the performance of a trained neural net classifier. We designed a classification problem with two classes A and B that are characterized by four attributes and the centres $\mu_A=(0,0,0,0,0,0)^T$ and $\mu_B=(2,2,2,2,2,2)^T$, respectively. The two classes have identical covariance matrices $\Sigma_A=\Sigma_B$ which are equal to covariance matrix 6 in table 1.

We sampled 1000 vectors from the normal distribution $N(\boldsymbol{x}|\mu_A,\Sigma)$ and 1000 from $N(\boldsymbol{x}|\mu_B,\Sigma)$. These were divided into a training and a test set each consisting of 1000 vectors. Three MLPs with 2 hidden nodes, all with different initial weight configurations, were trained 1500 cycles. The network with the highest correctness on the test, 0.898, was chosen. The correctness of a minimal error rate classifier is 0.902.

Com.	10%				20%			
	Data	FIT	REM-0	REM	Data	FIT	REM-0	REM
0111	0.904	0.904	0.904	0.904	0.883	0.883	0.883	0.883
1011	0.899	0.898	0.899	0.900	0.897	0.893	0.893	0.893
1101	0.886	0.876	0.876	0.876	0.902	0.883	0.883	0.883
1110	0.886	0.861	0.861	0.862	0.900	0.877	0.877	0.877
0011	0.907	0.915	0.913	0.913	0.899	0.899	0.899	0.898
0101	0.901	0.883	0.881	0.881	0.901	0.884	0.881	0.881
0110	0.901	0.883	0.883	0.881	0.889	0.867	0.861	0.860
1001	0.904	0.896	0.895	0.895	0.878	0.861	0.858	0.858
1010	0.904	0.876	0.874	0.873	0.880	0.856	0.851	0.853
1100	0.886	0.840	0.840	0.840	0.897	0.844	0.844	0.844
0001	0.887	0.847	0.842	0.841	0.901	0.847	0.840	0.839
0010	0.907	0.862	0.855	0.855	0.900	0.866	0.864	0.863
0100	0.901	0.832	0.829	0.829	0.899	0.833	0.826	0.827
1000	0.904	0.840	0.839	0.839	0.883	0.823	0.815	0.815

Table 6. The correctness of the neural net classifier on the subset of incomplete cases. Data=based on original values, FIT=estimated using RAA from whole sample, REM-0 is REM after one iteration.

We computed the performance of the neural network classifier on the test set when different fractions of *cases* were considered incomplete and imputed using the REM-algorithm. This experiment was carried out for each of the 14 different combinations of missing and observed data. For each such combination, a fraction $\pi \in \{0.1,0.2,0.4\}$ of the cases were made incomplete by deleting from each case values corresponding to a particular combination of missing and observed variables. I.e. a combination in which the first and third variables are considered missing is indicated with the label 0101. Which cases in the dataset were made incomplete, was chosen at random. The cases that were to be considered incomplete, were first classified with the neural-net classifier before any values were deleted. The resulting correctness is the maximal performance that can possibly be performed (called 'Data' in the tables 6 and 7). Secondly, the covariance matrix was estimated from *all* cases and the missing values were subsequently deleted from the fraction of incomplete cases. The recurrent auto associator was used to estimate the missing values in the incomplete cases using this covariance matrix. The classification results appear in the column 'FIT' in the tables 6 and 7. Finally, the incomplete cases were imputed by REM and REM-0, REM after one iteration. The experiment was repeated as 10 different subsets of cases was considered incomplete. For each fraction of missing data π, in total 14×10 patterns of missing and observed values were investigated.

Com.	40%			
	Data	FIT	REM-0	REM
0111	0.902	0.902	0.902	0.902
1011	0.895	0.891	0.891	0.892
1101	0.899	0.887	0.887	0.887
1110	0.900	0.879	0.879	0.879
0011	0.894	0.894	0.892	0.893
0101	0.900	0.883	0.881	0.880
0110	0.904	0.877	0.876	0.875
1001	0.897	0.874	0.872	0.881
1010	0.895	0.863	0.860	0.861
1100	0.896	0.854	0.854	0.855
0001	0.898	0.851	0.846	0.844
0010	0.893	0.856	0.852	0.851
0100	0.896	0.833	0.828	0.825
1000	0.901	0.838	0.834	0.834

Table 7. Classification results with 40% missing data.

The tables 6 and 7 contain the average correctness of the neural net classifier among the 10 (incomplete) samples for 10%, 20% and 40% incomplete cases, respectively. The results indicate that the classifier performance decreases when attribute values are substituted by their predictions. However, there is almost no difference in correctness between values that were predicted with a recurrent auto associator of which its parameters were estimated from the whole data set, or the REM algorithm based on only 90%, 80% or 60% of the cases. There was no difference between REM and REM-0. Only in the experiment with 40% incomplete cases, by virtue of the larger sample available to estimate the covariance matrix, FIT performs slightly better in the situation with 3 missing values than REM and REM-0.

5 Discussion

In the first experiment, we compared estimation in complete cases using the recurrent auto associator and multivariate regression. The experiment indicated that the standard deviation of predictions from the recurrent auto associator exceeded the standard deviation of the predictions from multivariate regression with about 12% when the data contain sufficient redundancy. The experiment indicated that when the variables did not have the same variance the estimates of the recurrent auto associator were poor. In the experiment, the estimated residual variance was allowed to exceed the variance of the variable itself.

The second and third experiment together indicated that the REM- and EM-algorithms estimated the correlation matrices equally well when the fraction of missing data is 10%-40% and all variables have the same variance. Based on this observation, we conclude that the absence of a correction term in the REM-algorithm does not cause it to be outperformed by the EM-algorithm. When the variables have different variance, EM resulted in better estimates of the correlation matrix than REM. The main reason is that some missing values are estimated with a higher residual variance by the recurrent auto associator than by multivariate regression.

When the subset of cases that could not be predicted well by the recurrent auto associator were excluded the REM-algorithm performed almost as well as the EM-algorithm. The implications of using the recurrent auto associator to "filter out" incomplete cases that contain too little redundancy, has only been preliminary investigated in this article and is an issue for further research.

In the fourth experiment, the influence on the classification result of imputing missing data completed with REM was investigated. The experiment indicated that REM performs as well as a recurrent auto associator of which the parameters were derived from the covariance matrix computed on the whole sample. No difference between REM and REM-0 (REM after the first iteration) could be found. This experiment only tests the usefulness of REM in an indirect way. Whether a missing value that is predicted will cause the imputed case to be misclassified depends also on how important the missing variables are for classifying the case correctly.

It was mentioned in the introduction that the computational complexity is a drawback of the EM-algorithm. For each combination of missing and observed variables, a different set of regression coefficients needs to be estimated. From the formula for the multivariate regression coefficients W

$$W = X_m X_k^T (X_k X_k^T)^{-1} \tag{22}$$

it is evident that when the number of observed variables k is large the computational complexity of the EM-algorithm increases as the inversion of a $k \times k$ matrix requires around k^3 computations [42]. In their book on estimation of missing values, Little and Rubin suggest to use the SWEEP-operator to reduce the computation time of $(\Sigma_k)^{-1}$ [29]. Originally developed by Dempster [10], the SWEEP-operator is a numerical method for the efficient computation of multivariate regression coefficients. It

reduces the computational complexity of computing the regression coefficients to $O(m^2k^2)$.

Using the auto associator, one has first to find the g eigenvalues and eigenvectors of the sample covariance matrix, which takes around g^3 computations. We neglect this computation as it is only performed once per REM-iteration. The *same* eigenvectors are used to compute the parameters $P^{(h)}$ for each combination of variables with missing and observed values. For each combination, one needs to perform $O(m^3+m^2k)=O(m^2(m+k))$ computations to compute $P^{(h)}$. Comparing $O(m^2k^2)$ with $O(m^2(m+k))$ shows that the coefficients of recurrent auto associator for a specific combination of "missing" and "observed" values can be computed between $g/2$ and g times faster than the multivariate regression coefficients with the SWEEP-operator, with $g=k+m$. Consequently, the REM-algorithm has less computational complexity than the EM-algorithm. The complexity of the REM-algorithm can even be reduced further by using the recurrent auto associator in recurrent mode (the recurrent mapping in Eq. (8)) such that the inversion $(I-W_m)^{-1}$ is avoided. In each successive REM-step, still fewer recurrent iterations will be needed because the initial value of x^m will be closer to its final estimate.

An idea is first to estimate the correlation matrix from an incomplete dataset using REM. When the algorithm has converged, one can use multivariate regression to reestimate the missing values based on the correlation matrix estimated with REM. Such a possibility entails only one E-step in the EM-algorithm.

6 Conclusion

Many classifiers cannot cope with incomplete cases. A neural net classifier, for example, cannot be used before all input values are known. The problem of incomplete cases has been given much attention in multivariate statistics. Methods such as the EM-algorithm and Multiple Imputation have been developed to estimate missing values in incomplete datasets. Both methods are also computationally complex. Previous experiments have indicated that using the EM-algorithm to impute incomplete data that contain only little redundancy is problematic. So we tried to develop a method that can predict missing values when data contain much redundancy whereas incomplete cases in which too much information is absent, are not imputed.

We investigated the connectionistic field as our idea was to use a pattern completion technique to estimate missing data. The notion of using the auto associator in recurrent mode as an alternative to multivariate regression was explored. The weights of the auto associator can be computed from the eigenvectors of the sample covariance matrix so that a time consuming training process can be omitted.

The conditions under which the recurrent auto associator converges to stable estimates were derived. It was shown that when convergence occurs, the estimates of the missing values are the solution to a set of linear equations based on the weights of the auto associator. This solution is the parameter matrix $P^{(h)}$ with an effective

rank h equal to the number of hidden nodes or principal components in the recurrent auto associator.

We embedded the recurrent auto associator in the Recurrent Estimation Maximization (REM) algorithm, which is an iterative EM-like approach to estimating the missing values and the covariance matrix from an incomplete dataset. In a number of experiments, we compared the REM-algorithm with the EM-algorithm with respect to estimating missing values and estimating the correlation matrix from an incomplete dataset. The experiments indicate that the REM- and EM-algorithms result in almost identical estimates of the correlation matrices when the fraction of missing data is not larger than 10%. Simulations with the three covariance matrices in which all variables have identical variances also indicated that REM and EM estimated the correlation matrix with the same precision even when the percentage of missing data is as high as 40%. When the missing values can be predicted only poorly using the REM-algorithm, one could better exclude the incomplete cases with such combinations of missing values. The second and third experiments indicated that the missing values are estimated better by EM (multivariate regression) than by REM (recurrent auto associator).

Finally, we performed also an experiment with a neural-net classifier which classified incomplete cases that had been imputed by an auto associator of which the covariance matrix had been estimated from the complete data and imputed using REM. The experiment indicated that performing more REM-iterations does not lead to a higher correctness of the classifier on the incomplete cases.

The advantage of the REM-algorithm is its computational simplicity as compared with the EM-algorithm. We suggest that the REM-algorithm can be used to obtain an estimate of the correlation matrix. When REM has iterated, multivariate regression can be used to estimate the missing values in the incomplete cases. It is still an issue for research whether the recurrent auto associator lends itself as a "filter" to separate incomplete cases in which the missing values contain little extra information from incomplete cases in which vital information is missing.

Appendix A

The REM-algorithm is defined as follows:

Estimate the means $\hat{\mu}_i^{(0)} = n^{-1} \sum_{p=1}^{n} x_{i,p}^{(0)}$, $i=1,..,g$

Estimate the covariance matrix $\hat{\Sigma}^{(0)} = X^{(0)} X^{(0)\,T} - n\, \mu^{(0)} \mu^{(0)\,T}$

both from the subset of *complete* cases

E-step (Iteration c):

1. Centre each variable $x^{<k>}$ in $X^{(c)}$ around a zero mean: $x_{i,p}^{(c)} = x_{i,p}^{(c)} - \hat{\mu}_i^{(c)}$, $i=1,..,g$, $p=1,..,n$.
2. Estimate the eigenvalues Λ and eigenvectors B from $\hat{\Sigma}^{(c)}$ and sort the eigenvalues in ascending order.

For each combination of variables with missing and observed values present in X:

3. Determine h^* as $\min_{h=1,..g-m}(\check{\varepsilon}^{(h)^2})$, s.t. $n^{-1}\, \check{\varepsilon}_i^{(h^*)^2} < \sigma_{i,i}^2$ $i=1,..,m$, and compute $W^{(h^*)}$ from

$$W^{(h^*)} = \sum_{j=g-h^*+1}^{g} b_j b_j^{T}, \quad b_j \in B$$

For all cases $p=1,..,n$, if in case $x_p^{(c)}$ the combination of missing and observed values that correspond to $(m \times k)$ occurs:

4. Rearrange and split $x_p^{(c)} = (x^{k\,T}, x^{m\,T})^T$
   and iterate $\hat{x}^m(t+1) = W_{mk}^{(h^*)} x^k + W_{mk}^{(h^*)} \hat{x}^m(t)$ until $\|\hat{x}^m(t+1) - \hat{x}^m(t)\|_2 < \alpha$.
5. Set $\hat{x}_p^{m\,(c+1)} = \hat{x}^m(t+1)$

M-step (Iteration c):

1. Reestimate the mean $\hat{\mu}_i^{(c+1)} = n^{-1} \sum_{p=1}^{n} x_{i,p}^{(c+1)}$, $i=1,..,g$, and the covariance matrix

$$\hat{\Sigma}^{(c+1)} = X^{(c+1)} X^{(c+1)\,T} - n\, \hat{\mu}^{(c+1)} \hat{\mu}^{(c+1)\,T}$$

2. If $Z_\alpha > \gamma$ then goto the E-step else stop.

Correct back $X^{(c+1)}$: $x_{i,p}^{(c+1)} = x_{i,p}^{(c)} + \sum_{j=0}^{c} \hat{\mu}_i^{(j)}$, $i=1,..,g$, $p=1,..,n$.

References

[1] S. Ahmad, V. Tresp. "Some solutions to the missing feature problem in vision", *Neural Information Processing Systems*, Vol. 5, Ed. S.J. Hanson, J.D. Cowlan, C.L. Giles, Morgan Kaufmann Publishers, San Mateo, 1993. pp 393-400.

[2] T.W. Anderson. "Asymptotic theory for principal component analysis", *Annals of mathematical statistics*, Vol. 34, pp. 122-148, 1963.

[3] P. Baldi, K. Hornik. "Neural networks and principal component analysis: learning from examples without local minima", *Neural networks*, Vol. 2, pp. 53-58, 1989.

[4] E.M. Beale, R.J.A. Little. "Missing values in multivariate analysis", *Journal of Royal Statistical Society B*, Vol. 37, pp. 129-145, 1975.

[5] Y. Bengio, P. Simard, P. Frasconi. "Learning long term dependencies with gradient descent is difficult", *IEEE transactions of neural networks*, Vol. 5, No. 2, pp. 157-166, 1994.

[6] C.M. Bishop. "Mixture density networks", *Technical report NCRG/4288*, Department of computer science, Aston University, UK, 1994.

[7] S.F. Buck. "A method of estimation of missing values in multivariate data suitable for use with an electronic computer", *Journal of Royal Statistical Society B*, Vol. 22, pp. 302-306, 1960.

[8] S. van Buuren, J.L.A. van Rijckevorsel. "Imputation of missing categorical data my maximizing internal consistency" *Psychometrika*, Vol. 57, No. 4, pp. 567-580, 1992.

[9] J.A. Calvin. "REML Estimation in unbalanced multivariate variance components models using an EM algorithm, *Biometrics*, Vol. 49, pp. 691-701, Sept., 1993.

[10] A.P. Dempster. "Elements of continuous multivariate analysis", Reading, Addison Wesley, 1969.

[11] A.P. Dempster, N.M. Laird, D.B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm", *Journal of royal statistical society series B*, Vol. 39, p. 1-38, 1977.

[12] K.I. Diamantaras, S.-Y. Kung. "Multilayer Neural Networks for reduced-rank approximation", *IEEE Transactions of neural networks*, Vol. 5, No. 5, pp. 684-697, 1994.

[13] M. Egmont-Petersen, J.L. Talmon, J. Brender, P. NcNair. "On the quality of neural net classifiers", *Artificial Intelligence in Medicine (AIM-journal)*, Vol. 6, No. 5, pp. 359-381, 1994.

[14] T.C. Gleason, R. Staelin. "A proposal for handling missing data", *Psychometrika*, Vol. 40, No. 2, pp. 229-252, 1975.

[15] M.J. Greenacre. "Theory and applications of correspondence analysis", *Academic press*, London, 1984.

[16] R.F. Harrison, S.J. Marshall and R.L. Kennedy. "A connectionist aid to the early diagnosis of myocardial infarction", in: M. Stefanelli, A. Hasman, M.Fieschi and J. Talmon (Eds.), *Proceedings to the third conference on artificial intelligence in medicine, Lecture Notes in Medical Informatics 44*, Springer Verlag, Berlin, pp. 119-128, 1991.

[17] A. Hart and J. Wyatt. "Connectionist models in medicine: an investigation of their potential", in: J Hunter, J. Cookson, and J. Wyatt (Eds), *Proceedings for the AIME-89 conference, Lecture Notes in Medical Informatics 38*, Springer Verlag, Berlin, pp. 115-124, 1989.

[18] M.H. Hassoun, C. Wang, R. Spitzer. "NNERVE: Neural network extraction of repetitive vectors for electromyography-Part I: Algorithm", *IEEE Transactions on biomedical engineering*, Vol. 41, No. 11, pp. 1039-1052, 1994.

[19] J. Hertz, A. Krogh, R.G. Palmer. "Introduction to the theory of neural computation", *The Santa Fe studies in the study of complexity, Addison Wesley*, Redwood City, 1991.

[20] J.J. Hopfield, D.W. Tank. "Computing with neural circuits: A model", *Science*, Vol. 233, August, pp. 625-633, 1986.

[21] R.A. Horn, C.R. Johnson. "Matrix analysis", Cambridge university press, Cambridge, 1985.

[22] G. Hripcsak. "Using connectionistic modules for decision support", *Methods of Information in Medicine*, Vol. 29, pp. 167-181, 1990.

[23] M.I. Jordan. "Attractor dynamics and parallelism in a connectionist sequential machine", *Proceedings of the eighth annual meeting of the cognitive science society*, Hillesdale, N.J., Erlbaum, 1986.

[24] A.N. Komolgorov. "On the representation of continuous functions of several variables by superposition of continuous functions of a smaller number of variables", *Dokl. Akad.* Vol. 108, pp. 179-182, 1956.

[25] B. Kosko. "Adaptive bidirectional associative memories", *Applied optics*, Vol. 26, No. 23, pp. 4947-4960, 1987.

[26] S.Y. Kung, F. Fallside, J. Aa. Sorenson, C. A. Kaufmann (Eds). "Neural networks for signal processing II", *Proceedings of the 1992 IEEE workshop on neural networks for signal processing*, 1992.

[27] W. Ledermann, S. Vajda. "Handbook of applied mathematical analysis IV. Analysis", John Wiley & Sons, N.Y., 1982.

[28] R.J.A. Little. "Maximum likelihood inference for multiple regression with missing values: A simulation study", *Journal of royal statistical society series B*, Vol. 41, p. 76-87, 1979.

[29] R.J.A. Little, D.B. Rubin. "Statistical analysis with missing data", John Wiley & Sons, New York, 1987.

[30] E. Másson. "A parallel neural network simulator trained for image compression", Master thesis, *Nordita preprint*, Nordita – 90/50 S, Copenhagen, 1990.

[31] J.L. McClelland, D.E. Rumelhart. "Distributed memory and the representation of general and specific information", *Journal of experimental psychology: General*, Vol. 114, pp. 159-188, 1985.

[32] X.-L. Meng, D.B. Rubin. "Maximum likelihood estimation via the ECM algorithm: A general framework", *Biometrika*, Vol. 80, pp. 267-278, 1993.

[33] M. Minsky and S. Papert. "Perceptrons", MIT Press, Cambridge, MA, 1969.

[34] C. Moallemi. "Classifying cells for cancer diagnosis using neural networks", *IEEE Expert*, No. 12, pp. 8-12, 1991.

[35] D.C. Montgomery, E.A. Peck. "Introduction to linear regression analysis", *Wiley series in probability and mathematical statistics*, John Wiley & Sons, New York, 2nd Ed., 1992.

[36] R.J. Muirhead. "Aspects of multivariate statistical theory", John Wiley & Sons, New York, 1982.

[37] A.M.M. Muijtjens, J.M.A. Roos, T, Arts, A. Hasman, R.S. Reneman. "Extrapolation of incomplete marker tracks by lower rank approximation", International journal of biomedical computing, Vol. 33, pp. 219-239, 1993.

[38] T. Orchard, M.A. Woodbury. "A missing information principle: Theory and Applications", *Proceedings of the 6th symposium on mathematical statistics and probability*, Vol. 1, pp. 697-715, 1972.

[39] J.M. Ortega, W.C. Rheinboldt. "Iterative solution of nonlinear equations in several variables", Academic press, New York, 1970.

[40] E. Pelikan, F. Vogelsang, B. Schultz, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf. "Röntgenbilsegmentierung durch topologische Karten oder Multilayer-Perceptron - ein Vergleich", *Proceedings 2'nd workshop on digital image processing in medicine*, Freiburg, 1994.

[41] R. Poli, S. Cagnoni, R. Livi, G. Coppini, G. Valli. "A neural network expert system for diagnosing and treating hypertension", *IEEE Computer*, No. 3, pp. 64-71, 1991.

[42] W.H. Press, B.P. Flannery, S.A. Teukolsky, V.T. Vetterling, "Numerical recipes in C", *Cambridge university press*, New York, 1988.

[43] J.R. Quinlan. "Unknown attribute values in induction", *Proceedings of the sixth international machine learning workshop*, pp. 164-164, San Mateo, California, Morgan Kaufmann, 1989.

[44] C.R. Rao. "Linear statistical inference and its applications", John Wiley & Sons, New York, 2nd ed., 1973.

[45] M.D. Richard, R.P. Lippmann, "Neural network classifiers estimate bayesian a posteriori probabilities", *Neural computation*, Vol 3, pp. 461-483, MIT Press, 1991.

[46] B.D. Ripley. "Statistical aspects of neural networks", pp. 40-123, In: O.E. Barndorff-Nielsen, J.L. Jensen, W.S. Kendall (Eds.), *Networks and Chaos – Statistical and Probabilistic Aspects*, Chapman & Hall, 1993.

[47] D.B. Rubin. "Inference and missing data", *Biometrika*, Vol. 63, No. 3, pp. 581-592, 1976.

[48] D.B. Rubin. "Multiple imputation for nonresponse in surveys", John Wiley & Sons, New York, 1987.

[49] D.B. Rubin, N. Schenker. "Multiple imputation for interval estimation from simple random variables with ignorable nonresponse", *Journal of the American Statistical Association*, Vol. 81, No. 394, pp. 366-374, 1986.

[50] D.E. Rumelhart, D.E. McClelland. "Parallel distributed processing: Explorations into the microstructure of cognition", Vol. 1, MIT Press, New Jersey, 1986.

[51] T.D. Sanger. "Optimal unsupervised learning in a single-layer linear feedforward neural network", *Neural networks*, Vol. 2, pp. 459-473, 1989.

[52] R.J. Serfling. "Approximation theorems of mathemical statistics", Wiley & Sons, New York, 1980.

[53] C. N. Schizas, C.S. Pattchis, I.S. Schofield, and P.R. Fawcett. "Artificial Neural Nets in Computer-Aided Macro Motor Unit Potential Classification", *Trans. IEEE Engineering in Medicine and Biology*, pp. 31-38, 1990.

[54] T. Schiøler, W. Grimson, P. Sharpe, M. Egmont-Petersen, G. Momsen, R. O'Moore and P. McNair. "Automatic decision support based on voting by independent decision support systems", pp. 58-66, *Proceedings Computing in Clinical Laboratories 1992*, 1992.

[55] M.F. Schlang, V. Tresp. "Neural networks for segmentation and clustering of biomagnetical signals", In: S.Y. Kung, F. Fallside, J. Aa. Sorenson, C. A. Kaufmann (Eds). "Neural networks for signal processing II", *Proceedings of the 1992 IEEE workshop on neural networks for signal processing*, pp. 343-349, 1992.

[56] G. Strang. "Linear algebra and its applications", Harcourt Brace Jovanovich Publishers, San Diego, 3. ed, 1988.

[57] V. Tresp, S. Ahmad, R. Neuneier. "Training neural network with deficient data", *Neural Information Processing Systems*, Vol. 6, Ed. J.D. Cowan, G. Tesauro, J. Alspector, Morgan Kaufmann Publishers, San Mateo, 1994. pp 128-135.

[58] J. Wyatt. "Clinical data systems, part 1: data and medical records", *The Lancet*, Vol. 344, Dec. 3, pp. 1543-1547, 1994.

[59] J. Wyatt. "Clinical data systems, part 3: development and evaluation", *The Lancet*, Vol. 344, Dec. 17, pp. 1683-1688, 1994.

[60] F. Vogelsang. "Segmentierung radiologisch dokumentierter fokaler Knochenläsionen auf Basis kontextbezogener Vektoren mit neuronalen Netzwerken", *Diplomarbeit* (Master Thesis), Fakultät für Informatik, Medizinische Fakultät, RWTH Aachen, 1993.

[61] F. Vogelsang, E. Pelikan, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf. "Segmentierung von Röntgenbildern fokaler Knochenläsionen durch neuronale Netzwerke – Optimierung durch Quality metrics und modifizierte Contribution Analysis", pp. 450-459, In: S.J. Pöppl, H. Handels (Eds.), *Proceedings of the GMDS workshop, Mustererkennung*, Springer Verlag, Berlin, 1994.

Robustness metrics for measuring the influence of additive noise on the performance of statistical classifiers

6

Authors: M. Egmont-Petersen, J.L. Talmon, A. Hasman

Submitted for publication.

Abstract:
This paper presents a novel quality measure called robustness. The robustness measure quantifies the influence of measurement noise in the attribute values on the credibility of the classification of a case. It is assumed that the type of distribution of the noise-generating process is known. It is not simple to measure the robustness in the general situation where the noise-free distribution of the attributes is unknown. Therefore, two approximations are suggested and compared with the robustness measure based on the noise-free distribution of the attributes. The usefulness of the suggested robustness measure is explored in a simulation experiment.

1 Introduction

Statistical classifiers have been developed for various medical classification tasks including diagnosis and therapy. The classification of a case can be based on many different types of attributes such as biochemical assays, bioelectric signals (EMG, ECG, EEG, etc.), patient history and clinical signs and symptoms. Some attributes are measured under uncertainty as noise is inherent in the measurement process. Biochemical assays, for example, are usually contaminated by measurement noise.

Often, the amount of measurement noise can be influenced through quality control programs or simply by repeated measurements. Thereby, it is possible to increase the credibility of a tentative diagnosis based on attributes that are contaminated by measurement noise. Improving the "signal-to-noise-ratio" (SNR), however, has its price. Measurement noise may only have influence on the decision when the measured value is observed close to a decision boundary relative to the variance of the measurement noise. When the measurement is far from any decision boundary, remeasuring the same sample will not lead to a change in classification.

In general, an object is represented by a set of attribute values that are observed from some underlying distribution. Due to measurement noise, the observed values differ from the true (but unknown) values. When a statistical classifier uses attributes that are measured with noise and the type of distribution of the noise generating process as well as its parameters are known, it is possible to quantify the uncertainty of a classification in relation to the measurement noise of the attributes. In this context we define:

The robustness of a classification of a case is the probability that the case would obtain the same class label if the (unknown) true attribute values were known.

The concept of *robustness* of classifications was first introduced by Brender *et al.* suggesting it can be measured for set of cases and hence is a property of a classifier [1]. The probability that a different class label can be obtained when one or more attributes is remeasured varies from case to case as this probability depends on how close the case is to the decision boundary. Our definition of robustness is a property of a classification of a particular case. Robustness resembles *confidence* as defined by Willard and Critchfield [4]. They measure the confidence of a classification of a case in relation to the variance of the parameters of the classifier. The difference is that the robustness measure assesses the uncertainty of a classification given the uncertainty inherent in the noisy measurements while the *confidence* measure relates the classification of a case to the uncertainty with which the parameters of the classifier were estimated.

In the following, we restrict ourselves to situations where the attributes are real valued. We assume that the distribution of the measurement noise in each attribute is Gaussian with a known variance. First, we give a mathematical definition of the robustness of a classification. Secondly, we analyze two two-class problems in the special situations: 1) where the attributes are normally distributed (unimodal) and 2) where the class-conditional distributions are Gaussians (bimodal). Unbiased estimation of the robustness requires knowledge of the type and parameters of the distribution of the true, noise-free, attribute values. In the general case, this information is seldomly

available. We therefore suggest two metrics that approximate the robustness measure. We will show to what extent these approximations result in biased estimates of the robustness for the unimodal and bimodal situations. Next, we discuss how the robustness measure can be used to guide the improvement of the SNR by means of repeated measurements. It is analyzed how often an attribute value has to be remeasured to ensure a classification with sufficient robustness.

2 Measuring the robustness

A classification task can be defined as a mapping from an n-dimensional attribute space to a c-dimensional class space. Let the c classes be characterized by the class-conditional probability density functions (PDFs) of the true, noise-free, attribute values $p_t(\boldsymbol{t}|\omega_j)$[1], $j=1,..,c$. Let the corresponding PDF be defined as $p_t(\boldsymbol{t})=\Sigma_{j=1,..,c} P(\omega_j)p_t(\boldsymbol{t}|\omega_j)$. Denote with $p(\boldsymbol{o}|\boldsymbol{t})$ the PDF of the measurement noise: the distribution of observable attribute values $\boldsymbol{o}$ given the true attribute values $\boldsymbol{t}$.

In the following, we will assume that the measurement process induces Gaussian noise with zero mean and that the noise in one attribute is independent of the noise in the other attributes as well as of the true attribute values $\boldsymbol{t}$:

$$p(\boldsymbol{o} \mid \boldsymbol{t}) = (2\pi)^{-n/2} \mid \Sigma_m \mid^{-0.5} \exp\left(-0.5(\boldsymbol{o}-\boldsymbol{t})^{\mathrm{T}} \Sigma_m^{-1} (\boldsymbol{o}-\boldsymbol{t})\right) \tag{1}$$

with Σ_m a diagonal matrix, with entry (i,i), $i=1,..,n$, the variance of the measurement noise of attribute i.

The PDF $p_o(\boldsymbol{o})$ of the measured (noisy) attribute values is given by the convolution of $p(\boldsymbol{o}|\boldsymbol{t})$ with $p_t(\boldsymbol{t})$

$$p_o(\boldsymbol{o}) = \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} p(\boldsymbol{o} \mid \boldsymbol{t})\, p_t(\boldsymbol{t})\, d\boldsymbol{t} \tag{2}$$

Knowledge about which true attribute values $\boldsymbol{t}$ could lead to the observed values $\boldsymbol{o}$ would allow us to compute the robustness. Using Bayes' rule we obtain

$$p(\boldsymbol{t} \mid \boldsymbol{o}) = \frac{p(\boldsymbol{o} \mid \boldsymbol{t})\; p_t(\boldsymbol{t})}{p_o(\boldsymbol{o})} \tag{3}$$

According to our definition, the robustness of a case classified as class j is given by

[1]The subscript 't' indicates that $p_t(\boldsymbol{t}|\omega_j)$ is the PDF of the attribute values without measurement noise.

$$\varrho = \int_{S_j} p(t \mid o)\, dt \tag{4}$$

with S_j the region in the input space for which the classifier assigns class label j.

Simplifying to a situation with one attribute t, we may rewrite the robustness measure as

$$\varrho = \int_{S_j} p(t \mid o)\, dt = \int_{S_j} \frac{p(o \mid t)\, p_t(t)}{p_o(o)}\, dt \tag{5}$$

Using the fact that $p_o(o)$ is constant, the robustness can also be written as

$$\varrho = \frac{\int_{S_j} p(o \mid t)\, p_t(t)\, dt}{\int_{-\infty}^{\infty} p(o \mid t)\, p_t(t)\, dt} \tag{6}$$

It is clear that the robustness of a classification based on a contaminated observation o, is the probability that t is located in the range S_j.

3 Robustness in two simple situations

Many (intermediate) medical decisions rely on the interpretation of *one* attribute value. In different clinical disciplines, the specialist has to make decisions based on noisy measurements. Whenever he thinks that no credible decision can be made based on the available information, he can collect additional information or remeasure the quantity. The latter decision is sensible when the noisy measurement is close to a decision boundary as in that case remeasuring the quantity can lead to a different decision.

The attribute used can often be modelled with either a unimodal or a bimodal distribution. In the unimodal case, one often wants to identify whether the measurement exceeds a certain threshold. This can be the case e.g. for diseases that gradually develop. In the bimodal situation the decision is whether a case belongs to one or another category, assuming that the case is from either of the two distributions. Such situations occur when a clinical condition is either present or not. Bimodally distributed attributes could for example be clinical chemical tests where each group may often be characterized by a normal distribution.

3.1 The unimodal situation

A simple classification task is to discriminate two groups with an attribute t that is unimodally distributed. When the attribute is normally distributed, it can be shown that $p(t|o)$ is a Gaussian PDF. Let $p_t(t)$ be

$$p_t(t) = (2\pi\sigma_t^2)^{-0.5} \exp\left(-\frac{(t-\mu_t)^2}{2\sigma_t^2}\right) \tag{7}$$

with σ_t^2 the variance and μ_t the mean of the noise-free attribute values. Using the fact that the convolution of two Gaussian densities is also a Gaussian density, $p_o(o)=p(o|t)*p_t(t)$, with a variance equal to the sum of their variances, $p(t|o)$ becomes

$$p(t \mid o) = \frac{(2\pi\sigma_m^2)^{-0.5} \exp\left(-\frac{(o-t)^2}{2\sigma_m^2}\right) (2\pi\sigma_t^2)^{-0.5} \exp\left(-\frac{(t-\mu_t)^2}{2\sigma_t^2}\right)}{(2\pi(\sigma_t^2+\sigma_m^2))^{-0.5} \exp\left(-\frac{(o-\mu_t)^2}{2(\sigma_t^2+\sigma_m^2)}\right)} \tag{8}$$

which simplifies to

$$p(t \mid o) = (2\pi\sigma_{mt}^2)^{-0.5} \exp\left(-\frac{\left(t-\frac{o+\mu_t\,\sigma_m^2/\sigma_t^2}{1+\sigma_m^2/\sigma_t^2}\right)^2}{2\sigma_{mt}^2}\right) \tag{9}$$

with

$$\sigma_{mt}^2 = \frac{\sigma_m^2\,\sigma_t^2}{\sigma_m^2+\sigma_t^2} \tag{10}$$

The standard deviation of the distribution $p(t|o)$, σ_{mt}, is smaller than the standard deviation of $p(o|t)$, σ_m. The mean of $p(t|o)$

$$\frac{o+\mu_t\,\sigma_m^2/\sigma_t^2}{1+\sigma_m^2/\sigma_t^2} \tag{11}$$

is different from o, the measured attribute value, when $o \neq \mu_t$.

In the situation where the two classes are discriminated by a threshold λ, $S_1=[-\infty,\lambda]$ and $S_2=[\lambda,\infty]$, the robustness of a classification based on the measurement that indicated class label $1 - o \in S_1$ – is given by

$$\varrho = \int_{-\infty}^{\lambda} p(t \mid o)\, dt \tag{12}$$

with $p(t|o)$ as defined in (9).

Figure 1 shows the density $p(t|o)$, o=1, μ_t=0, σ_t^2=1.0, for three different noise levels, σ_m^2=0.0009, 0.01, 0.09. The graph clearly shows that the deviation between the mean of $p(t|o)$ and o increases as a function of the noise level.

Figure 2 shows the robustness for different values of o, μ_t=0, and the same three different noise levels as used above, σ_t^2=1.0. The threshold λ is set to 1.

An interesting observation is that the robustness of classifications on both sides of the boundary do not approach the same value when the measured attribute value approaches λ from the left and right side, respectively. This is explained by the fact that the mean of $p(t|o)$ is different from o and hence

$$\lim_{\varepsilon \to 0} \int_{-\infty}^{\lambda} p(t \mid \lambda - \varepsilon)\, dt \neq \lim_{\varepsilon \to 0} \int_{\lambda}^{\infty} p(t \mid \lambda + \varepsilon)\, dt \tag{13}$$

Another observation in figure 2 is that for observations smaller than λ the robustness for a high level of measurement noise is not always smaller than the robustness for a

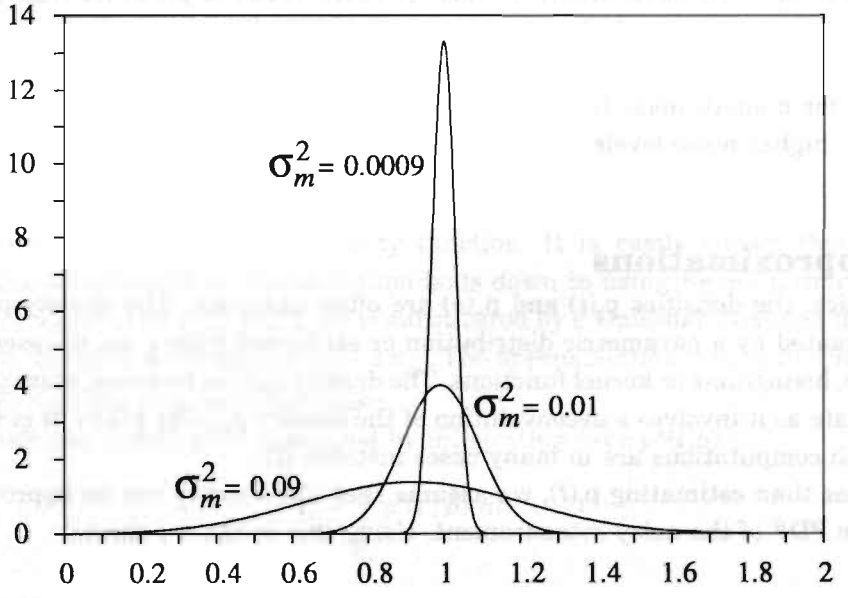


Figure 1. The distribution $p(t|o)$ with o=1, μ_t=0, for three different noise levels.

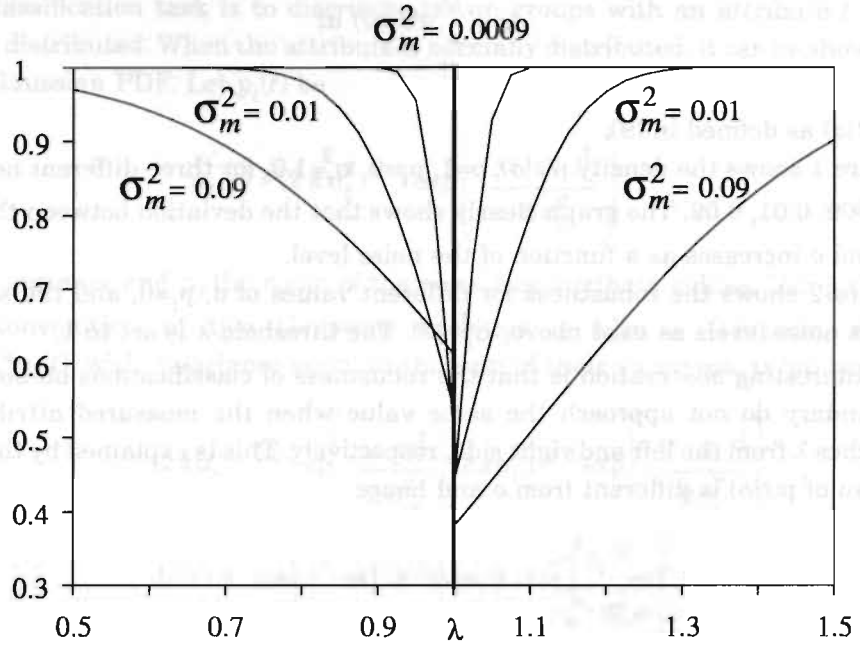


Figure 2. The robustness computed for three different noise levels. The threshold separating the two classes λ=1.

low noise level, i.e. the robustness curves cross. For high noise levels, the area of the upper tail of $p(t|o)$ beyond λ becomes significant, even when o is at a considerable distance from λ. For measurements close to λ, the mean of $p(t|o)$ for high noise levels will be further away from λ than for lower noise levels. When o is close enough to λ, this difference in means outweighs the difference in variance and the area of the upper tail of $p(t|o)$ for a small noise level will become larger than the area of the upper tail of $p(t|o)$ for higher noise levels.

3.2 Approximations

In practice, the densities $p_t(t)$ and $p_o(o)$ are often unknown. The density $p_o(o)$ can be approximated by a parametric distribution or estimated from a set of cases using, for example, histograms or kernel functions. The density $p_t(t)$ is, however, more problematic to estimate as it involves a deconvolution of the density $p_o(o)$ by $p(o|t)$. It is well-known that such computations are in many cases instable [2].

Rather than estimating $p_t(t)$, we assume that this density can be approximated by $p_o(t)$, the PDF of the noisy measurement. Using this in (3), we obtain

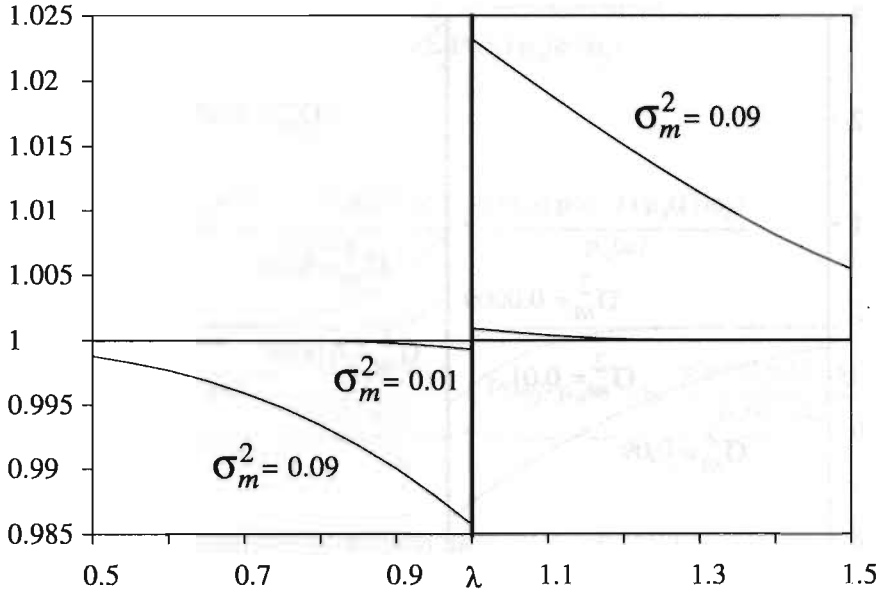


Figure 3. The ratio of the robustness and its approximation, ϱ/ϱ^*, computed for two noise levels.

$$p^*(t \mid o) = \frac{1}{c(o)} \frac{p(o \mid t)\ p_o(t)}{p_o(o)} \tag{14}$$

with

$$c(o) = \int_{-\infty}^{\infty} \frac{p(o \mid t)\ p_o(t)}{p_o(o)} \tag{15}$$

such that $p^*(t|o)$ is a probability density function. It is easily shown that in the unimodal Gaussian case this approximation boils down to using Bayes' formula (3) in which $p_t(t)$ is replaced by $p_o(t)$ and $p_o(o)$ is substituted by a Gaussian that has the same mean as $p_o(o)$ but with a variance of $\sigma_t^2 + 2\,\sigma_m^2$. The approximation results in robustness values with twice the measurement noise.

The robustness metric ϱ^* is computed by integrating over $p^*(t|o)$

$$\varrho^* = \int_{S_j} p^*(t \mid o)\ dt \tag{16}$$

Figure 3 shows the ratio ϱ^*/ϱ in the situation where $p_t(t)$ is a normal distribution, λ=1. The graph illustrates that the bias increases as o approaches the decision boundary λ.

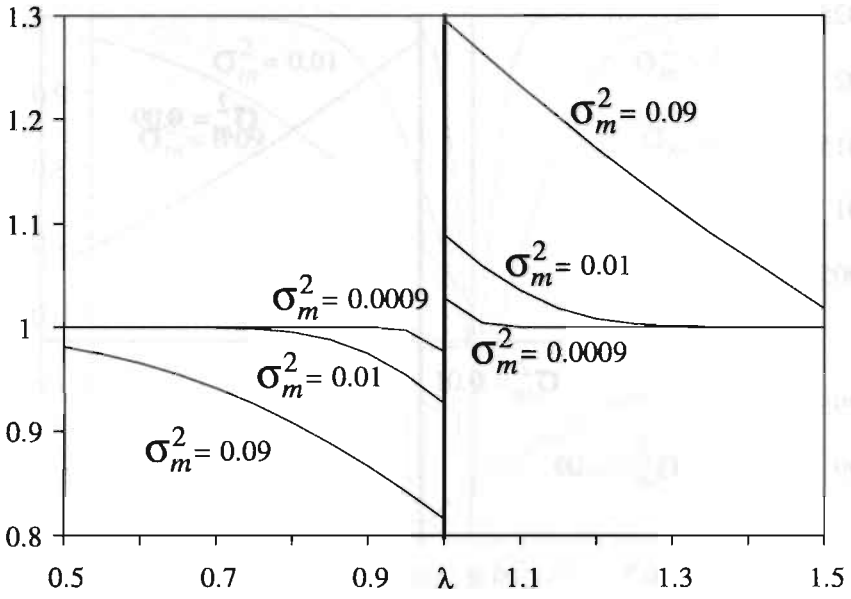


Figure 4. The ratio ϱ^{**}/ϱ between the robustness and the second approximation computed for three different noise levels.

For a variance σ_m^2=0.09, the induced bias is at most 2.5%. Note also the asymmetry around λ.

Another rough approximation would be to compute of the robustness using $p(o|t)$:

$$\varrho^{**} = \int_{S_j} p(o|t)\,dt \tag{17}$$

Figure 4 shows the ratio ϱ^{**}/ϱ in a setting similar to figure 3. The graph illustrates that ϱ^{**} is a much more biased estimate of the robustness than ϱ^{*}. Therefore, in the unimodal situation, we recommend to use the approximation ϱ^{*} which results in maximally 2.5% bias for a SNR (defined as σ_t^2/σ_m^2) equal to 10.

3.3 The bimodal situation

Another typical classification problem in medicine is the discrimination of two groups where each group is unimodally distributed. For a two-class problem with the class-conditional densities $p_t(t|\omega_1)$ and $p_t(t|\omega_2)$, $p_t(t)=P(\omega_1)p_t(t|\omega_1)+P(\omega_2)p_t(t|\omega_2)$ is the density of the attribute without measurement noise and $P(\omega_1)$ and $P(\omega_2)$ the prior probabilities of the two classes. The robustness is computed by integrating t over the density

$$p(t \mid o) = \frac{p(o \mid t) \sum_{j=1}^{2} P(\omega_j)\, p_t(t \mid \omega_j)}{\sum_{j=1}^{2} P(\omega_j)\, p_o(o \mid \omega_j)} \tag{18}$$

This can be written as

$$\frac{P(\omega_1)\, p(o \mid t)\, p_t(t \mid \omega_1)}{p_o(o)} + \frac{P(\omega_2)\, p(o \mid t)\, p_t(t \mid \omega_2)}{p_o(o)} \tag{19}$$

and

$$\frac{P(\omega_1)\, p_o(o \mid \omega_1) \dfrac{p(o \mid t)\, p_t(t \mid \omega_1)}{p_o(o \mid \omega_1)}}{p_o(o)} + \frac{P(\omega_2)\, p_o(o \mid \omega_2) \dfrac{p(o \mid t)\, p_t(t \mid \omega_2)}{p_o(o \mid \omega_2)}}{p_o(o)} \tag{20}$$

Using (3), this expression can be written as

$$\frac{P(\omega_1)\, p_o(o \mid \omega_1)\, p(t \mid o, \omega_1)}{p_o(o)} + \frac{P(\omega_2)\, p_o(o \mid \omega_2)\, p(t \mid o, \omega_2)}{p_o(o)} \tag{21}$$

which simplifies to

$$p(t \mid o) = \frac{p(t \mid o, \omega_1)}{1 + \dfrac{P(\omega_2)\, p_o(o \mid \omega_2)}{P(\omega_1)\, p_o(o \mid \omega_1)}} + \frac{p(t \mid o, \omega_2)}{1 + \dfrac{P(\omega_1)\, p_o(o \mid \omega_1)}{P(\omega_2)\, p_o(o \mid \omega_2)}} \tag{22}$$

It can be seen that the distribution of t given an observation o is a weighted sum of the class-conditional distributions $p(t|o,\omega_1)$ and $p(t|o,\omega_2)$ each of which corresponds to the density derived in (9). The two distributions $p(o|\omega_1)$ and $p(o|\omega_2)$ and the prior probabilities determine the two weights. When o is observed far from the decision boundary one weight will be close to 1 and the other close to 0. Nearby the minimal error boundary, however, both weights approach ½.

In this situation, we have used the same approximations as in the previous section. Figure 5 shows the ratios ϱ^*/ϱ and ϱ^{**}/ϱ in the bimodal situation where $p_t(t|\omega_1)$ and $p_t(t|\omega_2)$ are normal distributions, $\lambda=1$, $\sigma^2_{t|\omega_1}=\sigma^2_{t|\omega_2}=1$, $\sigma^2_m=0.09$. Only observations on one side of the decision boundary λ are shown. The two curves are symmetric round the decision boundary as the prior probabilities are equal, $P(\omega_1)=P(\omega_2)$, as are the variances of the two Gaussian distributions $p(t|o,\omega_1)$ and $p(t|o,\omega_2)$. In this situation, the robustness computed by ϱ, ϱ^* and ϱ^{**} are symmetrical around the decision boundary. The graph in figure 5 indicates that even for small signal-to-noise ratios the bias induced by the two metrics can be neglected, it is maximally 1‰.

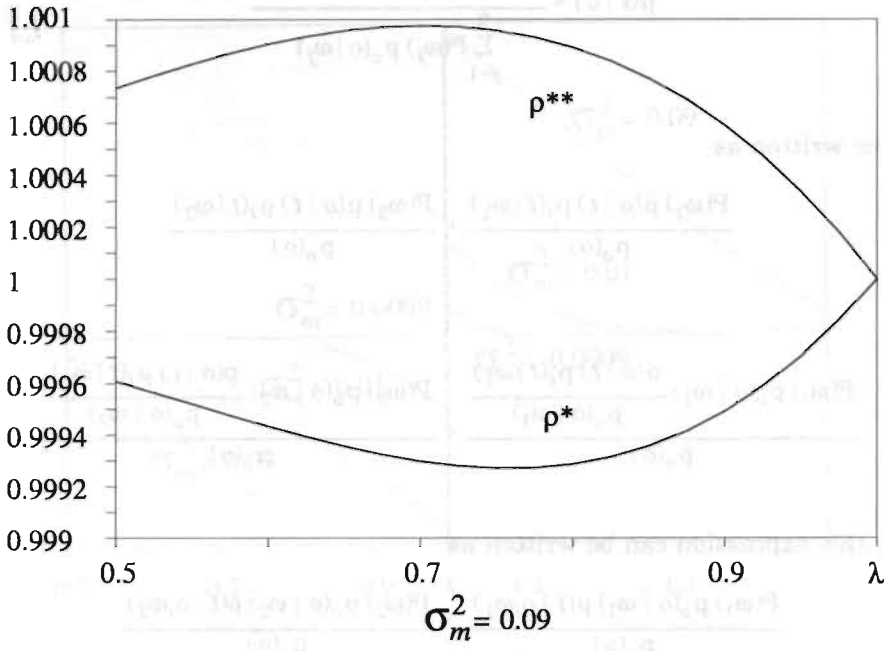


Figure 5. The relation between the robustness and the approximations suggested ϱ*/ϱ and ϱ**/ϱ computed for the highest noise level in the symmetric bimodal situation.

Other analyses indicated that when the two class-conditional distributions have different variances, or when the prior probabilities were unequal, the approximation with the metric ϱ** deviates quicker than the approximation with the metric ϱ*. When the prior probability of one of the classes becomes vary small, the unimodal situation is approached.

4 Application

The robustness measure can be useful in two situations. Firstly, it can be used to identify which attributes are the most critical for the classification of a case. Secondly, the robustness measure can be used as a means for quality improvement. It is possible to increase the robustness of a classification by repeated measurements.

It is well-known that the average of r independent measurements approaches a normal distribution with a variance σ_t^2/r. The PDF $p(o|t)$ becomes

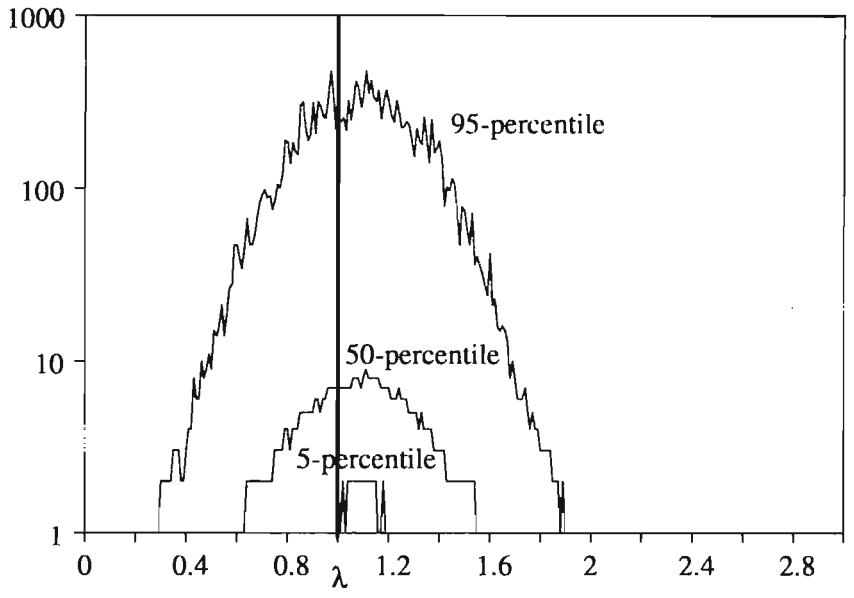


Figure 6. The 5, 50 and 95-percentiles of the number of repeated measurements necessary to obtain a robust classification as a function of o. Experiment 1.

$$p(\bar{o} \mid t) = (2\pi\sigma_m^2/r)^{-0.5} \exp\left(-\frac{(\bar{o} - t)^2}{2\sigma_m^2/r}\right) \tag{23}$$

with $\bar{o}$ the average of the r measurements. Substituting the density $p_o(o)$ with the density $p_{o|r}(\bar{o})$using Bayes' rule, the robustness of the classification based on $\bar{o}$ can be computed from

$$\varrho = \int_{S_j} \frac{p(\bar{o} \mid t)\, p_t(t)}{p_{o|r}(\bar{o})}\, dt \tag{24}$$

or in a similar manner using one of the two approximations.

We performed three simulation experiments with a unimodally distributed attribute. The purpose was to investigate how often the attribute has to be remeasured to ensure a classification with a robustness higher than β. In the first experiment, the following parameter settings were used: μ_t=0, σ_t^2=1, λ=1, σ_m^2=0.09 and β=0.975. The measured attribute value o was systematically varied from 0 to $3\sigma_t^2$ in steps of 0.01. For each value of o, we computed the robustness. When the robustness of a classification was less than β, we generated 1000 realizations of possible values from $p(t|o)$. For each of these realizations, a value was drawn from $p(o|t)$ and the average $\bar{o}$ was computed. New

values were drawn from $p(t|o)$ and $\bar{o}$ recomputed until a classification with a robustness higher than α could be obtained. So each value of t resulted in a distribution of the number of necessary remeasurements. This procedure is schematically shown in panel 1. We plotted the 5, 50 and 95-percentiles of these 1000 estimates of n for each value of o.

Quality control experiment procedure

```
for o=0 to 3*var(t) step 0.01 do
  for i=1 to 1000 do
    t=draw(p(t|o))
    o'=draw(p(o|t))
    n=1
    while robustness(o')<β and n<1000 do
      o''=draw(p(o''|t))
      o'=(n×o'+o'')*(n+1)
      n=n+1
    end while
    print(o,n)
  end for i
end for o
```

Panel 1. Procedure followed in quality control experiments.

In the second experiment, the parameter settings were: $\mu_t=0$, $\sigma_t^2=1$, $\lambda=2$, $\sigma_m^2=0.09$ and $\beta=0.975$. In the third experiment, the parameter settings were: $\mu_t=0$, $\sigma_t^2=1$, $\lambda=1$, $\sigma_m^2=0.01$ and $\beta=0.975$.

The figures 6 to 8 indicate how often one has to remeasure the attribute in the three situations to obtain a robust classification. Note the logarithmic scale of the ordinate. The first simulation indicates that for $0.8\leq o\leq 1.4$, the 50-percentile of the measurements is already larger than 3. The second simulation gave a similar result; when $1.9\leq o\leq 2.5$, the 50-percentile of the measurements is already larger than 3. In the third simulation, the corresponding interval is $0.9\leq o\leq 1.1$.

In general, this type of simulations give insight in for which ranges of o remeasurement is sensible for specific measurement costs. On each side of the class boundary there is an interval bounded by, at one side observations that always result in robust classifications. To the other side of this interval are values of o for which a robust classification is unlikely to be obtained even when the attribute is remeasured. In the latter situation, a decision should rely on other sources of information. We define these two intervals $I_{left}=[\gamma_{left},\eta_{left}]$ and $I_{right}=[\gamma_{right},\eta_{right}]$ as the *remeasuring* intervals of the noisy attribute o, with $\gamma_{left},\eta_{left},\gamma_{right},\eta_{right} \in (\mu_t,\infty)$ when $\lambda>\mu_t$. The remeasuring intervals I_{left} and I_{right} are determined by the variances σ_t^2 and σ_m^2 the decision boundary λ and the minimally required robustness β. When an attribute is observed in either I_{left} or I_{right}, it makes sense to remeasure it. The bounds of I_{left} and I_{right} can for a specific measurement situation be determined through simulations.

The three figures show that the maxima of the percentile curves appear at values larger than the decision boundary λ. This effect is caused by the 'regression to the mean' effect. When one extreme observation is made, the probability of obtaining again an observation that is just as extreme, is smaller than 0.5. Determining the remeasuring interval of an attribute using the same type of simulations as performed here, takes this effect into account.

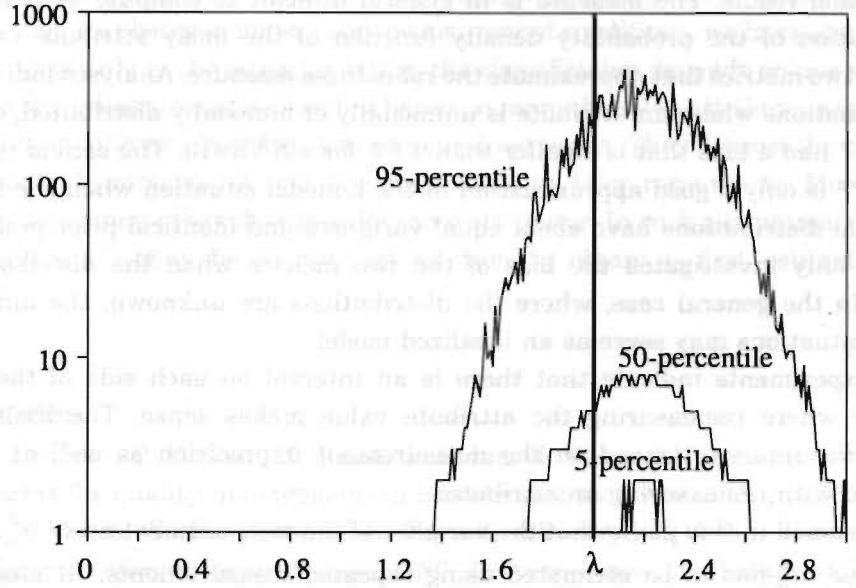


Figure 7. Number of repeated measurements in second simulation experiment.

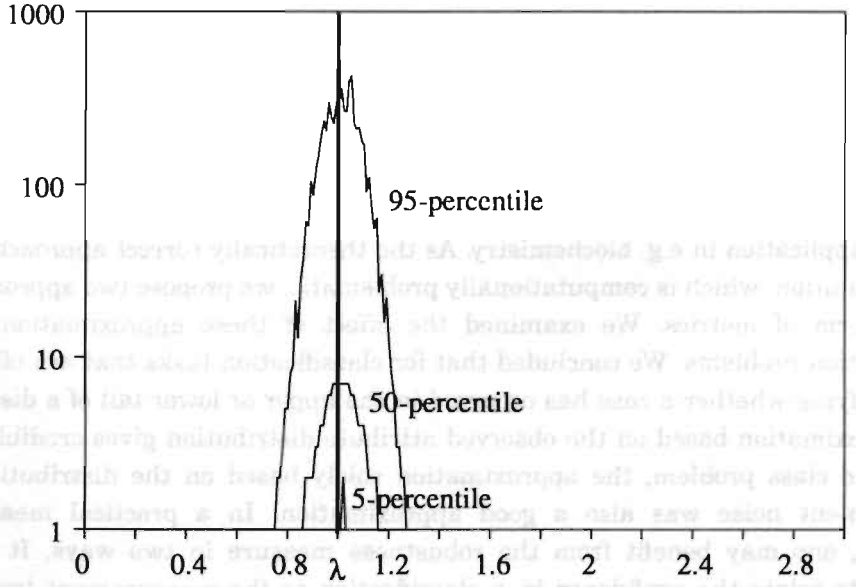


Figure 8. Number of repeated measurements in simulation 3.

5 Discussion

We have introduced a measure to quantify the influence of measurement noise on the classification result. The measure is in general difficult to compute as it involves a deconvolution of the probability density function of the noisy attribute values. We proposed two metrics that approximate the robustness measure. Analyses indicated that in the situations where an attribute is unimodally or bimodally distributed, one of the metrics ϱ^* had a bias that is smaller than 2.5% for a SNR=10. The second robustness metric ϱ^{**} is only a good approximation in the bimodal situation when the two class-conditional distributions have about equal variances and identical prior probabilities. We have only investigated the bias of the two metrics when the distributions are normal. In the general case, where the distributions are unknown, the unimodal or bimodal situations may serve as an idealized model.

The experiments indicate that there is an interval on each side of the decision boundary where remeasuring the attribute value makes sense. The limits of this *remeasuring* interval depend on the measurement imprecision as well as the costs associated with remeasuring an attribute.

We assumed in this paper that the variance of the measurement noise σ_m^2 is known. In practice, σ_m^2 has to be estimated using repeated measurements. In most clinical chemistry laboratories, for example, it is common to estimate the measurement noise of the biochemical assays as part of the quality control procedures. In such a situation, reliable estimates of σ_m^2 are available and the robustness of a diagnosis based on one or more noisy measurements can be computed. If the attribute is for example the pixel intensity in a digital radiograph, the measurement noise can be estimated as the intensity variation in a homogeneous part of the image (e.g. the background).

6 Conclusion

We have presented a novel measure for the quality aspect of the classification of a case called robustness. It expresses the probability that the true, noise-free, attribute value would receive the same class label as the observed ones. We examined the properties of the robustness for two classification problems that can be seen as prototypical for medical application in e.g. biochemistry. As the theoretically correct approach involves a deconvolution, which is computationally problematic, we propose two approximations in the form of metrics. We examined the effect of these approximations for the classification problems. We concluded that for classification tasks that are often based on identifying whether a case has occurred in the upper or lower tail of a distribution, the approximation based on the observed attribute distribution gives credible results. For a two class problem, the approximation solely based on the distribution of the measurement noise was also a good approximation. In a practical measurement situation, one may benefit from the robustness measure in two ways. It makes it possible to relate the confidence in a classification to the measurement imprecision. Secondly, the robustness measure can be used to identify remeasuring intervals on each side of a decision boundary for which remeasuring the attribute makes sense. If a

measurement is too close to the boundary, one might be better off performing another test.

Although the robustness measure was defined in the general situation with n attributes that are observed under noisy measurement conditions, we have analyzed the approximations only in the situation where the classification depends on one attribute. Measuring the robustness of a classifier based on more than one attribute is in practice not trivial for nonlinear classifies such as neural networks. They discern the classes by n-dimensional hypersurfaces and one may have to take recourse to Monte Carlo integration to estimate the robustness for more attributes. In such a situation, the rough approximation ϱ^{**} may be a practical solution to obtain a first estimate of the robustness.

References

[1] J. Brender. P. McNair, H. Raun, J. Nolan, S. Vingtoft. "Meta-knowledge as a means for quality management in knowledge-based systems", pp. 360-368, In: R. O'Moore, S. Bengtsson, J.R. Bryant, J.S. Bryden (Eds.), *Proceedings of Medical Informatics in Europe 1990*, Lecture Notes in Medical Informatics, Springer Verlag, Berlin, 1990.

[2] R.C. Gonzalez, R.E. Woods. *Digital image processing*, Addison-Wesley, Mass., 1992.

[3] E. Parzen. *Modern Probability Theory and its Applications*, John Wiley & Sons, New York, 1960.

[4] K.E. Willard, G.C. Critchfield. "Probabilistic analysis of decision trees using symbolic algebra", *Decision making*, Vol. 6, pp. 93-100, 1986.

General conclusion and discussion

7

Conclusion

The objective of this dissertation is to specify and explore methods and techniques to support development, verification and validation of neural-net classifiers. Various problems are addressed including quality assessment, attribute assessment and imputation of missing values that may arise in a development situation.

1 Development of neural-net classifiers

On the one hand, it is easy to train a neural network to classify a set of cases. On the other hand, it is very difficult to verify and validate the knowledge learned by the neural network. In the context of the iterative development process presented in chapter 1, it is indicated where the presented methods and techniques can be used (See figure 1).

Specification

The quality concepts presented in chapter 2 were conceived with the intention to help users to identify needs and requirements for a clinical decision aid. The concepts characterize possible requirements which may be important in a specific situation. The requirements can be specified quantitatively. An example is the requirement that the classifier should classify 95% of the cases *correctly*. The new

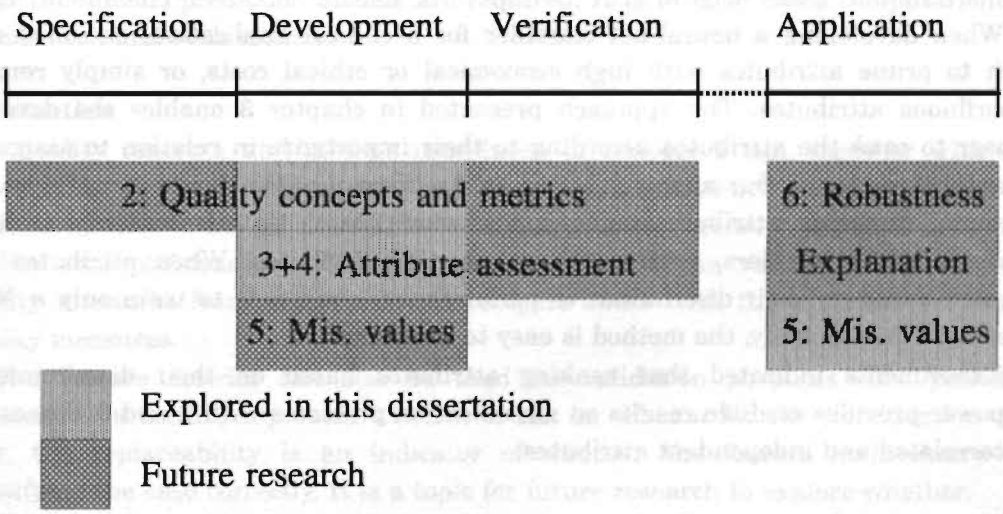


Figure 1. Different phases in the development of a decision aid that can be supported by the methods and techniques that are proposed.

quality concepts have not been used to specify requirements to a classifier developed for a clinical application. It is a topic for future research to investigate:

- The suitability of the quality concepts to help identifying user needs and requirements.

Clearly, not all kinds of requirements for a clinical decision aid can be formulated in terms of the quality concepts. The quality concepts and metrics defined in chapter 2 could be embedded in a large framework which can comprise also other properties of decision aids than their performance.

Development

The already existing metrics together with the new metrics give insight into the quality of a (neural-net) classifier. The quality metrics defined in chapter 2 are in a development situation useful for analyzing and comparing different (neural-net) classifiers. They provide information about:

- which classes are difficult to discern.
- which cases are not covered by the classifier.

When a number of neural nets have been developed and evaluated, new requirements may arise. Future research should explore whether:

- The developers and end-users can successfully employ a Multi Criterion Decision Making technique to choose the classifier(s) that are most suited to the specific application.
- The quality concepts that were used to specify the requirements constitute good decision criteria in such an MCDM-approach.

When developing a neural-net classifier for a clinical application, one may also wish to prune attributes with high economical or ethical costs, or simply remove superfluous attributes. The approach presented in chapter 3 enables the developer/user to *rank* the attributes according to their importance in relation to assigning class labels to cases. Our approach based on the *discriminative power* is useful as, for example, removing attributes using a backward search entails building at least $\frac{1}{2}(n^2+n)$ MLP classifiers, with n the number of attributes. When attributes are selected based on their discriminative performance, one needs to train only n MLP classifiers. Additionally, the method is easy to implement.

- Experiments indicated that ranking attributes based on their discriminative power provides credible results on classification problems with 2 and 3 classes for correlated and independent attributes.

The approach was used also to identify attributes that were important for discriminating specific classes from the others.

In chapter 4, we focused on estimating the performance decrease – the marginal contribution – that results when an attribute or feature is removed from a classifier. This is a measure for how much additional information an attribute provides to the

classifier. Measures and metrics were defined to estimate the performance decrease of a trained neural network. A numerical approach based on a Taylor expansion was used to compute three of the feature metrics. Experiments indicated that two of the metrics provide good estimates of the marginal contribution of a feature. Also a computationally simple method to pruning (removing) input nodes from a trained MLP was introduced. Important conclusions are:

- The metric which resulted in the best estimates of the marginal contribution, the replaceability, gives exactly the performance that is obtained when an attribute is pruned from the network.
- No Taylor expansion is needed to estimate the replaceability of a feature so feature selection based on replaceability is a computationally simple approach.

In some situations, training only *one* MLP that uses all attributes may be sufficient to obtain a (pruned) classifier based on a good subset of features.

- In our experiments pruned networks need not be retrained as their performance were very close to the performance of a minimal error-rate classifier.

The discriminative power is a heuristic to determine quickly unimportant attributes. It is simple to implement and can also be used for networks with more than one hidden layer. Three of the feature metrics developed in chapter 4 use a Taylor expansion that is developed specifically for feed-forward networks with one hidden layer. The expansion is not required to compute replaceability. However, this metric will provide good estimates of the marginal contribution only when multiple linear regression results in good predictions. To obtain better estimates of the replaceability in the general case where the attributes are not normally distributed, other (nonlinear) prediction models are required. It is in most cases computationally complex to build such models.

Verification

The quality concepts and metrics introduced in chapter 2 can, together with the approach to attribute assessment, help to verify a neural-net classifier. It can be checked whether a neural net fulfils the requirements that were specified using the quality concepts/measures. One can simply estimate the values of the relevant quality measures from a test set and compare these with the minimally required quality measures.

The feature metrics can also be used for validation purposes. The potential influence of a feature expresses whether it can be *relevant* for the classification of a case, the replaceability is an indicator of whether the feature is *necessary* for classifying the case correctly. It is a topic for future research to explore whether:

- The feature metrics introduced in chapter 4 can also be used for validation against clinical domain knowledge.

Application

When neural-net classifiers are used in domains with incomplete information, the developer has three possibilities. One is to leave out the cases with missing values. Another is to develop a (possibly large) number of networks that use different subsets of the available attributes. A third possibility is to estimate the missing data.

- The REM-algorithm developed in chapter 5 lends itself to a fast estimation of missing data and the characteristic parameters in samples of incomplete cases.

It acts also as a "filter" that can identify incomplete cases in which too much information is missing. It is a topic for future research to:

- Investigate how much information can be missing before the incomplete case cannot be imputed due to lack of redundancy.

The missing values can only be estimated well when the data contain redundancy. Whether they influence the classification of a case, depends on how much additional information they contain with respect to the other variables. It is possible that (missing) attributes which cannot be predicted well have no influence on the classification of a case, i.e. their replaceability is large. It is also possible that attributes which can be predicted well lead to wrong classifications.

The robustness metric can be used in a practical application of both neural networks as well as other types of classifiers. With robustness one can estimate the possible influence of measurement noise on a particular classification. It was demonstrated how:

- The robustness measure can be used to increase the credibility of a classification by remeasuring an attribute.

The notion of a remeasuring interval was introduced. This is the interval in which remeasuring a particular attribute makes sense.

The general situation where n attributes are contaminated by measurement noise was not investigated. For a neural-net classifier, it is computationally complex to compute the robustness in n dimensions. It is a topic for future research to explore:

- The usefulness of the robustness measure and the two metrics in a situation where one wants to assess the robustness with respect to more than one (noisy) attribute.

Explanation

It was mentioned in chapter 1 that a major obstacle for the widespread use of feed-forward neural networks is the lack of explanation for the classification of a case. Providing thorough explanation to a user is, however, a complex problem. Such an explanation should relate the decision suggested by the classifier to the domain knowledge of the user. Such a neural net can only contain part of the knowledge that was used to construct it. A neural-net classifier is a statistical classifier and its weights capture only stochastic information about the relation between attribute values and class labels. Causal domain knowledge is extrinsic to such a classifier. Future research should aim at:

- Developing an approach for explaining *why* a case obtains a specific class label.

We believe that the methods developed in the chapters 3 and 4 can be used to explain which *inputs play an important role* for a specific classification.

2 Using neural networks in clinical practice

The intention behind this dissertation is to expedite the use of neural-net classifiers in medicine. Despite the large number of medical problems for which neural networks have already been developed (see chapter 1), it is not guaranteed that neural networks will quickly obtain a widespread use in the clinic. Before one can make statements about the clinical applicability of neural networks, it is necessary to consider how physicians work. Herbert Simon has paraphrased the appreciation specialists often have for their work by:

> "The pleasure that the good professional experiences in his work is not simply a pleasure in handling difficult matters; it is a pleasure in using skilfully a well-stocked kit of well-designed tools to handle problems that are comprehensible in their deep structure but unfamiliar in their detail" [4].

A good specialist finds encouragement and satisfaction in his work. Physicians are highly specialized through long training. This again makes them a scarce and expensive resource. Mintzberg says:

> "Given the high cost of the professionals, it makes sense to back them up with as much support as possible, to aid them and have others do whatever routine work can be formalized." [3]

Indeed not all tasks performed by physicians have the same potential of being automated. According to Van Bemmel, in the clinic, the largest potential of computers lies in communication, registration and computation tasks. The higher the abstraction level the more complex the information processing becomes and increasing interaction between the human and the computer is necessary [1]. Tasks such as the establishment of a diagnosis or proposing a therapy requires complex information processing. They are therefore not well suited for automatization by a neural network as it is difficult to validate the classification result. However, the methods and techniques developed in this dissertation may improve such an interaction.

Another factor that hinders high-level information processing has to do with the kind of work physicians do and the organization they participate in. Physicians in a hospital form part of a *professional bureaucracy* [3], a term introduced by Henry Mintzberg. 'Professional' here refers to the high level of education of the staff and the high degree of autonomy of the individual staff members in their work as compared to participants in other types of organizations. The physicians gain *influence* within the rather flat organization by virtue of their knowledge

> "The professional bureaucracy emphasizes authority of a professional nature – the power of expertise" [3].

One can question whether knowledge-based systems that (suggest) clinical decisions will one day be accepted by physicians. It is clear that trying to build automated systems with expertise of their own will interfere directly with the very basis of the physicians' authority and power. They might resist taking advice from a computer program. Henry Mintzberg writes:

> "The professional [physician] resists the rationalization of his skills – their division into simply executed steps – because that [...] destroys his basis of autonomy..." [3].

We expect that computer systems in general and neural-net classifiers in particular have their largest potential in low-level information processing. Tasks that require only little knowledge, routine work or processing of large amounts of data will lend themselves to automatization. Two examples of low-level classification tasks mentioned in chapter 1 are classification of ECG-patterns and segmentation of X-rays. A neural network can, for example, form part of the data processing software in an encephalograph or a CT-image workstation.

The future will show to which extent neural-net classifiers will support medical information processing, be it for the general practitioner, in technical specialities such as radiology, cardiology and clinical chemistry, or the clinical specialities as for example internal medicine.

References

[1] J.H. van Bemmel, J.L. Willems. *Handboek medische informatica*, Bohn, Scheltema & Holkema, 2'nd Ed., 1989.

[2] K. Clarke, R. O'Moore, R. Smeets, J. Talmon, J. Brender, P. McNair, P. Nykänen, J. Grimson, B. Barber. "A methodology for evaluation of knowledge-based systems in medicine", *Artificial intelligence in medicine*, Vol. 6. No. 2, pp. 107-121, 1994.

[3] H. Mintzberg. *Structure in fives. Designing effective organizations*, Prentice Hall, Englewood Cliffs, N.J., 1993.

[4] H.A. Simon. *The new science of management decision*, Prentice Hall, Englewood Cliffs, N.J., 1977.

Summary

1 Scope of the dissertation

This doctoral dissertation presents methods and techniques that may expedite application of neural networks in medicine. Research on neural networks started as a branch of neurology. A neural network consists of a set of interconnected nodes (neurons). Each node works like a junction between "nerve" paths. The neuron receives a number of inputs and produces an activation. The activation of a neuron is functionally dependent on the input signals the neuron receives. Each input signal is modified by a weight. Since their introduction by McCulloch and Pitts in 1943, neural networks migrated to cognitive science, artificial intelligence, statistical regression and decision theory, signal processing and other engineering disciplines. Neural networks have been developed for a large number of applications in economy, computer science, telecommunication and medicine. Chapter 1 contains a brief overview of neural networks that have been developed for clinical decision support.

Clinical application of neural networks is problematic because of their black-box nature. It is very difficult to assess the knowledge encoded in the weights of a trained neural network as it constitutes a nonlinear mapping between the feature (input) space and the class (output) space. In the dissertation, different techniques that characterize the properties of a trained neural network are suggested. Thereby, development and verification/validation of neural networks is expedited. To enhance the application of neural networks, the topic of missing data is also addressed.

2 A neural network performs a mapping

The most general notion of a classifier is a mathematical mapping from an n-dimensional input space to a c-dimensional output space, $N: \mathbb{R}^n \rightarrow \mathbb{R}^c$, where n is the number of features or attributes[1] and c the number of classes to be discriminated. Neural networks can process combinations of qualitative and quantitative data. The c classes can be decisions such as diagnoses or therapies. The mapping is performed by a neural network, more specifically by the weighted connections between the input, hidden and output layers. During training of a neural network, the weights are adapted to minimize a function that measures the difference between the correct output of the learning cases and the output from the neural network.

[1]The terms feature and attribute are used interchangeably.

3 Assessing the output of a classifier

In chapter 2, metrics are defined that characterize the performance of a trained neural network. The performance is measured by letting the neural network classify a set of test cases of which the true class label is known. A contingency table (confusion matrix) is used to characterize the performance of a neural-net classifier. Existing metrics that characterize different properties of the neural-net classifier are discussed. Also some new metrics are introduced. Although these metrics are defined for a neural-net classifier, they can be applied to other classifiers of which the results can be characterized by a contingency table. The metrics include *correctness* – the fraction of correctly classified cases – and *coverage* – the fraction of cases to which the neural network can assign a class label. The misclassified cases are characterized by the metrics for *bias* and *dispersion*. Standard errors and confidence intervals for some of the metrics are specified. The usefulness of the metrics is explored in a set of experiments in which neural networks are trained for classification of thyroid disorders.

4 Assessing the importance of attributes for a classifier

The chapters 3 and 4 address how to assess the contribution of individual attributes to the performance of a neural-net classifier. The motivation for performing attribute assessment is twofold. First, one wants to obtain insight into which attributes are important for assigning a correct class label to one or more cases. Secondly, one needs a criterion to rank the attributes according to their contribution to the performance of the neural-net classifier, before unimportant attributes can be pruned.

In chapter 3, different approaches to attribute selection such as forward, backward and Branch-and-Bound search are discussed. It is argued that backward search is a suitable selection strategy. Based on a mathematical analysis of a minimal error-rate classifier, a metric for the *discriminative power* of an attribute is introduced. This metric is used as a criterion to rank the attributes for each case in the test set. The ranks for each attribute are summed over all cases. The summed ranks are compared using Friedman's two-way analysis of variance. Attributes with a high average rank are unimportant for the neural network whereas attributes with a low average rank have the most influence on the classification performance. The usefulness of this approach is assessed in a number of experiments with artificial classification problems. The experiments indicated that the approach ranks the attributes correctly, when applied on classifiers trained with independent attributes as well as on classifiers trained with dependent attributes. The approach is also used in an application to identify attributes that are important for discriminating four different types of texture in radiographs of focal bone lesions.

In chapter 4, a mathematical framework is developed in which four different feature measures are derived from a minimal error-rate classifier. Each measure allows one to compute a lower bound for the *marginal contribution* of a feature to the performance of a statistical classifier. These measures characterize the *influence* and

replaceability of a feature. Influence is the probability that a feature can possibly change the class label of a case while the other feature values are kept fixed. Replaceability is the expected decrease in performance when a feature value is substituted by the conditional mean of the feature.

Each feature measure is made operational by a feature metric. Computation of three of the four metrics requires the identification of the attribute-conditional decision boundaries. The decision boundaries for a given feature depend on the values of the other $n-1$ features and have to be identified in each case. The boundaries are identified with a piecewise polynomial approximation which is based on a Taylor expansion of the output of a neural-net classifier as a function of the given feature.

A pruning method called *LMS-pruning* is introduced. A feature is LMS-pruned by removing the links that connect the input node of the feature with the hidden nodes and changing the weights that connect the remaining features with the hidden nodes. The weights are modified such that the pruned neural network classifies the training cases identically to a network based on n feature values with the value of the pruned feature replaced by its expected value.

In experiments with artificial classification tasks, the four metrics are compared with respect to their ability to rank the features. These experiments indicate that replaceability is the best ranking criterion. The experiments showed that for neural-net classifiers with a performance close to the minimal error rate, LMS-pruning a feature resulted in a pruned network with a performance that remains close to the maximal (Bayesian) correctness.

5 Estimation of missing data

In chapter 5, a method for iterative estimation of missing data is suggested. Statistical classifiers such as neural networks require all inputs to be able to assign a class label to a case. This impedes application of such classifiers in environments where incomplete data frequently occur. Different approaches to estimate missing data such as the EM-algorithm and Multiple Imputation are discussed. To cope with some drawbacks of these two methods, it is suggested to use an auto associator neural network in recurrent mode to estimate missing values. The properties of an auto associator that is trained with complete cases is analyzed. Subsequently, it is suggested to use the auto associator in recurrent mode to estimate missing values. The conditions that ensure convergence of the recurrent auto associator are derived. It is proven that convergence is only possible when the number of hidden nodes of the auto associator is smaller than or equal to the number of observed values in an incomplete case.

The recurrent auto associator is embedded in the Recurrent Expectation Maximization (REM) algorithm, an iterative approach for estimating missing values in a set of cases. In a set of experiments, the residual variance of predictions made by the recurrent auto associator is compared with the residual variance obtained using multivariate linear regression. Also the REM and EM-algorithms are compared with respect to their ability to estimate missing values (residual variance) and to estimate

the covariance matrix from the incomplete sample. The experiments indicate that the recurrent auto associator results in poorer estimates of the missing values than multivariate regression. The REM-algorithm estimates the covariance matrix slightly worse than the EM-algorithm when the data are fairly correlated and all variables have identical variances. However, the REM-algorithm gives an indication of those combinations of variables with missing and observed values in which the missing data will be predicted poorly. Leaving out such cases leads to an improvement in the estimation of the covariance matrices by the REM-algorithm.

6 Classification from noisy attributes

In chapter 6, the influence of measurement noise on the classification of a case is analyzed. Based on ideas of Brender *et al.*, a quality measure called robustness is specified. The robustness of a classification is the probability that the class label assigned to the case would not be different from the classification based on the (unknown) true attribute values. It is assumed that the measurement noise is Gaussian with a zero mean and uncorrelated with the attributes. A formula for the robustness of a classification is specified.

In practice, it is difficult to estimate the robustness of a classification when the probability density function of the uncontaminated attributes is unknown. Therefore, two approximations are suggested. The bias introduced by these two approximations is analyzed for the special situations where an attribute comes either from a unimodal or a bimodal distribution and is to be classified into one of two classes.

A simulation experiment illustrates how often an attribute has to be remeasured to achieve a robust classification (the measurements are averaged, which reduces the influence of the measurement noise on the attribute value). It is clear that remeasuring a (noisy) attribute makes sense when only a few remeasurements are required to ensure a classification with a sufficiently high robustness. When, however, the robustness of a classification becomes too low, the number of measurements that are necessary to obtain a more accurate estimate of the attribute values becomes very high. The notion of *remeasuring intervals* is introduced. Such intervals indicate when remeasuring an attribute makes sense.

7 General conclusion

The methods and techniques developed in this dissertation are explored in a set of experiments. In chapter 7, it is discussed to which extent these methods and techniques may support development, verification and validation of neural networks. The possibility of applying knowledge-based systems in general and neural networks in particular in the clinic is discussed as well. It is argued that introduction of such systems in clinical practice interferes directly with the work processes of physicians. One can expect that such systems will have their largest potential in low-level information processing. It is an issue for further research to investigate the value of the presented methods and techniques in the development and evaluation of neural networks for clinical application.

Samenvatting

1 Onderwerp van het proefschrift

Dit proefschrift presenteert een aantal methoden en technieken die de bruikbaarheid van neurale netwerken in de kliniek kunnen vergroten. Onderzoek op het gebied van (kunstmatige) neurale netwerken begon binnen de neurologie. Een neuraal netwerk bestaat uit een aantal met elkaar verbonden neuronen (knopen). Elke kunstmatige neuron werkt als een knooppunt van "zenuwbanen". Het neuron ontvangt een aantal ingangssignalen, hetgeen resulteert in een bepaalde activatie van het neuron. Deze activatie is functioneel afhankelijk van de invoer die het neuron ontvangt. Elk ingangssignaal wordt gemodificeerd door een gewicht. Sinds hun introductie door McCulloch en Pitts in 1943 vindt onderzoek naar kunstmatige neurale netwerken plaats binnen de cognitieve wetenschap, kunstmatige intelligentie, statistische regressie en beslissingstheorie, signaalverwerking en andere technische wetenschappen. Neurale netwerken zijn ontwikkeld voor een groot aantal toepassingen in de economie, informatica, telecommunicatie en geneeskunde. Hoofdstuk 1 bevat een kort overzicht van neurale netwerken die zijn ontwikkeld voor klinische beslissingsondersteuning.

De klinische toepassing van neurale netwerken is problematisch vanwege hun 'black-box' karakter. Het is zeer moeilijk vast te stellen welke kennis gecodeerd is in de gewichten van een getraind neuraal netwerk, omdat het netwerk een nietlineaire afbeelding vormt tussen de invoer (kenmerk) ruimte en de uitvoer (klassen) ruimte. In dit proefschrift wordt een aantal technieken gepresenteerd die de eigenschappen van een getraind neuraal netwerk kunnen karakteriseren. Hiermee wordt de ontwikkeling, verificatie en validatie van neurale netwerken ondersteund. Om de toepasbaarheid van neurale netwerken in de medische praktijk te vergroten wordt tevens een methode geïntroduceerd voor het schatten van ontbrekende data.

2 Een neuraal netwerk verricht een afbeelding

De meest algemene beschrijving van een classificator is een wiskundige afbeelding van een n-dimensionale kenmerkruimte naar een c-dimensionale klassenruimte, N: $\mathbb{R}^n \rightarrow \mathbb{R}^c$. Neurale netwerken kunnen combinaties van kwalitatieve en kwantitatieve kenmerken verwerken. De c klassen kunnen bijvoorbeeld diagnoses of therapieën zijn. De afbeelding wordt verricht door het neuraal netwerk, meer specifiek door de gewogen connecties tussen de invoer-, de verborgen- (hidden) en de uitvoerlaag. De gewichten worden in het leerproces zodanig aangepast dat het verschil tussen de geproduceerde en de gewenste uitvoer voor een aantal leercasus minimaal is.

3 Beoordeling van de prestaties van een classificator

In hoofdstuk 2 worden metrieken gedefinieerd die de prestaties van een getraind neuraal netwerk karakteriseren. De prestaties worden gemeten door testcasus, waarvan het juiste klassenlabel bekend is, aan te bieden aan een neuraal netwerk. De prestaties van een netwerk worden in een kruistabel weergegeven. Bestaande metrieken die de waarden in een kruistabel omzetten in kengetallen worden bediscussieerd en nieuwe metrieken worden geïntroduceerd. Alle metrieken zijn gedefinieerd voor een neuraal netwerk, maar ze zijn ook geschikt om de prestaties van andere soorten classificatoren te meten, indien de resultaten hiervan kunnen worden weergegeven in een kruistabel. Besproken worden onder andere de metrieken voor *correctheid* – de fractie correct geclassificeerde casus – en *dekking* (coverage) – de fractie van casus waaraan de classificator een klassenlabel kan toekennen. De incorrect geclassificeerde casus worden gekarakteriseerd met de metrieken *bias* en *dispersie*. Formules voor standaard fouten en betrouwbaarheidsintervallen worden voor een aantal van de metrieken gegeven. De toepasbaarheid van de metrieken is in een aantal experimenten onderzocht. In deze experimenten zijn neurale netwerken getraind om patiënten met schildklierafwijkingen te classificeren op basis van de concentraties van een aantal hormonen in het bloed.

4 Beoordeling van de kenmerken van een classificator

In de hoofdstukken 3 en 4 worden twee methoden beschreven die de bijdrage meten van individuele kenmerken aan de prestaties van een neuraal netwerk. Het doel van kenmerkanalyse is tweeledig. Ten eerste wil men graag weten welke kenmerken belangrijk zijn voor het toekennen van het correcte klassenlabel aan casus. Ten tweede is een criterium nodig om de kenmerken te kunnen rangschikken op basis van hun bijdrage aan de prestatie van het neuraal netwerk voordat onbelangrijke kenmerken die weinig bijdragen verwijderd kunnen worden.

In hoofdstuk 3 worden verschillende zoekstrategieën bediscussieerd, zoals forward en backward search en het Branch-and-Bound algoritme. Hieruit volgt dat backward search een geschikte zoekstrategie is. Op basis van een wiskundige analyse van een minimale-fout classificator wordt een metriek voor het discriminerend vermogen van een kenmerk gedefinieerd. Deze metriek dient als criterium om de n kenmerken te sorteren voor elke casus afzonderlijk. De rangordes worden per kenmerk opgeteld over alle casus. De n sommen worden vervolgens met elkaar vergeleken met behulp van Friedman's tweezijdige variantieanalyse. Kenmerken met een hoge gemiddelde rang zijn onbelangrijk; kenmerken met een lage gemiddelde rang hebben een hoge bijdrage aan het discriminerend vermogen van de classificator. De bruikbaarheid van de voorgestelde methode wordt getoetst in een aantal experimenten met kunstmatige classificatieproblemen. De experimenten laten zien dat de methode de n kenmerken op de juiste manier ordent, ongeacht of deze binnen de klassen afhankelijk zijn of niet. De bruikbaarheid van deze methode wordt gedemonstreerd aan de hand van een toepassing waarbij vier soorten textuur moeten worden onderscheiden in röntgenbeelden van botten die mogelijk een tumor bevatten.

In hoofdstuk 4 wordt een mathematisch kader geschetst waarbinnen vier verschillende maten voor de bijdrage van een kenmerk zijn afgeleid voor een minimale-fout classificator. Elke maat maakt het mogelijk een ondergrens te berekenen voor de *marginale bijdrage* van een kenmerk aan de prestaties van een statistische classificator. Deze maten karakteriseren samen de *invloed* en *vervangbaarheid* van een kenmerk. Invloed geeft de waarschijnlijkheid aan dat een waarde van een kenmerk kan bepalen welke klassenlabel wordt toegekend aan een casus casus gegeven de waarden van de overige kenmerken die constant worden gehouden. De vervangbaarheid geeft de te verwachte daling in correctheid aan, wanneer de kenmerkwaarden worden vervangen door hun conditionele gemiddelde.

Elke kenmerkmaat wordt geoperationaliseerd door een kenmerkmetriek. Om deze metrieken te kunnen berekenen moet men de kenmerk-conditionele klassengrenzen bepalen. De klassengrenzen voor een bepaald kenmerk zijn afhankelijk van de waarden van de $n-1$ andere kenmerken en daarom moeten de grenzen voor elke casus afzonderlijk bepaald worden. De grenzen worden bepaald met een polynomische benadering van het uitvoer van het neuraal netwerk als functie van het gekozen kenmerk.

Een methode om invoerneuronen weg te snoeien – *LMS-pruning* – wordt geïntroduceerd. LMS-pruning van een kenmerk bestaat uit het verwijderen van de connecties tussen het invoerneuron overeenstemmend met het te verwijderen kenmerk en de hidden neuronen, en het modificeren van de gewichten tussen de resterende kenmerken en de hidden neuronen. Deze gewichten worden zodanig aangepast dat het gesnoeide netwerk de leercasus classificeert als een netwerk gebaseerd op alle n kenmerkwaarden waarbij de gesnoeide kenmerkwaarde is vervangen door zijn conditionele gemiddelde.

De ordening van de kenmerken op basis van de vier kenmerkmaten wordt vergeleken met de correcte rangschikking van elk kenmerk in experimenten met kunstmatige classificatietaken. Deze experimenten tonen aan dat de vervangbaarheid het beste ordeningscriterium is. De experimenten laten tevens zien dat voor neurale netwerken die bijna zo goed presteren als de minimale-fout classificator, LMS-pruning van een kenmerk een gesnoeid netwerk oplevert met een prestatie die dicht bij de maximale (Bayesiaanse) correctheid komt.

5 Het schatten van ontbrekende data

In hoofdstuk 5 wordt een methode voorgesteld voor het iteratief schatten van ontbrekende data. Voor statistische classificatoren, zoals neurale netwerken, zijn alle invoergegevens nodig om een klassenlabel te kunnen toekennen aan een casus. Dit maakt het moeilijk deze classificatoren toe te passen in situaties waarin vaak gegevens ontbreken. Verschillende methoden voor het schatten van ontbrekende gegevens, zoals het EM-algoritme en Multiple Imputatie worden bediscussieerd. Om een aantal nadelen van deze methoden te vermijden wordt voorgesteld een zelf-associerend neuraal netwerk in recurrent mode te gebruiken en daarmee de ontbrekende gegevens te schatten. Eerst wordt de situatie geanalyseerd waarin een zelf-associerend netwerk getraind is met een set van complete casus. De condities

waaronder het recurrent zelf-associerende netwerk convergeert worden afgeleid. Uit het bewijs blijkt dat convergentie alleen plaats kan vinden wanneer het aantal hidden neuronen van de auto associator kleiner of gelijk is aan het aantal geobserveerde waarden in een incomplete casus.

De auto associator in recurrent mode maakt deel uit van de Recurrent Expectation Maximization (REM) algoritme, hetgeen een iteratieve methode is voor het schatten van ontbrekende data in een database. In een aantal experimenten wordt de residuele variantie behaald met het zelf-associerend neuraal netwerk in recurrent mode vergeleken met de residuele variantie die behaald wordt met multiple regressie. De REM en EM-algoritmen worden vergeleken voor wat betreft het schatten van ontbrekende data en het schatten van de covariantiematrix van de incomplete database. Volgens de experimenten levert de auto associator in recurrent mode slechtere schattingen op van de ontbrekende data dan multiple regressie. Het REM-algoritme resulteert in iets slechtere schattingen van de covariantiematrix dan het EM-algoritme, indien de data redelijk gecorreleerd zijn en alle variabelen dezelfde varianties hebben. Echter, het REM-algoritme indiceert welke combinaties van variabelen met ontbrekende en geobserveerde data slechte schattingen van de (onbekende) ontbrekende data opleveren. Wanneer deze casus weggelaten worden, ontstaan nauwkeuriger schattingen van de covariantiematrix door het REM-algoritme.

6 Classificaties gebaseerd op met ruis vervuilde kenmerkwaarden

In hoofdstuk 6 wordt de invloed van meetruis op de classificatie van een casus geanalyseerd. In het hoofdstuk wordt de kwaliteitsmaat *robuustheid* voorgesteld die gebaseerd is op ideeën van Brender *el al.* De robuustheid van een classificatie is de waarschijnlijkheid dat de aan een casus toegekende klassenlabel niet zou veranderen indien men de classificatie had gebaseerd op de (onbekende) ruisvrije kenmerkwaarden. Een vooronderstelling hierbij is dat de meetruis Gaussisch is met een gemiddelde 0 en die ongecorreleerd is met de echte kenmerkwaarden. Een formule voor de robuustheid van een classificatie wordt gegeven.

In de praktijk blijkt dat het moeilijk is de robuustheid van een classificatie te schatten wanneer de dichtheidsfunctie van de ruisvrije kenmerkwaarden onbekend is. Daarom worden twee benaderingen voorgesteld waarbij men deze functie niet hoeft te kennen. Deze twee benaderingen veroorzaken een systematische fout in de schattingen van de robuustheid. De grootte van de geïntroduceerde fout wordt geanalyseerd in twee speciale situaties waarin een kenmerk unimodaal of bimodaal Gaussisch verdeeld is en de casus geclassificeerd zijn in een van twee klassen. Het blijkt dat een van de benaderingen een kleinere relatieve fout in de robuustheid geeft. De tweede benadering geeft grotere fouten, maar is rekentechnisch eenvoudiger.

Een simulatie-experiment laat zien hoe vaak een kenmerk opnieuw moet worden gemeten om een robuuste classificatie te garanderen. Het opnieuw meten van een

met ruis vervuild kenmerk is alléén zinvol wanneer weinig extra metingen vereist zijn om een robuuste classificatie te waarborgen. Indien de robuustheid van een casus zodanig laag is dat een kenmerk vaak opnieuw gemeten moet worden om een robuuste classificatie te verkrijgen, kan men beter afzien van het opnieuw meten van dit kenmerk. Het begrip *remeasuring interval* wordt geïntroduceerd. Deze intervallen geven aan wanneer het opnieuw meten van een kenmerk zin heeft.

7 Algemene conclusies

De methoden en technieken die worden voorgesteld in dit proefschrift zijn exploratief onderzocht in een aantal experimenten. In hoofdstuk 7 wordt bediscussieerd in welke mate deze methoden de ontwikkeling, verificatie en validatie van (toepassingsgerichte) neurale netwerken ondersteunen. De klinische toepasbaarheid van kennissystemen in het algemeen en neurale netwerken in het bijzonder wordt hierbij ook kort besproken. Het blijkt dat de introductie van deze systemen in de kliniek direct interfereert met de werkprocessen van de arts. Men verwacht dat deze systemen hun grootste bijdrage kunnen leveren bij het verwerken van grote hoeveelheden gegevens zonder dat daarbij hogere orde interpretaties nodig zijn. In vervolgonderzoek dient de waarde van de voorgestelde methoden en technieken voor ontwikkeling en evaluatie van neurale netwerken voor medische toepassingen vastgesteld te worden.

Zusammenfassung

1 Ziel dieser Arbeit

Diese Dissertation präsentiert Methoden und Techniken, die die Anwendung neuronaler Netze in der Medizin zu beschleunigen (helfen) können. Die Forschung über neuronale Netze begann als Zweig der Neurologie, da ein neuronales Netz typischerweise aus einer Menge miteinander verbundener Neuronen besteht. Die Neuronen, die als Verbindungs- oder Knotenpunkte zwischen Nervenfasern arbeiten, antworten auf eingehende Signale durch ein Ausgabesignal, eine Aktivierung. Diese Aktivierung steht in einem funktionalen Zusammenhang zu den Eingabesignalen, die zumeist durch Gewichte modifiziert werden. Seit der Einführung neuronaler Netze durch McCulloch und Pitts im Jahr 1943 erfuhr diese wissenschaftliche Feld eine Erweiterung hin zu den Kognitionswissenschaften, der künstlichen Intelligenz, der Statistik und Entscheidungstheorie, der Signaltheorie und den Ingenieurwissenschaften. Eine Vielzahl von Anwendungen neuronaler Netze in den Wirtschaftswissenschaften, der Informatik, der Telekommunikation und der Medizin belegt diese Entwicklung. Kapitel 1 enthält einen kurzen Überblick über Anwendungen neuronaler Netze, die für die Entscheidungsunterstützung in der Medizin entwickelt wurden.

Klinischer Anwendungen neuronaler Netze sind nicht unproblematisch aufgrund ihrer "black-box"-Eigenschaft. Es ist – zur Zeit – verhältnismäßig schwierig, das in den Gewichten des Netzes codierte Wissen eines trainierten neuronalen Netzes greifbar zu machen, da ein solches Netz eine nichtlineare Abbildung zwischen dem als Eingabe bereitgestellten Merkmalsraum und den als Ausgabe erwarteten Klassen realisiert. In dieser Arbeit werden deshalb Techniken vorgestellt, die die Eigenschaften eines trainierten neuronalen Netzes charakterisieren. Damit kann die Entwicklung neuronaler Netze beschleunigt und ihr Verhalten verifiziert beziehungsweise validiert werden. Die Diskussion des "missing value" Problems schließt die Arbeit ab.

2 Ein neuronales Netz realisiert eine Abbildung

Die allgemein verwandte Beschreibung für einen Klassifikator ist die mathematische Abbildung eines n-dimensionalen Eingaberaumes auf einen c-dimensionalen Ausgaberaum, N: $\mathbb{R}^n \rightarrow \mathbb{R}^c$, wobei n die Anzahl der Merkmale und c die Anzahl der zu unterscheidenden Klassen bezeichnet. Neuronale Netze können prinzipiell Kombinationen qualitativer und quantitativer Merkmale verarbeiten. Die c Ausgabeklassen können Entscheidungen, z.B. Therapien oder Diagnosen repräsentieren. Die Abbildung wird durch das neuronale Netz, genauer gesagt durch die gewichteten Verbindungen zwischen der Eingabe-, der verdeckten und der Ausgabeschicht realisiert. Dabei findet die Einstellung der Gewichte durch ein Training mit Hilfe eines Lerndatensatzes statt.

3 Beurteilung der Ausgabe eines Klassifikators

In Kapitel 2 werden Metriken definiert, mit denen die Leistungsfähigkeit eines trainierten neuronalen Netzes beschrieben werden kann. Die Klassifikationsleistung wird gemessen, indem das Netz eine Menge von Testfällen – den Testdatensatz – klassifizieren muß. Hierbei ist für jeden Fall die korrekte Klassenzugehörigkeit bekannt. Die Ergebnisse werden sodann in einer Kontingenztafel dargestellt, daß sie die Grundlage für die Anwendung bestehender Maße zur Leistungscharakterisierung neuronaler Netze sind. Diese Maße werden diskutiert und erweitert. Obwohl sie für neuronale Netze entwickelt wurden, sind sie nicht auf diese beschränkt, sondern können zur Charakterisierung jedes Klassifikators herangezogen werden, dessen Ergebnisse durch einen Kontingenztafel beschrieben werden können. Die Maße umfassen die Korrektheit (correctness) – die Menge aller korrekt klassifizierten Fälle – und die Bedeckung (coverage) – die Menge aller Fälle, zu denen das neuronale Netz eine Klassenzugehörigkeit vergeben kann. Die fehlklassifizierten Fälle werden durch die Maße Verschiebung (bias) und Streuung (dispersion) charakterisiert. Für einige Maße ist der Standardfehler und das Konfidenzintervall beschrieben. An Hand von Labordaten aus einer Studie über Schilddrüsenerkrankungen wird die praktische Anwendung und Aussagekraft der Maße illustriert.

4 Messung der Bedeutung von Merkmalen

Die Kapitel 3 und 4 der Arbeit beschäftigen sich mit der Messung der Bedeutung einzelner Merkmale des Eingaberaumes für die Gesamtklassifikationsleistung des neuronalen Netzes. Hierbei werden zwei Ziele verfolgt. Einerseits soll ein Einblick gewonnen werden, welche Merkmale wichtig sind für die Vergabe des richtigen Klassenlabels bezogen auf eine oder mehrere Klassen. Andererseits soll eine Abstufung der Merkmale hinsichtlich ihres Gesamtbeitrages realisiert werden, um unwichtige Merkmale zu eliminieren, sei es um sie durch bessere zu ersetzten oder um die Netzstruktur zu reduzieren.

In Kapitel 3 werden verschieden Suchtechniken zur Merkmalsauswahl diskutiert. Die Rückwärts-Suche erweist sich dabei als geeignet. Basierend auf der mathematischen Analyse des Bayes-Klassifikators wird eine Metrik der Diskriminationsleistung eines Attributs eingeführt. Diese Metrik wird verwendet, um die Merkmale des Eingaberaumes an Hand eines Testdatensatzes anzuordnen. Eine Summierung dieser Werte für alle Merkmale über alle Fälle des Testdatensatzes wird verglichen mit Friedman's zweiseitiger Varianzanalyse. Merkmale mit hoher mittlerem Rang erweisen sich als unwichtig für das Netz, während solche mit niedrigem mittleren Rang sich als äußerst "einflußreich" auf die Klassifikationsleistung des Netzes erweisen. Die Nützlichkeit dieses Ansatzes wird anhand verschiedener künstlicher Klassifikationsprobleme erläutert. Dabei wird deutlich, daß der Ansatz die Merkmale korrekt anordnet, unabhängig davon ob die Merkmale klassenbezogene Abhängigkeiten aufweisen oder nicht. Eine Anwendung auf Röntgenbilder fokaler Knochenläsionen zur Differenzierung von vier Texturen wird als Abschluß des Kapitels vorgestellt.

Im vierten Kapitel wird ein mathematischer Rahmen für vier Maße zur Charakterisierung der Merkmalseigenschaften für einen Bayes-Klassifikators festgelegt. Jede Metrik erlaubt die Berechnung einer unteren Schranke für den zusätzlichen Beitrag eines Merkmals auf die Leistung eines statistischen Klassifikators. Die Maße charakterisieren den Einfluß (*influence*) und die Austauschbarkeit (*replaceability*) eines Merkmals. Der Einfluß ist die Wahrscheinlichkeit, mit der ein Merkmal die Zuordnung eines Klassenlabels verändern kann, wenn die Werte aller anderen Merkmale eingefroren werden. Die Austauschbarkeit beschreibt den erwarteten Leistungsabfall, wenn ein Merkmal durch seinen Mittelwert ersetzt wird. Jedes dieser Maße wird durch die Definition einer zugeordneten Metrik handhabbar gemacht. Die Berechnung von drei von vier Metriken erfordert die Identifikation merkmalsbezogener Entscheidungsgrenzen. Die Entscheidungsgrenzen eines gegeben Merkmals hängen von den Werten der $n-1$ Merkmale ab und müssen für jeden Fall gesondert ermittelt werden. Diese Identifikation wird durch eine polynomiale Approximation realisiert, die auf der Taylor-Reihenentwicklung der Ausgabe des neuronalen Netzes als Funktion eines gegeben Merkmals basiert.

Anschließend wird eine Reduktionsprozedur, die als LMS-pruning bezeichnet wird, vorgestellt. Bezogen auf ein Merkmal erfolgt die Reduktion durch Entfernen aller Links zwischen dem für dieses Merkmal zuständigen Eingabeneuron und der verdeckten Schicht und dem Modifizieren aller verblieben Links zwischen Eingabeschicht und Hidden Layer so, daß die Klassifikationsleistung mit $n-1$ Merkmalen identisch ist zu der Variante mit n Merkmalen.

In Versuchen mit künstlichen Klassifikationsaufgaben werden die vier Maße hinsichtlich ihre Leistungsfähigkeit zur Anordnung der Merkmale nach Wichtigkeit untersucht und verglichen. Diese Versuche zeigen, daß die Austauschbarkeit (replaceability) das beste Anordnungsmaß darstellt. Die Experimente zeigten weiterhin, daß ein neuronales Netz mit einer Leistung dicht an der eines Bayes-Klassifikators durch Anwendung des LMS-prunings mit seiner Leistung dicht an der maximalen Korrektheit bleibt.

5 Abschätzung fehlender Daten

In Kapitel 5 wird eine Methode zur iterativen Abschätzung fehlender Daten vorgestellt. Statistische Klassifikatoren so wie neuronale Netze benötigen einen vollständigen Eingabevektor, um ein Klassenlabel vergeben zu können. Dies verhindert die Anwendung solcher Klassifikatoren in Gebieten, wo unvollständige Daten häufig auftreten. Verschieden Ansätze zur Abschätzung solcher fehlenden Daten wie der EM-Algorithmus oder die "multiple imputation" werden diskutiert. Um mit einigen Unzulänglichkeiten dieser Verfahren umgehen zu können wird ein wiederkehrendes auto-assoziatives Verfahren zur Abschätzung der fehlenden Daten vorgeschlagen. Zunächst wird die Situation analysiert, in der das auto-assoziative Verfahren mit vollständigen Trainingsdaten trainiert wird. Danach wird eine Abschätzung der fehlenden Daten mit wiederkehrendem Modus des auto-assoziativen Verfahrens versucht (recurrent auto associator). Danach werden Konvergenzbedingungen untersucht und es wird gezeigt, daß eine Konvergenz nur dann möglich ist, wenn die

Anzahl der verdeckten Neuronen des Auto-Assoziators kleiner oder gleich der Anzahl der beobachteten Werte eines unvollständigen Falls ist.

Der "recurrent auto associator" ist eingebettet in den "recurrent expectation maximization" (REM) Algorithmus, ein iterativer Ansatz zur Abschätzung fehlender Daten in einer Menge von Fällen. In einem Satz von Experimenten wird die residuale Varianz der Vorhersage durch den Auto-Assoziator verglichen mit der residualen Varianz einer multiplen Regression. Sowohl der REM- als auch der EM-Algorithmus werden im Hinblick auf ihre Fähigkeit zur Abschätzung der fehlenden Daten und der Kovarianzmatrix aus dem unvollständigen Datensatz untersucht und verglichen. Die Experimente zeigen, daß der REM-Algorithmus eine schlechtere Abschätzung der fehlenden Daten liefert als die multivariate Regression. Ferne schätzt der REM-Algorithmus die Kovarianzmatrix geringfügig schlechter ab als der EM-Algorithmus, wenn die gegeben Daten korreliert sind und alle Variablen gleiche Varianz haben. Durch den REM-Algorithmus kann aber die Aussage getroffen werden, in welchen Kombinationen von fehlenden und vorhandenen Merkmalen eine Abschätzung der fehlenden Daten schlecht möglich ist. Diese Kombinationen können dann gezielt unterdrückt werden, was zu einer Verbesserung der Abschätzung der Kovarianzmatrix führt.

6 Klassifikation mit verrauschten Merkmalen

In Kapitel sechs wird der Einfluß von meßtechnisch bedingtem Rauschen auf die Klassifikation eines Falls untersucht. Ausgehend von Ideen von *Brender et. al* wird das Qualitätsmaß Robustheit (robustness) spezifiziert. Die Robustheit einer Klassifikation ist die Wahrscheinlichkeit, daß das einem Fall zugewiesene Klassenlabel sich nicht verändert, wenn die Klassifikation auf den wahren (aber unbekannten) Werten durchgeführt wird. Dabei wird vorausgesetzt, daß das Meßwertrauschen als mit den Merkmalen unkorreliertes Gaußrauschen mit Mittelwert 0 angenommen werden kann. Auf dieser Basis wird eine Formel zur Berechnung der Robustheit einer Klassifikation angegeben.

In der Praxis ist es schwierig, die Robustheit einer Klassifikation zu schätzen, wenn die Verteilungsdichtefunktion der unverrauschten Merkmale unbekannt ist. Es werden deshalb zwei Näherungen vorgeschlagen. Der durch diese beiden Näherungen bewirkte systematische Fehler wird für die Spezialfälle untersucht, daß ein Merkmal einer unimodalen oder bimodalen Verteilung folgt und in ein oder zwei Klassen eingeordnet wird.

Eine Simulation illustriert, wie oft ein Merkmal gemessen werden muß, bis eine robuste Klassifikation vorliegt. Dabei wird deutlich, daß eine Wiederholung der Messung eines Merkmals nur Sinn macht, wenn wenige erneute Messungen zu einer hinreichend robusten Klassifikation führen. Umgekehrt wird deutlich, daß bei zu niedriger Robustheit die Anzahl der notwendigen Messungen sehr hoch wird, um eine hinreichend verläßliche Klassifikation zu erzielen. Es wird deshalb der Begriff der Messwiederholungsintervalle eingeführt. Diese Intervalle kennzeichnen, ob die erneute Messung eines Merkmals Sinn macht oder nicht.

7 Abschlußbemerkung

Die in dieser Dissertation entwickelten Methoden und Techniken werden an Hand verschiedener Experimente untersucht. In Kapitel 7 wird diskutiert, in welchem Umfang sie bei Entwicklung, Verifikation und Validierung neuronaler Netze helfen können. Die Möglichkeit, generell wissensbasierte Systeme und speziell neuronale Netze in der Medizin anzuwenden, wird ebenfalls angesprochen. Dabei wird deutlich, daß die Einführung solcher Systeme direkt mit dem Arbeitsbereich klinisch tätiger Ärzte in Konkurrenz treten kann. Das größte Potential ist deshalb in der "low-level" Informationsverarbeitung zu sehen. Es ist Gegenstand weiterer Forschung, den Wert der hier vorgestellten Methoden für die Entwicklung und Evaluation neuronaler Netze in medizinischen Anwendungen zu untersuchen.

Resume

1 Afhandlingens emneområde

I denne afhandling præsenteres en række metoder og teknikker der vil kunne fremme klinisk anvendelse af neurale netværk. Kunstige neurale netværk blev oprindeligt introduceret indenfor neurologi. Et kunstigt neuralt netværk består af et antal med hinanden forbundne neuroner. Hver neuron udgør et knudepunkt for et antal nervebaner. Gennem hver nervebane sendes et inputsignal til neuronen, hvilket resulterer i et outputsignal, der er funktionelt afhængigt af de modtagne input. Hvert input modificeres af en specifik vægt, et reelt tal. Siden neurale netværk blev introduceret i 1943, har de spredt sig til andre discipliner såsom kognitiv videnskab, kunstig intelligens, statistisk regressions- og beslutningsteori, signalbehandling samt et antal ingeniørdiscipliner. Neurale netværk er blevet udviklet til at løse bestemte opgaver indenfor økonomi, datalogi, telekommunikation og medicin, for bare at nævne nogle områder. I kapitel 1 gives et kort overblik over medicinske anvendelse af neurale netværk.

Det er problematisk at anvende neurale netværk til at løse opgaver indenfor det medicinske område på grund af deres black-box problem, som består i, at kvaliteten af den gennem træning af netværket indlærte viden meget vanskeligt lader sig bedømme. Netværkets forbindelser etablerer en ikkelineær mapning fra et featurerum (input) til et klasserum (output), der parametriseres af netværkets vægte. I afhandlingen udvikles en række teknikker, med det formål at beskrive egenskaber ved et trænet neuralt netværk. Derved understøtter disse teknikker iterativ udvikling, verifikation og validering af neurale netværk. Med det formål yderligere at fremme anvendelsen af neurale netværk, bliver der i kapitel 4 udviklet en metode til estimation af manglende data.

2 Et neuralt netværk udfører en afbildning

Den mest generelle specifikation af en klassifikator er en matematisk afbildning fra et n-dimensionals inputrum til et c-dimensionals outputrum, N: $\mathbb{R}^n \rightarrow \mathbb{R}^c$, hvor n udgør antallet af features eller attributter og c antallet af klasser, man ønsker at skelne imellem. En fordel ved neurale netværk er, at deres input kan være både af kvantitativ og af kvalitativ art. De c klasser består typisk af beslutninger som for eksempel diagnoser eller behandlinger. Afbildningen fra features til klasser foretages at det trænede neurale netværk, mere specifikt af de vægtede forbindelser mellem inputlaget, det skjulte lag og outputlaget. Vægtene bliver trænet gennem en læreproces, hvorunder netværket lærer at klassificere flest mulige træningsvektorer korrekt.

3 Kvalitetsvurdering af en klassifikators output

I kapitel 2 defineres en række metriker, som karakteriserer forskellige egenskaber ved et neuralt netværk i relation til dets præsterede output. Dette sker ved at lade et neuralt netværk klassificere en række testvektorer, for hvilke deres sande klassetilhørselsforhold er kendt. Ved at rubricere klassifikationsresultatet i form af kontingenstabel, kan forskellige egenskaber ved et netværk karakteriseres. Først diskuteres egnetheden af et antal eksisterende kvalitetsmetriker til at karakterisere et neuralt netværk. Alle disse metriker baserer sig på information, der er tilstede i en kontingenstabel. Derefter introduceres en række nye metriker. Selvom disse metriker alle er defineret med det formål for øje at beskrive egenskaber ved et trænet neuralt netværk, er metrikerne defineret så generelt, at de ligeledes kan anvendes til at beskrive egenskaber ved andre typer af klassifikatorer, blot disses præstationer kan karakteriseres ved hjælp af en kontingenstabel. Metrikerne inkluderer korrekthed – andelen af korrekt klassificerede vektorer – og dækningsgrad – andelen af vektorer der kan klassificeres af et neuralt netværk. De misklassificerede vektorer karakteriseres af metrikerne *bias* og *spredning*. Såkaldte standard errors og konfidensintervaller er angivet for nogle af metrikerne. Metrikernes anvendelighed undersøges gennem en række eksperimenter, hvor neurale netværk trænes til at klassificere Thyreodeapatienter.

4 Bedømmelse af attributbidrag til en klassifikator

Kapitel 3 og 4 behandler begge bedømmelse af det marginale bidrag, en attribut yder til et neuralt netværks præstation (korrekthed). Sådan en bedømmelse tjener dels det formål at opnå indsigt i hvilke attributter er vigtigere for at kunne klassificere en vektor korrekt, dels det formål at kunne ordne attributterne i henhold til det bidrag, de yder til en neuralt netværks præstation. På basis af denne viden kan man beslutte ikke at benytte attributter med et forsvindende lille bidrag til netværkets præstation.

I kapitel 3 diskuteres indledningsvis forskellige strategier til udvælgelse af attributter såsom forward, backward og Branch-and-Bound søgning. Backward søgning findes velegnet som udvælgelsesstrategi. På basis af en matematisk analyse af en minimal fejlrate (bayesiansk) klassifikator defineres en metrik, der karakteriserer det marginale bidrag, en attribut yder til klassifikatorens præstation. Denne metrik bruges til at rangordne de n attributter for hver vektor i et testsæt i henhold til deres bidrag. For hver attribut summeres over alle vektorer de til denne attribut tilkendte rang. Disse n summer, der er et udtryk for hver enkelt attributs bidrag, sammenlignes med Friedman's tovejs variansanalyse. Attributter med en gennemsnitlig høj rang har et relativt ringe bidrag til det neurale netværks præstation, hvorimod attributter med en gennemsnitlig lav rang bidrager mere til netværkets præstation. Igennem en række eksperimenter med kunstigt generede data undersøges det, hvorvidt den introducerede metode resulterer i en korrekt rangordning af attributterne. Disse eksperimenter indicerer, at metoden resulterer i den korrekte rangordning af attributterne, når den anvendes på neurale netværk, der er trænet til

at løse klassifikationsproblemer med uafhængige såvel som afhængige attributter. Metoden benyttes også i anvendelsesøjemed til at rangordne attributter, der er vigtige for at kunne skelne forskellige typer af textur i røntgenbilleder af blandt andet knogle tumorer.

I kapitel 4 udvikles et matematisk rammeværk, indenfor hvilket fire forskellige mål for det marginale bidrag af en feature bliver udledt for en minimal fejlrate-klassifikator. Hvert at disse featuremål gør det muligt at beregne en absolut undergrænse for det marginale bidrag, en feature yder til præstationen af en statistisk klassifikator. Disse mål karakteriserer den *indflydelse* en feature har samt dens *substituerbarhed*. Indflydelse er i denne kontekst defineret som sandsynligheden for, at en feature er i stand til at ændre klassifikationen af en vektor, mens de observerede værdier af de resterende features holdes konstante. Substituerbarhed af en feature defineres som den forventede reduktion af netværkets korrekthed, der optræder, når denne feature erstattes af sin betingede middelværdi.

Hvert af de fire featuremål operationaliseres ved en featuremetrik. For at kunne beregne tre af disse featuremetriker, kræves information om de grænser, der for en feature skiller de forskellige klasser. Disse betingede klassegrænser for en specifik feature afhænger af de $n-1$ featureværdier og må derfor identificeres for specifikt hver vektor. Til dette formål benyttes en polynomapproksimation, der er baseres på en Taylorrækkeudvikling af et neuralt netværks output som funktion af den givne feature. Den sidste metrik kan beregnes uden kendskab til de betingede klassegrænser.

En metode til beskæring af inputneuroner kaldet *LMS-pruning* bliver introduceret. LMS-pruning består i at fjerne forbindelserne mellem den inputneuron, man vil bortelimenere, og alle neuroner i det skjulte lag samt at ændre alle vægte mellem de resterende inputneuroner og de skjulte neuroner. Vægtene ændres således, at de beskårne netværk klassificerer træningssættet (baseret op $n-1$ features) identisk med et netværk, baseret på alle n features, men hvor værdien af den bortelimenerede inputneuron er erstattet af sin betingede middelværdi.

I en række eksperimenter baserede på kunstige klassifikationsproblemer sammenlignes de fire featuremetriker med hensyn til deres rangordning af de n features. Disse eksperimenter viser, at featuremetriken for substituerbarhed resulterer i den bedste ordning, nemlig den der er tættest på den sande (Bayesianske) rangordning af de n features. Eksperimenterne viste endvidere, at for neurale netværk, hvis korrekthed er næsten optimal i Bayesiansk forstand, resulterede LMS-pruning i beskårne netværk, hvis korrektheder forblev næsten optimale.

5 Estimation af manglende data

I kapitel 5 introduceres en metode til iterativ estimation af manglende data. Statistiske klassifikatorer såsom neurale netværk kan ikke anvendes på inkomplette vektorer, da netværkets output så ikke kan beregnes. Dette hæmmer blandt andet klinisk anvendelse af sådanne klassifikatorer, da man her ofte konfronteres med inkomplette data. Der eksisterer et antal metoder til behandling af datasæt med manglende værdier. To sådanne metoder er EM-algoritmen og Multipel Inputation;

fordele og ulemper ved begge diskuteres i dette kapitel. For at modvirke nogle af deres ulemper, foreslås det at benytte et cyklisk autoassociativt netværk til at estimere de manglende værdier. Først bliver den situation analyseret, hvori dette netværk trænes med komplette vektorer. Det bevises, at konvergens kun kan optræde, når antallet af neuroner i de skjulte lag er mindre eller lig antallet af observerede værdier i en inkomplet vektor.

Det cyklisk autoassociative netværk udgør byggestenen i REM-algoritmen, der er en iterativ metode til estimation af manglende værdier i et datasæt. I et antal eksperimenter sammenlignes den residualvarians, hvormed de manglende værdier forudsiges ved hjælp af det cyklisk autoassociative netværk, med den residualvarians, der opnås når de samme værdier forudsiges ved hjælp af multivariat lineær regression. REM- og EM-algoritmerne sammenlignes ligeledes, både hvad angår residualvarians af de estimerede manglende værdier og hvad angår deres estimationer af kovariansmatricen. Disse eksperimenter viser, at det cyklisk autoassociative netværk resulterer i ringere estimater af de manglende værdier end multivariat regression. REM-algoritmen estimerer kovariansmatricen lidt ringere end EM-algoritmen, når de inkomplette variable udviser en pæn korrelation med de komplette variable. Imidlertid kan REM-algoritmen indicere, hvilke kombinationer af variable med manglende og observerede værdier, der resulterer i estimater af de manglende værdier, som er behæftede med stor residualvarians. Udeladelse af sådanne vektorer, medfører et forbedret estimat af kovariansmatricen.

6 Indflydelse af støj på klassifikationsresultatet

I kapitel 6 analyseres den øgede usikkerhed i henhold til en klassifikation, som følger af, at attributter måles med støj. Baseret på en idé af Brender *et al.* defineres et kvalitetsmål kaldet *robusthed*. Robustheden af en klassifikation af en vektor er sandsynligheden for, at vektoren ville blive klassificeret anderledes, hvis klassifikationen baserede sig på de sande (støjfrie) men ukendte attributværdier. Det antages, at støjen er normalfordelt med en middelværdi på 0 og at støjen er ukorreleret med de sande attributværdier. En formel for robusthed specificeres for en statistisk klassifikator.

I praksis viser robusthed sig svær at estimere, når fordelingsfunktionen for de støjfrie attributter er ukendt. Derfor foreslås to approksimationer til robustheds-målet, der ikke baserer sig på denne fordelingsfunktionen. Disse approksimationer forårsager dog en bias af robusthedsestimatet, som analyseres for de to specialtilfælde, hvor en attribut er unimodal eller bimodal fordelt og man ønsker at skelne mellem to klasser.

Et simulationseksperiment illustrerer, hvor mange gange man er nødt til at måle en attribut (og udregne deres gennemsnit, hvilket reducerer indflydelsen af målestøj) for at opnå en tilstrækkelig robust klassifikation. Selvfølgelig er det kun meningsfuldt at genmåle en attribut, når få målinger er tilstrækkelige til at sikre en robust klassifikation. Hvis robustheden af en klassifikation er for lille, visers simulationerne, at det nødvendige antal genmålinger bliver så uforholdsmæssigt stort, at de med genmålingerne forbundne omkostninger bliver for store. Baseret på robustheds-

konceptet defineres såkaldte *genmålingsintervaller* for en attribut, der indicerer, hvornår genmåling er meningsfuld.

7 Generel konklusion

De i afhandlingen udviklede metoder og teknikker undersøges eksplorativt i eksperimenterne, som refereret i de enkelte kapitler. I kapitel 7 diskuteres det, hvorvidt de kan understøtte udvikling, verifikation og validering af neurale netværk. Klinisk anvendelse af vidensbaserede systemer og mere specifikt af neurale netværk bliver diskuteret. Der argumenteres for, at introduktion af sådanne systemer i klinikken griber direkte ind i lægernes arbejdsprocesser. Baseret herpå samt på det faktum, at man ikke har tilstrækkelig indsigt i deres egenskaber, konkluderes det, at neurale netværk har deres største potentiale indenfor lavniveau databehandling. Det er et emne for yderligere forskning at undersøge, hvorvidt de i denne afhandling præsenterede metoder og teknikker understøtter udvikling og kvalitetsvurdering af neurale netværk for medicinsk anvendelse.

Curriculum Vitae

1967	Born 17.03.67 in Charlottenlund.
1985	Matriculated at the Copenhagen Business School.
1990	Graduated as a master in Computer Science/Business administration from the Copenhagen Business School. Major in Artificial Intelligence and Neural Networks. Master thesis: "*Connectionistic mental models and cognitive architecture*".
1990	Obtained a grant from the Danish Academy of the Technical Sciences (ATV) for a 2½ years industrial Ph.D. study by CRI a/s and Copenhagen Business School.
1991	Spent 7 moths at the Department of Medical Informatics, Maastricht University, The Netherlands.
1993	Migration to the Netherlands. Employed as researcher by the Department of Medical Informatics at Maastricht University.
1995	Employed as PostDoc researcher by the Department of Biophysics at Maastricht University.

Publications

M. Egmont-Petersen. "An approach for generating explanations in neural networks", *Proceedings for the KAVAS Workshop on Knowledge Acquisition, Visualization, and Assessment*, 1990, pp. 131-136, J. Brender, P. McNair (Eds.), Computer Resources International, Copenhagen.

M. Egmont-Petersen. "Mental models as cognitive entities", *Proceedings of the Scandinavian Conference on Artificial Intelligence*, 1991, pp. 205-210, ED. B. Mayoh, IOS Press, Amsterdam.

M. Egmont-Petersen. "Homomorphic transformation from neural networks to rule bases". In: E. Mosekilde (Eds.), *Proceedings of the european simulation multi-conference*, pp. 260-265, 1991.

M. Egmont-Petersen. "Methods and Metrics for operationalization of Quality Measures for Neural Networks", Technical report, *deliverable ML-4.1 in the KAVAS-II (A2019) project*, AIM-programme, 70 pages, 1992, Computer Resources International, Copenhagen.

M. Egmont-Petersen. "Quality of neural classifiers", *DASY-paper no. 11/92*, 1992, Institute of computer and systems sciences, 12 pages. This paper has been presented at an AIMCOM meeting in Brussels 1 December 1992.

M. Egmont-Petersen. "Coverage of neural classifiers", *DASY-paper no. 12/92*, 1992, Institute of computer and systems sciences, 13 pages.

T. Schiøler, W. Grimson, P. Sharpe, M. Egmont-Petersen, G. Momsen, R. O'Moore, P. McNair. "Automatic decision support based on voting by independent decision support systems", *Proceedings of the CCL congress*, Dublin, pp. 58-66, 1992.

F. Vogelsang, E. Pelikan, M. Egmont-Petersen, T. Tolxdorf, K. Bohndorf. "Segmentierung von Röntgenbildern fokaler Knochenläsionen durch neuronale Netzwerke. Optimierung durch Quality Metrics und modifizierte Contribution Analysis", *Proceedings of the workshop on neural networks at the RWTH-Aachen*, H. Hüning, S. Neuhauser, M. Raus, W. Ritschel (Eds.), pp. 201-210, Aachen, 1993.

E. Pelikan, M. Egmont-Petersen, F. Vogelsang, T. Tolxdorff, K. Bohndorf. "Segmentierung von Röntgenbildern durch implizite Texturklassifikation mittels neuronaler Netze - Ansätze zur Optimierung durch Contribution Analysis and Quality Metrics", Workshop - Visualisierung in der Medizin, 10 March 1993, Universität Freiburg.

F. Vogelsang, E. Pelikan, M. Egmont-Petersen, T. Tolxdorf, K. Bohndorf. "Segmentierung von Röntgenbildern fokaler Knochenläsionen durch neuronale Netzwerke. Optimierung durch Quality Metrics und modifizierte Contribution Analysis", *Proceedings of the 15th workshop of the German Society of Pattern Recognition (DAGM)*, S.J. Pöppl (Ed.), pp. 450-459, 1993.

J.L. Talmon, M. Egmont-Petersen. "Inductive learning and neural networks", *Klinische Fysica*, Vol. 1, pp. 17-20, 1993.

J. Brender, J. Talmon, M. Egmont-Petersen, T. Schiøler, P. McNair, "KAVAS's framework for quality assessment of medical knowledge", *Proceedings of Artificial Intelligence in Medicine*, S. Andreassen, R. Engelbrecht, J. Wyatt (Eds.), pp. 421-424, IOS press, Amstersam, 1993.

M. Egmont-Petersen. "Quality measures from machine learning", Technical report, *deliverable ML-4.1a in the KAVAS-II (A2019) project*, AIM-programme, 35 pages, 1994, Maastricht University, Maastricht.

M. Egmont-Petersen, J.L. Talmon, J. Brender, P. NcNair. "On the quality of neural net classifiers", *Artificial Intelligence in Medicine (AIM-journal)*, Vol. 6, No. 5, pp. 359-381, 1994.

E. Pelikan, F. Vogelsang, B. Schulz, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf. "Röntgenbildsegmentierung durch Topologische Karten oder Multilayer Perzeptron - ein Vergleich", *Proceedings of the workshop on digital image processing in medicine (Digitale Bildverarbeitung innerhalb der Medizin)*, Freiburg, 1994.

E. Pelikan, F. Vogelsang, U. Gattermann, M. Egmont-Petersen, T. Tolxdorff, K. Bohndorf. "Implizite Texturklassifikation mittels neuronaler Netze zur Segmentierung fokaler Knochenläsionen in Röntgenübersichtsaufnahmen. In: *Ein integrierender Teil arztunterstützender Technologien*, S.J. Pöppl SJ, H. G. Lipinski, T. Mansky T (Eds.), Medizinische Informatik, Biometrie und Epidemiologie 77, MMV Medizin Verlag, München, pp. 201-206, 1994.

E. Pelikan, F. Vogelsang, B. Schulz, M. Egmont-Petersen, K. Bohndorf, T. Tolxdorf, *Texturbasierte Segmentierung von Röntgenbildern mittels Multylayer-Perzeptron und Topologischer Karte*, Proc. 16 Sypm. DAGM, Wien, pp. 589-600, 1994

M. Egmont-Petersen, J.L. Talmon, E. Pelikan, F. Vogelsang. "Contribution analysis of multi-layer perceptrons. Estimation of the input sources' importance for the classification", pp. 347-358, In: *Proceedings of Pattern Recognition in Practice 94*, Ed. E. Gelsema, L. Kanal, Elsevier, 1994.

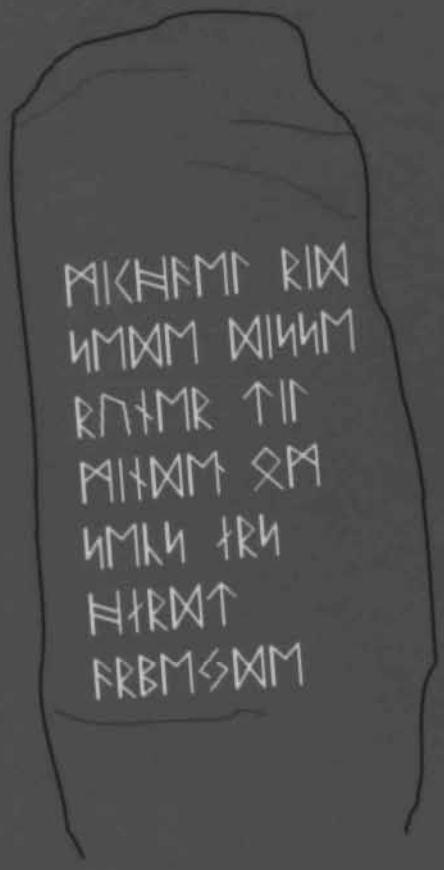
V. Karthaus, H. Thygesen, M. Egmont-Petersen, J. Talmon, J. Brender, P. McNair. "User-requirements driven learning", *Computer Methods and Programs in Biomedicine*, Vol. 48, No. 1-2, pp. 39-44, 1995.

J. Brender, J. Talmon, M. Egmont-Petersen, P. McNair. "Measuring quality of medical knowledge", pp. 69-74, In: *Proceedings of Medical Informatics in Europe 1994*, Ed. P. Barahona, M. Veloso, J. Bryant, Lisbon, 1994.

M. Egmont-Petersen, E. Pelikan. "Erweiterte Kriterien zur Beurteilung von Segmentierungsergebnissen", pp. 24-30, In: *Dititale Bildverarbeitung in der Medizin*, B. Arnolds, H. Müller, D. Saupe, T. Tolxdorf (Eds.), Proceedings 4th workshop on digital image processing in medicine, Freiburg, 1996.

M. Egmont-Petersen, E. Pelikan. A.W. Ambergen. "Towards an explanation facility for feed-forward neural networks", To appear in: Tagesband zur GMDS'96 (Proceedings of the German conference of Medical Informatics), Bonn, 1996.

M. Egmont-Petersen, T. Arts. "Detection of implanted markers in radiographic image sequences", pp. 209-214, In. Bildverarbeitung für die Medizin (Proceedings of the workshop in Aachen on Medical Image Processing), T. Lehmann, I. Scholl, K. Spitzer (Eds.), Aachen, 1996.



ISBN 90-423-0002-7