To appear in: Proceedings of the Conference on Prerational Intelligence—Adaptive and Learning Behavior, Bielefeld, Germany, April 1994

Nonparametric Regression for Learning

Stefan Schaal
Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology
ATR Human Information Processing Research Laboratories

Abstract: In recent years, learning theory has been increasingly influenced by the fact that many learning algorithms have at least in part a comprehensive interpretation in terms of well established statistical theories. Furthermore, with little modification, several statistical methods can be directly cast into learning algorithms. One family of such methods stems from nonparametric regression. This paper compares nonparametric learning with the more widely used parametric counterparts and investigates how these two families differ in their properties and their applicability.

1 Introduction

This paper will investigate learning in a very restricted sense in that it only focuses on a low level of learning, i.e., how to establish a transformation from input data to output data under a given performance criterion. In statistics, such a mapping is called regression if the output variables are continuous, and it is called classification if the outputs are discrete. In learning theory, the acquiring of such input-output mappings is called supervised learning. Due to the focus on these topics, this paper will not address many of the other important questions in learning, for example, where the definitions of the input and output variables come from, whether they should be defined on a microscopic or macroscopic level, how performance criteria can be developed, and how learning modules can be arranged in a competitive and/or hierarchical way. Nevertheless, the input-output mapping problem, formulated as the search for nonlinear correlation, is one of the main issues of the formation of structured processes, and despite the research progress in the recent years, many details have yet not been solved in a satisfying way.

The major difference between statistical data analysis of input-output mappings and the approach taken by learning theory is that statistical analysis tries to incorporate as much domain knowledge as possible to interpret the data, while in learning theory a "black box" approach is preferred. The notion "black box" expresses the desire for autonomy: Is it possible to have a general system that is fed with a stream of input and output data in order to "self-organize" to reflect the nonlinear correlation between inputs and outputs? The various forms of neural network architectures are an attempt to address this question, but so far no positive answer has been found. Every method has serious weaknesses in certain classes of problems, be it the perceptron on non linearly separable tasks (Minsky & Papert, 1969), the ordinary feedforward multi–layer neural network in terms of catastrophic interference, and the Boltzman machine in terms of learning speed. However, the understanding of the appropriateness of an algorithm for a given problem has always significantly advanced at the moment at which statistical theories could be found to apply. The Hopfield net (Hopfield, 1982) and the Boltzman machine (Hinton & Sejnowski, 1983), for instance, have their foundations in statistical mechanics, the Kohonen algorithm can be formulated as a Fokker-Planck equation (Ritter & Schulten, 1988), and multi–layer linear feedforward networks can be analyzed using linear regression methods.

If statistical theories underlie the understanding of many learning algorithms, it is consistent to also try to derive learning algorithms directly from the statistical viewpoint by grounding their development in a statistical framework from the very beginning. As an advantage, probabilistic assumptions of the learning framework are made explicit (e.g., MacKay, 1993, Jordan & Jacobs, 1993). A drawback, on the other hand, is that these algorithms often lose the connection to a desired hardware, for instance, the neurons of a brain, and focus instead on a purely computational theory of learning. However, it might subsequently be possible to reconcile these theories with more biologically motivated processes by trying to find biological structures that approximate the computations of the developed method, as it has often been possible to find statistical methods which explain the behavior of biologically motivated algorithms. Even if this is not possible—or often also not intended—the insight into what is achievable by a certain learning theory will always provide valuable benchmark performance for comparisons and will be useful in the development of artificial systems.

The emphasis of this paper lies on a strand of statistical learning methods from nonparametric regression analysis. One of the key characteristic of these methods (as investigated in the current context) is their relatively large insensitivity to the structural complexity of the input-output function to be learned. The resulting advantages and disadvantages of nonparametric learning in comparison to the more well-known parametric methods will be the topic of this paper. Before starting this comparison, it should be noted that despite the attempt to review a certain amount of related work in the following sections, it is absolutely impossible to cover the vast literature on this topic. Hence, this paper can only attempt to briefly point to some references in the literature which could lead the interested reader to more information.

2 Parametric vs. Nonparametric Learning

In order to compare different learning approaches for input-output mappings, it is useful to base the comparison on properties which are generally accepted to be important for this type of learning. The following list is not meant to be complete, but it should cover a large number of the important issues:

– *Autonomy*: How many parameters of the learning approach need human or heuristic adjustment, like, for instance, learning rates?
– *Bias*: Is the learning algorithm inherently biased towards certain solutions, or, if desired, can bias be incorporated easily?
– *Complexity Control*: Can the learning approach use regularization to avoid overfitting?
– *Consistency*: Does the learning result become increasingly less biased when the number of learning experiences goes to infinity?
– *Continuous Learning*: Can the learning system be trained forever without reaching a limit of its adaptation capacity or degrading its performance?
– *Capacity*: How many parameters are required to represent a given amount of experience?
– *Flexibility:* How well can learning follow a dynamically changing environment?
– *Generalization*: How well does the learning box infer an appropriate output to an unknown input?
– *Incremental Learning*: Is learning possible in an experience by experience way?
– *Interference*: Does previously learned knowledge degrade if new data is incorporated in the learning box?
– *Interpretation:* When opening the "black box", is it possible to interpret the way the correlation between input and output data has been established? In biological modeling as well as in statistical modeling, this is often a desirable property in order to gain a better understanding of the underlying processes.
– *Lookup Speed*: How quickly can a response to a query be formed, i.e., what is the computational complexity of a lookup?
– *Scaling*: How well does the learning approach scale when increasing the dimensionality of the problem?
– *Real-Time Learning*: How quickly can the learning box extract the relevant information from new experiences and incorporate it into its representation?
– *Statistical Measures*: Is it possible to make qualitative and quantitative, usually statistical statements about the quality of what has been learned?

This list of criteria forms the basis of most of the discussions in the following sections. Due to space limitations, however, not all the issues will receive the attention which they actually deserve.

2.1 Global Parametric Learning

The most common approach to learning an input-output mapping has been to fit a *global* parametric function

$$\mathbf{y} = f(\mathbf{x}, \theta) \tag{1}$$

with finite length parameter vector θ to the data $(\mathbf{x}, \mathbf{y})$. The fitting procedure minimizes some error function J such that $\theta^* = \min_\theta\left(\sum J_i\right)$ for all data pairs $(\mathbf{x}_i, \mathbf{y}_i)$ in a training set. Very often, J is a least squares criterion. If f is linear in the parameters θ, least squares minimization corresponds to linear regression and the parameters can be calculated in closed form. In more interesting applications, however, f is nonlinear in the parameters and the error function J must be minimized iteratively.

Parametric learning methods have found wide application in the form of neural networks, and most of the following discussions will relate to this. In neural networks, a weight matrix $\mathbf{W}$ is adjusted with the help of backpropagation or other methods to model the training data (e.g., Hertz, Krogh, & Palmer, 1992); the weights in $\mathbf{W}$ are another way to write the parameter vector θ above. Ripley (1992) gives an extensive discussion of various forms of neural networks from a statistician's viewpoint.

Advantages

Generally, parametric learning has been very successful if the assumed structure of the function $f(\mathbf{x}, \theta)$ is sufficiently close to the function which generated the data to be modeled. However, in contrast to the approach of a statistician, who would spend a large amount of time on comparing different kinds of structures before deciding on a particular one, learning approaches avoid this problem. The hope is that a general structure rich enough to model a large fraction of all possible functions could take care of the model selection issue and automatically adjust to the given problem by means of a learning algorithm. The reason for such hope was based on the work of Kolmogorov (1957) and Sprecher (1965), who proved that a certain kind of network can *exactly* represent any continuous function g from $[0,1]^n \to \Re^m$. This work was adapted to the standard neural network representation by Hecht-Nielson (1989), Cybenko (1989), Funahashi (1989), and others, who, in various steps of improvements of the theories, finally showed that a one-hidden-layer neural network can *approximate* any continuous function.

An appealing advantage of parametric methods lies in the computational cost to perform a lookup, i.e., the generation of a predicted output $\hat{\mathbf{y}}_q$ to a new input $\mathbf{x}_q$; the speed of lookup is proportional to the complexity of the model structure and is usually rather fast. Two pleasant properties of parametric learning are also to be mentioned. The first one is the high degree of data compression which can be accomplished in reasonably chosen networks. Second, there is a fair chance that the internal representations evolved by the parameters can be interpreted to some extent; examples are Hinton's (1986) "family relationship" network, or Zipser and Anderson's (1988) work in which hidden neurons modeled the activation characteristics as observed in neurons of parietal cortex of monkeys. Generally, however, it is hard to interpret the parameters of a complex parametric learning box: It has often been observed that training an initially randomized system on the same data anew leads to quite different values of the parameters for every training run.

Disadvantages

Despite this list of benefits of parametric learning, a variety of disadvantages remain. The proof of universal learning capability for continuous functions does not imply that the model selection problem in parametric learning was solved by sigmoidal neural networks: None of these results makes any statements about the appropriate size of the network for a given problem nor about whether it is possible to learn the weights. In order to achieve acceptable generalization, which is at the heart of each learning system, the number of parameters must not be too large since this would significantly reduce the generalization quality due to overfitting of the data. On the other hand too few parameters will not suffice to approximate the data well enough. The work of Vapnik (1982) and Barron (1994), for example, addresses the issues of bounds on the generalization error in parametric learning and, hence, the selection of an appropriate number of parameters.

A further problem which parametric learning methods have to face is the so-called "catastrophic interference". If learning takes place incrementally, nonuniform sampling of the input space often leads to a strong degradation of the mapping f in regions of the input space where good learning results were achieved before. In global parametric learning, training on any data point causes a change in all the parameters, i.e., it has a global effect. Hence, if a block of training data stems from only a small subregion of the entire input space, the global parametric learning system will tend to model this subregion very well by changing *all* its parameters in favor of modeling this subregion. In doing so, however, the performance degrades for other regions of the inputs space. A variety of methods have been suggested to avoid these properties, for example pre-training the network on more general data before finally incrementally training it on more specialized data, or trying to sharpen the receptive fields of the neurons to make them as local as possible. All these methods, however, are more or less of a heuristic nature and will be successful in some cases and fail in others. The structure of *global* parametric learning algorithms possesses the property of catastrophic interference as an *inherent* property, and it will remain a hard task to overcome this problem in a general and principled way without backing away from global learning methods.

To complete the list of disadvantages of parametric learning, the problems of time-consuming iterative learning procedures as well as sensitivity towards some meta parameter settings has to be addressed. The latter problem concerns parameters like learning rates, momentum terms, decreasing temperatures, regularization terms, etc., which are often tuned manually or by some heuristic processes (e.g., Hertz, Krogh, & Palmer). However, faster computer hardware and better algorithms (e.g., Sutton, 1992; Jacobs, 1988) may overcome these problems in the near future.

Iterative learning methods often suffer from getting captured in local minima, and the fact that they may have extremely slow convergence rates has been reported quite frequently and will not be addressed here. One particularly unfavorable side-effect of the time consuming training, however, is that the assessment of statistical quantities, such as measuring the generalization error by means of crossvalidation methods (Stone, 1974; Wahba & Wold, 1975), becomes computationally expensive. For instance, the calculation of one crossvalidation error requires the retraining of the entire system, and in order to achieve statistically valid results several sets of cross validation must be performed. Thus, cross validation and other statistics (e.g., Cohn, 1993) become almost impossible to compute for complex parametric learning systems. As a final point it should be mentioned that in general many nonlinear global parametric learning systems lack a body of useful statistics to assess the quality of what has been learned by the learning box.

2.2 Local Parametric Learning

In order to avoid some of the problems of global parametric learning, local parametric methods may offer an interesting solution. Instead of finding a complicated parametric function $\mathbf{y} = f(\mathbf{x}, \theta)$ which is capable of modeling all the data in the entire input space, the input space may be divided into many partitions. In each of the partitions, a much simpler parametric model can be sought to fit the data. If the number of partitions is fixed, one obtains a local parametric (or piecewise parametric) learning system, usually preferred to be of an additive form

$$\mathbf{y} = \sum_i f_i(\mathbf{x}, \theta_i) \qquad (2)$$

If the number of the partitions can grow, so can the number of parameters, and the learning method enters the realm of nonparametric learning, which is the topic of the next section. The function f_i need not have identical structure, and quite often they are again the sum or product of other simple functions. Importantly, the f_i are only valid for a subset of the inputs space and, in the case of the summation (2), they are zero elsewhere.

Advantages

The key intent of local parametric learning is to keep the constituent functions f_i as simple as possible in order to remove the need for cumbersome iterative training and to avoid interference of the functions by ensuring that they are only locally valid. Moreover, if the f_i have simple statistical properties, the entire learning system may be characterized by statistical measures.

The purest form of a local parametric learning system is a fixed-grid lookup table with a parametric function inside of each partition. It avoids any kind of interference during incremental learning since new training data only affects a parameter change in the partition it falls into. An overlapping fixed-grid lookup-table like Albus' Cerebellar Model Articulation Controller (CMAC) (Albus, 1975) is very close to this pure form, although parameter changes also affect neighboring partitions. As soon as the partitioning is allowed to change during learning, a global component is added to the local parametric learning system. This is the case for the most traditional members of local parametric learning systems, the spline methods (e.g., de Boor, 1978). Kohnen vector quantizers in which each template vector learns a local linear model of the input-output relation in addition of the input tessellation (Ritter et al., 1992) also have this global component.

While in all the aforementioned approaches the global component results from the propagation of parameter changes over an increasing neighborhood of partitions (like in a diffusion process), there is a class of local parametric learning systems which make explicitly use of a global process, a gating process, to learn the partitioning. A large field of such local parametric models are finite mixture models (e.g., Duda & Hart, 1973, McLachlan & Basford, 1988), in which the data are usually modeled from the viewpoint of maximum likelihood estimation. Recently, mixture models found increased attention for learning in the work of Jacobs et al. (1991), Nowlan (1991), Jordan & Jacobs (1994), Ghahramani & Jordan (1994), Zemel (1993), and several others. The reason for choosing this kind of a model for learning is grounded in the statistical accessibility of the approach. Moreover, mixture models can often be trained by Expectation-Maximization (EM) algorithms, a reliable iterative maximum likelihood estimation technique (Dempster et al., 1977). EM methods avoid the introduction of learning parameters like learning rates and, thus, provide an interesting tool for adding autonomy to learning systems. One other popular approach, radial basis function neural networks (Moody & Darken, 1988; Poggio & Girosi, 1990), can be of a local parametric nature. Such radial basis function networks model the data in terms of

$$\mathbf{y} = \sum_i c_i f_i(\mathbf{x}, \theta_i) \tag{3}$$

where the f_i are functions which decay to zero in all directions from their center (e.g., Gaussians) and are thus local. The c_i are found by fitting the data pairs $(f_i, \mathbf{y})_j$ by linear regression, which can be interpreted as the gating process in this approach.

Disadvantages

The advantages of local parametric learning systems are bought at the expense of either an inflexible, fixed a priori partitioning of the input space, or at the computational effort of finding an appropriate partitioning of the input space, essentially a classification problem. For instance, in some of the aforementioned mixture approaches, most of the "intelligence" of the learning system was deferred to this classifier component. The good news becomes that both regression and classification problems can now be solved by just addressing the problem of classification for the partitioning; learning the function within a partition is by definition simple. The shortcoming, however, is that powerful classification techniques tend to be computationally complex such that the removal of complexity at the level of the individual partition was achieved by introducing new complexity at the gating level. Within this problem, the nastiness of interference re-appears. If the gating component is insensitive towards a nonuniform sampling of the input domain, interference will be avoided. The parametric nature of the gating component, however, makes this property rather unlikely. Interference problems will therefore persist in local parametric models unless the system uses a fixed grid partitioning, which has usually limited learning capabilities. Hence, the term "local" in local parametric learning systems which employ a gating process is slightly misleading.

At last, the question remains of how many partitions should be chosen for a local parametric learning system. This question is similar to the question of how many hidden neurons should be allocated for a neural network. Having too few partitions will result in a large bias of the model, having too many partitions will result in overfitting of the data. Nevertheless, since the training of the learning box is much faster than in global parametric systems, methods like crossvalidation for model selection become feasible, or if the local parametric model has a clear statistical model, statistical tests like likelihood ratios or F-test may be used as well.

2.3 Nonparametric Learning

In the previous sections, the implicit question arose of whether it is possible to not restrict a learning system to a certain number of partitions, hidden neurons, or similar quantities. It would be more reasonable to allocate resources as needed, but still not to overfit the data with the current representation. In so doing, one enters the field of nonparametric learning techniques. There is no formal definition of nonparametric regression. The name "nonparametric" is to indicate that the function to be modeled consists of very large families of distributions which cannot be indexed by a finite-dimensional parameter vector in a natural way (Hájek, 1969). Hence, the nonparametric estimation of the input-output correlation is often called "distribution-free", implying that no assumptions about the statistical structure are made. In Scott (1992) it is stated that in a nonparametric estimate of an output $\hat{\mathbf{y}}_q$ for a given input point $\mathbf{x}_q$, the influence of any other point $\mathbf{x}_i \neq \mathbf{x}_q$ on the estimate of $\hat{\mathbf{y}}_q$ should asymptotically vanish when the estimate is based on a increasing number of data points. This latter definition emphasizes that nonparametric methods should become increasingly local with more experiences which assures the consistency of the learning method, i.e., the decrease of bias. However, the assumption of being local will not be fulfilled by all the methods discussed below.

In the current context, an appropriate working definition will be that a nonparametric learning system can change the number of parameters used by the learning box. This may be achieved by (i) growing the number of parameters of the learning system and/or (ii) re-evaluating the parameters as a function of the inputs. It is useful to separate these two points in the following since they have different characteristics.

2.3.1 Fixed Number of Parameters

The key ingredient of "Fixed number of Parameters Nonparametric Learning" (FPNL) is to re-evaluate a finite set of parameters of the learning system for every lookup point in order to cope with the theoretically infinitely large number of parameters of a nonparametric learning system. This finite parameter set belongs to a simple parametric model which is fitted to the neighboring data of the current query point. For this purpose, all the data encountered during previous learning trials must be stored in a memory, which is the reason that such techniques have often been termed memory-based learning (Stanfill & Waltz, 1986). It also explains that a FPNL system is not trained in the parametric sense: All data is simply stored in the memory, and only when it comes to creating an output for a given input is a computational process needed. Thus, "training" of FPNL systems is extremely fast. For a lookup for a given query point, the contributing neighborhood must be determined and subsequently be processed by a local averaging method. The criteria of how to determine an appropriate size of this neighborhood and which averaging method to use to fit the data in this neighborhood are the major concerns in FPNL.

The most popular functions to do the local averaging have been locally constant functions, locally linear functions, locally quadratic functions, and local splines. There is no limit to what is used as the local function, but the goal is clearly to have as simple a parametric function as possible. Two types of neighborhoods exist, hard and soft neighborhoods. Hard neighborhoods select k-nearest neighbors to be included in the averaging process. Soft neighborhoods weight the data according to the distance from the current query point by means of a smooth function whose maximal value is at the query point and which decays in all input dimensions to zero; (multivariate) Gaussians have often been chosen for this purpose. Soft neighborhood FPNL systems are also called kernel regression or kernel smoothers (Hastie & Tibshirani, 1990; Scott, 1992).

The most well-known member of FPNL systems is the k-nearest neighbor system (Fix & Hodges, 1951; Duda & Hard, 1973; Eubank, 1988), a hard neighborhood method with the simplest of all local functions, the locally constant function. Parzen windows (Parzen, 1962), a technique for density estimation, can be interpreted as a kind of kernel regression. Another popular kernel smoother is the Nadaraya-Watson method (Watson, 1964; Nadaraya, 1964; Härdle, 1991). Hastie and Tibshirani (1990) give an extended bibliography on this topic.

Local polynomial models have been used for smoothing time series in Macauley (1931), Sheppard (1912), Sherriff (1920), and Whittaker and Robinson (1924). Crain and Bhattacharyya (1967), Falconer (1971), and McLain (1974) proposed weighted regression to fit local polynomials to data, i.e., a kernel smoother with a first order or higher order model. Cleveland (1979) analyzed locally weighted running line smoothers and made this method popular in nonparametric statistics. Farmer and Siderowich (1987, 1988a,b) applied local linear models for time series prediction. Cleveland et al. (1988b) extended kernel regression with local linear models to the multivariate case. Atkeson (1992) introduced locally weighted regression (LWR) for learning systems and gives a broad review of related literature. Schaal and Atkeson (1994a,b) extended the LWR approach by several statistical tools and demonstrated its usefulness in high dimensional spaces for learning control.

Finally, there is a large body of literature on applying higher order polynomials to nonparametric regression. Cleveland et al. (1988a) discuss the use of quadratic models for locally weighted regression. Nonparametric spline methods, in particular cubic splines, were analyzed in the work of deBoor (1978), Wahba and Wold (1975), Silverman (1985), and numerous other papers; see Hastie and Tibshirani (1990) and Silverman (1985) for further references.

Advantages

FPNL systems are truly local learning techniques because there is no hierarchically higher process to create something like a partitioning of the input space. They remove any need for pre-defining a global structure of the learning box. The function to be learned is not represented by *piecewise* parametric models but rather by an "infinity" of parametric models—an individual model for every query point—in the same sense as a truncated Taylor series expansion gives a different representation for every basis point about which the expansion is carried out. Since FPNL systems are memory-based and always take all data into account for answering a query, they avoid interference problems.

For an infinitely large number of data points, FPNL systems could become infinitesimally local in order to generate the correct answer to a query: In a deterministic system, the nearest neighbor would then be the best answer. For a finite number of stochastic data, however, some averaging over several data points is necessary. In a hard weighting method, this raises the question of how many data points should be included in the averaging process, or in the soft weighting method, how wide the smoothing function should be. Too large a neighborhood will introduce too much bias in the predictions of the learning system, while too small a neighborhood will have too much variance and therefore bad generalization properties. These problems have been addressed primarily by crossvalidation methods, but many alternative techniques like F-tests, Mallow's Cp-test, the control of the equivalent degrees of freedom (Cleveland, 1979), likelihood ratio tests, and Akaike's information criterion can be applied (e.g., Cleveland et al., 1988b; Hastie & Tibshirani, 1990; Tibshirani & Hastie, 1987). When selecting the most appropriate neighborhood, a particular advantage of FPNL results from its very fast training: methods like leave-one-out crossvalidation can be performed at great ease since the point to be left out is just temporarily deleted from memory and an evaluation of the crossvalidation error is then performed with the reduced memory.

In univariate input spaces, the locality parameter is the only open parameter in the FPNL system. For multivariate input spaces, however, it becomes the distance metric. The distance metric can be conceived of as the matrix necessary to compute the Mahalanobis distance (e.g., Duda & Hard, 1973) of each input point in memory to the current query point. By running the negative distance measure through an exponential function, the weight of the corresponding point in memory is computed. One obvious purpose of the distance metric is to normalize each input dimension such that different units of the input dimensions are scaled appropriately. More interestingly, however, the distance metric is also able to adjust the importance of the individual input dimensions for the current problem. This, indeed, is a method for detecting features in input space. Unimportant input dimensions can be canceled by a zero entry in the appropriate elements of the distance metric matrix, while important dimensions receive a large entry. Methods to calculate the distance metric are usually based on crossvalidation and use gradient decent techniques (e.g., Atkeson, 1992; Lowe, 1993).

Distance metrics can be the same for all query points or they can be a function of the query point. Imagine a mapping from two input dimensions to one output dimension which can be depicted as a mountain landscape where the output dimension is the height of mountain. Assume furthermore that this landscape is a ridge of constant height, but that the slopes of the ridge are sometimes very steep and sometimes rather shallow. The ridge also changes its direction as a function of the input space. If one wants to estimate the height of the landscape from a finite data sample, a *constant* distance metric will not be very useful. For example, sometime the neighbors in the South and North are more important than those in the East and West, and sometimes vice versa. In such cases, a variable distance metric is required, a technique which is related to supersmoothing in statistics literature (e.g., Silverman, 1985; Fan & Gijbels, 1992). Local distance metrics can be calculated based on local versions of crossvalidation and other statistical methods (Schaal & Atkeson, 1994). Finding the locally best distance metric is a much faster process than finding the globally distance metric. However, the local distance metric must be recomputed for each query point while the global distance metric remains the same for all query points.

In sum, the major advantages of FPNL systems is the removal of the necessity to predefine any kind of global structure for the learning box, the very fast training, and due to the truly local models, the avoidance of interference. This results in the very large flexibility of this approach.

Disadvantages

The price which must be paid for the appealing properties of FPNL is a larger memory requirement and often a larger computational effort during lookups. The first computational process to perform a lookup is the selection of the k-nearest neighbors. This can be done by means of k-d trees (Friedman et al., 1977; Omohundro, 1987), an $O(\log n)$ process (n is the number of data in memory). The computational overhead to create the k-d tree does usually not increase the training time of FPNL significantly. For soft neighborhoods, k-d trees can be used, too, in order to determine the data points which are close enough to contribute to the local model at all. Then, the Mahalanobis distance and the weight for each contributing point must be calculated. Finally, the weighted data is submitted to the averaging process. In case of a locally constant model, this proceeds very fast. For higher order models, however, the computational cost increases notably. A linear model, for instance, requires a weighted regression analysis on the included data. A solution to these computational problems is to parallelize the formation of the weights and the regression equations: Each point in memory can be thought of as a receptive field which computes its contribution to the averaging process. The results of these receptive fields are summed up and the remaining computations are normally not very significant any more.

In general, higher order local model are computationally costly and a query becomes more expensive than in global or local parametric learning systems. This would vote for locally constant models as the most appropriate way to do FPNL, e.g., as realized in the "Classification And Regression Trees" (CART) systems of Breiman et al. (1984). On the other hand, locally constant models tend to be highly biased at the edges of the input space and generalize inappropriate for simple smooth functions. Linear models offer a better compromise for local averaging: They have a good balance of bias and variance, and the amount of data needed to accomplish accurate estimates of the parameters is still moderate in comparison to second order and higher models (Cleveland et al., 1988b).

Local averaging in input spaces with many dimensions is said to suffer from the curse of dimensionality. This is illustrated in an example taken from Hastie and Tibshirani (1990). Suppose that points are uniformly distributed in a d-dimensional cube of unit edge length and that one wishes to construct a cube-shaped neighborhood capturing 10% of the data. The edge length of this subcube can easily be calculated to be $0.1^{1/d}$. For a one-dimensional space, the subcube has edge length 0.1, but for a 10-dimensional space it has edge length 0.8. Hence, the concept of *local* in terms of percentage of data fails in high dimensions. This requires the use of smaller neighborhoods in local averaging in high dimensions and thus much more data. However, such an argument only holds if one assumes that the input space has a uniform density. In many high dimensional problems, however, this is not the case. The data covers a subspace or submanifold of the entire state space since most of the regions in state space are impossible, for instance for physical reasons. On such submanifolds, the data is distributed much more densely and allows for local averaging methods. This property motivates the approach of Hastie and Tibshirani (1990) to seek a way out of the "curse of dimensionality" by means of "Generalized Additive Models", an approach which replaces the local multivariate averaging process by a sum of univariate (or low dimensional multivariate) averaging processes (similar too Friedman and Stützle's (1981) projection pursuit regression). The idea of these methods is to model the data with low dimensional projections of the entire state space. If the additive combination of the low dimensional functions spans the submanifold on which the data are distributed, the input-output correlation of the data can be modeled well. Generalized additive models, however, require an iterative training process.

Finally, FPNL systems need sufficient memory resources since, in the pure form, they do not attempt any kind of data compression. This purist attitude, however, is not always necessary. If the learning system only stores data which constitute a surprise, i.e., which it could not predict within a certain accuracy, much less data will be stored and the data collection process will rapidly slow down. Another concept would be that the FPNL consolidates nearby data points in some kind of receptive fields, and then uses the center of the receptive field and its activation as the data which enter the averaging process (Cleveland et al., 1988a). In contrary to the local parametric methods, however, the number of the receptive fields should be able to grow and shrink appropriately, which leads to the second form of nonparametric learning techniques in the next section.

2.3.2 Variable Number of Parameters

By extending the idea of local parametric learning to not restricting oneself to a certain number of local models, one obtains "Variable number of Parameters Nonparametric Learning" (VPNL). Actually, it is not at all necessary to confine VPNLs to consist of *local* functions, since global functions can be used, too. This results in two different strands of VPNLs, a *local* version and a *global* one.

As alluded to in the previous section, the local version of VPNLs performs data compression by creating receptive fields as needed, for instance in the form of partitions of a k-d tree. A local parametric model is then fitted to the subset of data inside of the receptive field, and only within this field is the local model valid. Receptive fields may be either non-overlapping hard partitions, or overlapping soft partitions. There usually exists a smoothing process between the neighboring partitions such as to avoid sudden jumps in the prediction of the learning box when the query location moves from one partition to another.

Most often the receptive fields are created by a recursive splitting technique. Initially, all data is thought to belong to one receptive field (or one partition). Whenever necessary, a receptive field is split into two, a recursive process which leads to a tree-like structure. A variety of possibilities of when and how to create a new receptive field have been suggested. In the CART system (Breiman et al., 1984), the splitting is driven to achieve the maximal variance reduction for each split; CART only fits locally constant models and is best suited for classification tasks. An extension of CART to regression problems is Friedman's "Multivariate Adaptive Regression Spline" (MARS) algorithm. MARS uses spline basis functions in each partition and achieves smooth fits of the data; the splitting is driven by a least squares criterion. Schaal & Atkeson (1994d) introduced a variant of k-d tree regression trees in which splitting is done until each partition has a valid local linear model. Although the k-d tree splits are hard, non-overlapping splits of the input space, the linear models are not confined to be built only of the data in one partition. This results in a piecewise linear representation with overlapping data support. Fritzke (1993) presented the "Growing Cell Structure", a Kohonen-like vector quantizer with a variable number of template vectors. In contrast to the algorithms above, no tree-like splitting scheme is employed, but the algorithm behaves rather like an elastic net into which a new node is inserted in the middle of an edge of the net if the edge is stretched too much. This, of course, results in an overall shift of all nodes in the net and adds a global component to the method. Most of the local VPNL algorithms also have methods to prune unnecessary partitions or nodes and try to accomplish optimal splitting and pruning with respect to some chosen criterion.

In contrary to local VPNLs, global VPNLs use global functions to model the input-output data. The key idea is to increase the number of components in the system when the current system becomes unable to model the data with a sufficiently small error. For instance, cascade correlation (Fahlman & Lebiere, 1990) is a neural network in which new hidden neurons are added in such cases. The new hidden neuron is connected to all inputs, all existing hidden neurons, and all output neurons. Adaptation, i.e., learning, only takes place for the weights of the connections from the input neurons to the new hidden neuron, but the connections weights of all hidden-to-output neuron connections are trained as well. The new hidden neuron should increase the performance of the network at the previous stage. The Upstart algorithm (Frean, 1990) and the work of Littmann and Ritter (1994) are similar to this procedure.

Advantages

VPNL systems can be trained very quickly in comparison to many other network models, and the computational effort to generate an output for a query is low. This is true for both local and global VPNL and has made these methods successful in many applications. Local VPNL is particularly suited for problems which require variable resolution of the input-output mapping, and also as a pre-processing stage for FPNL in order to add data compression to FPNL.

Global VPNL is a general purpose learning method and it could find an interesting application for quick, temporary adaptation. For instance, attaching a weight to a robot arm requires some learning in order to recover the normal performance of the arm. It would be unreasonable, however, to incorporate this adaptation of the arm control mechanism in a permanent way. A learning system which could learn to compensate for a change in the environment by adding structure to a previously learned structure, like global VPNL does, would be better. After a new change in the environment, e.g., when the weight was removed, the system should be reset to its original structure.

Disadvantages

Generally, it is hard to prune elements out of global VPNL systems since this will globally affect the performance. Methods which keep on adding computational elements to improve the current, seemingly incapable model, also run into the danger of overfitting the data and must thus deal with this issue. Local VPNL avoids such problems by adding resources such that they are only locally effective. The results of local VPNL are also much easier to interpret than the cascading of global error reducing elements. A dynamically changing environment, moreover, may cause a global VPNL system to grow indefinitely, while the local schemes, due to their pruning capabilities, will be able to keep up with the changes without indefinite growth. Both global and local VPNL requires iterative training methods which are slower than training in FPNL.

2.3.3 Memory Requirements of Nonparametric Learning

A frequent question about nonparametric learning systems is how much memory resources they need. As before, this question has to be answered separately for fixed parameter (FPNL) and variable parameter (VPNL) systems.

Since FPNL requires an evaluation of the nearest neighbors of a query point in order to generate a lookup it is inherently memory-based and may need large storage capacities for complicated functions. In contrast, VPNL methods are also targeted at data compression. For *local* VPNL, as soon as a receptive field (or partition) has sufficient statistical evidence that its predictions are within a certain error bound, the data within this partition can be discarded and the prediction within this partition is subsequently carried out by means of the parametric model of the partition. The benefit of retaining the data for as long as possible is that further splitting and pruning can be directly based on the data, as realized, for instance, in CART or MARS. If, however, one is willing to use less sophisticated statistics for splitting and pruning, e.g., just variances and means, the retention of these statistical variables alone suffices to determine future splits, and the data does not need to be kept in memory (e.g., Fritzke, 1993). As local VPNLs only grow locally, the adding or pruning of an element will not affect other regions of the state space such that retraining of other regions after a split is unnecessary. Hence, local VPNLs can function without remembering the data.

Global VPNLs add parameters to the learning box at a time when the performance of the learning system does not improve anymore despite significant residual error. As it is not differentiated where the error comes from, the added element in the model will globally interfere in order to reduce the error. Thus, just inserting this element into the current structure without pre-training will degrade the entire performance of the system. Pre-training, however, would require the memorization of at least a large enough fraction of the previously encountered data. Alternatively, the new element could be inserted into the learning system such that its initial parameter setting did not affect the current performance. When encountering new data this new element would be gradually “unfrozen” and integrated it into the overall performance. From this point of view, global VPNLs should not require the retention of old data in memory. However, they will be vulnerable to catastrophic interference problems due to their global characteristics and the retention of the data for re-training could help to avoid this.

3 Biological Relevance of Nonparametric Learning

This section will attempt to explore some parallels of nonparametric methods with biological information processing. Since such a review deserves a paper in its own right, the discussion will be held rather brief here.

First, global VPNL will be excluded from further considerations in this section. This is not to say that such a mechanism is biologically implausible, but it should be hard to interpret global VPNL neurophysiologically since the firing of many neurons of a VPNL system will just contribute as an error reduction signal. Due to the restricted memory capabilities of a brain, it is also unlikely to look for pure, memory-based FPNL methods. This leaves local VPNL by itself or in combination with FPNL for further examination.

The characteristic of local VPNL is the specialization into locally receptive fields which only react to input data falling into the range of the receptive field. This is in analogy with receptive fields in biology which elicit a response under a specific sensory stimulation. All the topographic maps in the cortex share the feature of receptive fields. That the number and range of the receptive fields can change has been shown in plasticity studies of the cortical organization, for example in somatosensory cortex by Merzenich et al. (1983) or in motor cortex by Hess and Donohue (1994). The possibility to reorganize requires the possibility to change the number of elements which contribute to the representation, a typical feature of nonparametric methods.

Besides the topology preserving maps, receptive fields are also found for other brain processes. In early visual information processing of higher vertebrates, a specialization of receptive fields to shapes at specific orientations is found (area V1, e.g.., Churchland & Sejnowski, 1992; Kandel et al., 1991). In area MT, similar receptive fields are found, but they are sensitive to movement directions. Another kind of local specialization is found in motor cortex, where certain neurons are sensitive towards the movement direction of a limb (Georgopoulus, 1991). Field (1994) argues that this kind of a representation which he calls “sparse distributed coding” may be one of the major principles in biological information processing, and he contrasts this viewpoint with “compact coding”, for example, a factorical code. The major feature of sparse distributed coding is that there is no attempt to reduce the number of dimensions of incoming signals like in compact coding but rather to have a divergence to many specialized receptive fields. Only few of these receptive fields will respond significantly to a stimulus, and only those will primarily contribute to the next stage of information processing. Again, this is the principle of local VPNL.

In the superior colliculus and in the motor cortex, the principle of population vectors seems to have relevance. The bottom layer of superior colliculus corresponds to a motor map for relative eye movements. Lee, Rohrer, and Sparks (1988) showed that the weighted average of the activation of the neurons of this layer corresponds to the relative movement the eye is going to make. A similar result was found by Georgopoulus (1991) in motor cortex of monkeys. Here the movement direction of the arm of the monkey coincided with the weighted average of direction selective cells in the cortex. Both these examples make use of locally weighted averaging methods, which is the major principle in FPNL.

In an investigation of learning of the hippocampus and the neocortex, McClelland et al. (1994) suggested that the hippocampus could be thought of as a fast learning system with minimal interference. From the understanding of the previous discussions of this paper, this minimal interference would demand for a local representation in the sense of FPNL and local VPNL.

In general, local specialization and averaging of the responses of specialized elements seems to be a key concept in biological information processing. The formation of the locally specialized elements must be a learning process which flexibly recruits and prunes computing elements in order to suit the desired information processes. The summarizing of certain responses by local averaging also seems to be a concept which is made use of. These are the processes which nonparametric learning theories try to understand. It would be interesting to find out whether there exists some kind of higher order averaging as in locally linear models.

4 Discussion

One important part of learning is the establishing of nonlinear correlation between input and output data, for example, between sensory information and appropriate actions. Two ways of approaching this problem can be taken, parametric and nonparametric -methods. Parametric approaches have to make assumptions about the global structure of the problem to be learned. In the case that these assumption are appropriate, the parametric learning system will perform well. However, if these assumptions do not meet the structure of the problem to be learned, parametric learning will not achieve satisfying results.

Nonparametric learning, in contrast, tries to avoid modeling the global structure of a problem. It does so by focusing only on the local structure around a point of interest, i.e., a query point, or by adding computational resources as needed. A nonparametric method cannot outperform a well chosen parametric system but can at most achieve the same level of performance. If the problem to be modeled is unknown, however, it can be expected that a nonparametric model will, on average, have superior performance than the parametric counterpart. As most of the nonparametric approaches consist of local models, they avoid catastrophic interference, an inherent problem of parametric learning. This and the fast training of nonparametric learning makes it ideally suited for incremental learning and learning in dynamic environments. As a disadvantage, nonparametric learning sometimes needs more computational power for a lookup and more memory resources. However, these disadvantages can be overcome by combinations of different kinds of nonparametric learning and parallelization.

Local specialization seems to be a key concept in biological information processing. A variety of neurophysiological investigations showed results which can best be described in the framework of nonparametric regression. Many more processes could make use of the principles of these techniques, such that nonparametric methods may offer an interesting alternative for biological modeling.

An issue which, so far, has not received appropriate attention in the discussions of this paper is the bias/variance tradeoff in learning (Geman et al., 1992). The desired goal of a learning system is to faithfully model a given input-output mapping by minimizing both the bias and the variance of the model. In order to be truly unbiased, the learning system has to sacrifice variance until it has seen sufficiently many experiences—for complex function this may be a huge amount of data. On the other hand, in order to minimize variance and generalize well early on, the system usually has to sacrifice bias. This is the so-called bias/variance dilemma. Geman et al. (1992) correctly argue that for many biological and machine learning problems, it is almost impossible to achieve computationally feasible results without some bias. Since nonparametric learning systems inherently try to avoid structural bias, the question arises whether a nonparametric approach to learning is a reasonable approach at all. To answer this question, it is useful to examine some kinds of biases which can be employed. As mentioned before, if a parametric learning system chooses a favorable structure for the problem at hand, there will be no way that a nonparametric approach outperforms the parametric system. This bias concerns the representation inside of the learning box of the function to be learned. However, there are many other forms of biases. The chosen input-output representation, i.e., which variables are inputs to the learning box and which are the outputs, plays a dominant role in the success of learning. A good choice of the input-output representation can make learning trivial (e.g., Schaal et al., 1990; Schaal & Sternad, 1992). Geman et al (1992) demonstrate how a bias on the distance metric in a nonparametric learning system can increase the learning performance significantly. In local averaging (FPNL), the type of averaging function, e.g., constant, linear, quadratic, or cubic, is a bias on the generalization properties of the nonparametric learning system.

Thus, in general, there are a variety of ways to include bias in nonparametric systems. The emphasis of being unbiased in nonparametric regression is concerned with the internal representation of the function to be learned and does not mean that the system is—and must be—unbiased with respect to all issues. However, this brief discussion of the importance of bias also serves as a reminder of the first lines of this paper, namely that learning involves much more than just establishing an input-output mapping between two sets of variables.

Acknowledgments

The author is grateful for the very helpful remarks on the manuscript of Christopher G. Akteson and Peter Dayan. Support was provided by the Air Force Office of Scientific Research, by the Siemens Corporation, the German Scholarship Foundation and the Alexander von Humboldt Foundation.

References

Albus, J. S. (1975). "A new approach to manipulator control: The Cerebellar Model Articulation Controller (CMAC)." *ASME Journal of dynamic systems, Measurements, Control*, **97**, pp.228-233.

Atkeson, C. G. (1992). "Memory-based approaches to approximating continuous functions." In: Casdagli, M., & Eubank, S. (Eds.), *Nonlinear Modeling and Forecasting*, pp.503-521. Redwood City, CA: Addison Wesley.

Barron, A. R. (1994). "Approximation and estimation bounds for artificial neural networks." *Machine Learning*, **14**, 1, pp.115-133.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth International Group.

Churchland, P. S., & Sejnowski, T. J. (1992). *The computational brain*. Boston, MA: MIT Press.

Cleveland, W. S. (1979). "Robust locally weighted regression and smoothing scatterplots." *Journal of the American Statistical Association*, **74**, pp.829-836.

Cleveland, W. S., Devlin, S. J., & Grosse, E. (1988a). "Regression by local fitting: Methods, properties, and computational algorithms." *Journal of Econometrics*, **37**, pp.87-114.

Cleveland, W. S., & Devlin, S. J. (1988b). "Locally weighted regression: An approach to regression analysis by local fitting." *Journal of the American Statistical Association*, **83**, pp.596-610.

Cohn, D. A. (1994). "Neural network exploration using optimal experiment design." In: Cowan, J. , Tesauro, G., & Alspector, J. (Eds.), *Advances in Neural Information Processing Systems 6*, pp.679-686. San Mateo, CA: Morgan Kaufmann.

Crain, I. K., & Battacharyya, B. K. (1967). "Treatment of nonequispaced two-dimensional data with a digital computer." *Geoexploration*, **5**, pp.173-194.

Cybenko, G (1989). "Approximation by superpositions of a sigmoidal function." *Mathematics of Control, Signals, and Systems*, **2**, pp.303-314.

de Boor, C. (1978). *A practical guide to splines*. New York: Springer.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). "Maximum likelihood from incomplete data via the EM algorithm." *Journal of the Royal Statistical Society B*, **39**, pp.1-38.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: Wiley.

Eubank, R. L. (1988). *Spline smoothing and nonparametric regression*. New York: Marcel Dekker.

Fahlman, S. E. , Lebiere, C. (1990). "The cascade-correlation learning architecture." In: Touretzky, D. S. (Ed.), *Advances in Neural Information Processing Systems II*, pp.524-532. San Mateo, CA: Morgan Kaufmann.

Falconer, K. J. (1971). "A general purpose algorithm for contouring over scattered data points." Nat. Phys. Lab. Report NAC 6, .

Fan, J., & Gijbels, I. (1992). "Variable bandwidth and local linear regression smoothers." *The Annals of Statistics*, **20**, 4, pp.2008-2036.

Farmer, J. D., & Sidorowich (1987). "Predicting chaotic time series." *Phys. Rev. Lett.*, **59 (8)**, pp.845-848.

Farmer, J. D., & Sidorowich (1988b). "Exploiting chaos to predict the future and reduce noise." In: Lee, Y. C. (Ed.), *Evolution, Learning, and Cognition*, p.27. Singapore: World Scientific.

Farmer, J. D., & Sidorowich (1988d). "Predicting chaotic dynamics." In: Kelso, J. A. S., Mandell, A. J., & Schlesinger, M. F. (Eds.), *Dynamic Patterns in Complex Systems*, pp.265-292. New Jersey: World Scientific.

Field, D. J. (1994). "What is the goal of sensory coding?." *Neural Computation*, **6**, pp.559-601.

Fix, E., & Hodges, J. L. (1951). "Discriminatory analysis, nonparametric regression: Consistency properties." Project 21-49-004, Report No.4, Contract AF-41-(128)-31, USAF School of Aviation Medicine Randolph Field, Texas.

Frean, M. (1990). "The upstart algorithm: A method for constructing and training feedforward neural networks." *Neural Computation*, **2**, pp.198-209.

Friedman, J. H., Bentley, J. L., & Finkel, R. A. (1977). "An algorithm for finding best matches in logarithmic expected time." *ACM Transactions on Mathematical Software*, **3 (3)**, pp.209-226.

Friedman, J. H., & Stützle, W. (1981b). "Projection pursuit regression." *Journal of the American Statistical Association, Theory and Models*, **76**, 376, pp.817-823.

Funahashi, K (1989). "On the approximate realization of continous mappings by neural networks." *Neural Networks*, **2**, pp.183-192.

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

Georgopoulus, A. P. (1991). "Higher order motor control." *Annual Review of Neuroscience*, **14**, pp.361-377.

Hájek, J. (1969). *A course in nonparametric statistics*. San Francisco, CA: Holden-Day.

Härdle, W. (1991). *Smoothing techniques with implementation in S*. New York: Springer.

Hastie, T. J., & Tibshirani, R. J. (1990). *Generalized additive models*. London: Chapman and Hall.

Hecht-Nielson, R. (1989). "Theory of the backpropagation neural network." In: *International Joint Conference on Neural Networks*, **1**, pp.593-611, Washington.

Hertz, J., Krogh, A., & Palmer, R. G. (1991). *Introduction to the theory of neural computation*. Redwood City, CA: Addison Wesley.

Hess, G., & Donoghue, J. P. (1994). "Long-term potentiation of horizontal connections provides a mechanism to reorganize cortical motor maps." *Journal of Neurophysiology*, **71**, 6, pp.2543-2547.

Hinton, G. E., & Sejnowski, T. J. (1983). "Optimal perceptual inference." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, p.448-453, Washington, DC.

Hinton, G. E. (1986). "Learning distributed representation of concepts." In: *The Eighth Annual Conference of The Cognitive Science Society*, p.1-12, Amherst, MA.

Hopfield, J. J. (1982). "Neural networks and physical systems with emergent collective computational abilities." *Proc. Natl. Acad. Sci. USA,*, **79**, pp.2554-2558.

Jacobs, R. A. (1988). "Increased rates of convergence through learning rate adaptation." *Neural Networks*, **1**, pp.295-307.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). "Adaptive mixtures of local experts." *Neural Computation*, **3**, pp.79-87.

Jordan, M. I., & Jacobs, R. (1994). "Hierarchical mixtures of experts and the EM algorithm." *Neural Computation*, **6**, pp.79-87.

Kandel, E. R., Schwartz, J. H., & Jessell, T. M. (1991b). *Principles of neural sciences*. 3rd edition. New York: Elsevier.

Kolmogorov, A. N. (1957). "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition." *Dokl. Akad. Nauk. USSR*, **114**, pp.953-956 [in Russian].

Lee, C., Rohrer, W. R., & Sparks, D. L. (1988). "Population coding of saccadic eye movement by neurons in the superior colliculus." *Nature*, **332**, pp.357-360.

Littmann, E., & Ritter, H. (1993). "Generalization abilities of cascade network architectures." In: Hanson, S. J., Cowan, J. , & Giles, C. L (Eds.), *Advances in Neural Information Processing Systems 5*, pp.188-195. Morgan Kaufmann.

Lowe, D. G. (1993). "Similarity metric learning for a variable-kernel classifier." Submitted paper. Contact: lowe@cs.ubc.ca, .

Müller, H. -G. (1988). *Nonparametric regression analysis of longitudinal data*. Lecture Notes in Statistics Series, vol.46. Berlin: Springer.

Macauley, F. R. (1931). *The smoothing of time series*. New York: National Bureau of Economic Research.

MacKay, D. J. C. (1992). "Bayesian interpolation." *Neural Computation*, **4**, 3, pp.415-447.

Marr, D. (1971). "Simple memory: A theory for archicortex." *The Philosopical Transactions of the Royal Society of London B*, **262**, pp.23-81.

McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1994). "Why there are complementary learning systems in the hyppocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory." Technical Report PDP.CNS.94.1, Pittsburgh, PA: Department of Psychology, Carnegie Mellon University.

McLachlan, G. J., & Basford, K. E. (1988). *Mixture models*. New York: Marcel Dekker.

McLain, D. H. (1974). "Drawing contours from arbitrary data points." *The Computer Journal*, **17 (4)**, pp.318-324.

Merzenich, M. M., Kaas, J. H., Nelson, R. J., Sur, M., & Felleman, D. (1983). "Topographic reorganization of somatosensory cortical areas 3b and 1 in adult monkeys following restricted deafferentation." *Neuroscience*, **8**, pp.33-55.

Minsky. M., & Papert, S. (1969). *Perceptrons: An introduction to computational geometry*. Cambridge, MA: MIT Press.

Moody, J., & Darken, C. (1988). "Learning with localized receptive fields." In: Touretzky, D., Hinton, G., & Sejnowski, T. (Eds.), *Proceedings of the 1988 Connectionist Summer School*, pp.133-143. San Mateo, CA: Morgan Kaufmann.

Myers, R. H. (1990). *Classical and modern regression with applications*. Boston, MA: PWS-KENT.

Nadaraya, E. A. (1964). "On estimating regression." *Theor. Prob. Appl.*, **9**, pp.141-142.

Nowlan, S. J. (1991). "Soft competitive adaptation: Neural network learning algorithms based on fitting statistical mixtures." Technical Report CMU-CS-91-126, Pittsburgh, PA: Carnegie Mellon University.

Omohundro, S. (1987). "Efficient algorithms with neural network behaviour." *Complex Systems*, **1**, pp.273-347.

Parzen, E. (1962). "On estimation of a probability density function and mode." *Annals of Mathematical Statistics*, **33**, pp.1065-1076.

Poggio, R., & Girosi, F. (1990). "Regularization algorithms for learning that are equivalent to multilayer networks." *Science*, **247**, .

Ripley, B. D. (1992). "Statistical aspects of neural networks." In: *Proceedings of Séminaire Européen de Statistique*, April 25-30, Sanbjerg, Denmark.

Ritter, H., & Schulten, K. (1988b). "Convergence properties of Kohonen's topology conserving maps: Fluctuations, stability, and dimension selection." *Biological Cybernetics*, **60**, pp.59-71.

Ritter, H., Martinetz, T., & Schulten, K. (1992). *Neural computation and self-organizing maps – An introduction*. Redwood City, CA: Addison-Wesley.

Schaal, S., Atkeson, C. G., & Botros, S. (1992b). "What should be learned?." In: *Proceedings of Seventh Yale Workshop on Adaptive and Learning Systems*, p.199-204, New Haven, CT.

Schaal, S., & Sternad, D. (1993c). "Learning passive motor control strategies with genetic algorithms." In: Nadel, L., & Stein, D. (Eds.), *1992 Lectures in complex systems*. Redwood City, CA: Addison-Wesley.

Schaal, S., & Atkeson, C. G. (1994a). "Robot juggling: An implementation of memory-based learning." *Control Systems Magazine*, **14**, 1, pp.57-71.

Schaal, S., & Atkeson, C. G. (1994b). "Assessing the quality of learned local models." In: Cowan, J. , Tesauro, G., & Alspector, J. (Eds.), *Advances in Neural Information Processing Systems 6*. Morgan Kaufmann.

Schaal, S., & Atkeson, C. G. (1994d). "Robot learning by nonparametric regression." In: *Proceedings of the IROS Conference*, Munich, Germany.

Scott, D. W. (1992). *Multivariate Density Estimation*. New York: Wiley.

Sheppard, W. F. (1912). "Reductions of errors by means of negligible differences." In: Hobson, E. W., & Love, A. E. H. (Eds.), *Proceedings of the Fifth International Congress of Mathematicians*, **II**, pp.348-384. Cambridge, MA: Cambridge University Press.

Sherriff, C. W. M. (1920). "On a class of graduation formulae." In: *Proceedings of the Royal Society of Edinburgh*, **XL**, pp.112-128.

Silverman, B. W. (1985). "Some aspects of the spline smoothing approach to nonparametric regression curve fitting." *Journal of the Royal Statistical Society B*, **47**, pp.1-52.

Specht, D. F. (1991). "A general regression neural network." *IEEE Transactions on Neural Networks*, **2**, 6, Nov..

Sprecher, D. A. (1965). "On the structure of continuous functions of several variables." *Transactions of the American Mathematical Society*, **115**, pp.533-541.

Stanfill, C., & Waltz, D. (1986b). "Towards memory-based reasoning." *Communications of the ACM*, **29 (12)**, pp.1213-1228.

Stone, M (1974). "Cross-validatory choice and assessment of statistical predictors." *Journal of the Royal Statistical Society*, **B36**, 111-147.

Sutton, R. S. (1992). "Gain adaptation beats least squares." In: *Proceedings of Seventh Yale Workshop on Adaptive and Learning Systems*, New Haven, CT.

Tibshirani, R., & Hastie, T. (1987). "Local likelihood estimation." *Journal of the American Statistical Association*, **82**, 398, pp.561-567.

Vapnik, V. N. (1982). *Estimation of dependences based on empirical data*. Berlin: Springer.

Wahba, G., & Wold, S. (1975). "A completely automatic french curve: Fitting spline functions by cross-validation." *Communications in Statistics*, **4 (1)**, .

Watson, G. S. (1964). "Smooth regression analysis." *Sankhaya: The Indian Journal of Statistics A*, **26**, pp.359-372.

Whittaker, E., & Robinson, G. (1924). *The calculus of observations*. London: Blackie & Son.

Zemel, R. S. (1993). "A minimum description length framework for unsupervised learning." PhD. Thesis, Toronto: University of Toronto, Department of Computer Science.

Zipser, D., & Anderson, R. A (1988). "A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons." *Nature*, **331**, 6158, pp. 679-684.