Scaling properties of neural classifiers

Włodzisław Duch
Department of Computer Methods, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland.
E-mail: duch@phys.uni.torun.pl

Abstract

One of the problems in evaluating the usefulness of various classification techniques for real-world applications is the lack of information about their scaling properties: how will the complexity of a given method depend on the number of attributes, classes and cases? In this paper a simple classification task is presented as a challenge for different methods, allowing to determine their scaling properties as well as to evaluate the complexity of the solutions obtained by various methods.

I. Introduction

DESPITE of the great number of conferences and papers on machine learning, statistics, pattern recognition and neural networks the relative advantages and disadvantages of various classification methods are still not known. Experts in one field rather seldom talk to experts in other fields and as a result it is hard to recommend an optimal classification method for a complex, real life problem. Several papers have addressed some aspects of this problem in the past. First, there is a large mathematical literature on computational learning – cf. three volumes on computational learning theory edited by Hanson *et.al* [1], or some recent books on neural networks [2]. This theory tries to elucidate such issues like the size of the training set needed for reliable classification, find approximate error bounds and compare performance of different classifiers.

Second, several empirical comparisons have been performed between different classification systems on various data sets. Weiss and Kapouleas [3] were among the first to perform such empirical comparisons of statistical, machine learning and neural classification methods on several medical datasets. Recently Rhower and Morciniec [4] have made an extensive comparison of 24 classification methods on eleven datasets. Even such large scale study was not helpful in determining the relative merits of classification methods: differences between many methods are within a few percent, which is not significant [2]. In case of methods that use many adjustable parameters (such as neural networks) an additional hard problem is to find a real optimal solution, so one cannot claim with confidence that the results obtained are the best that an MLP network may give. Results of all this theoretical and empirical investigations have led so far to even greater confusion. It is not clear that from the fact that a given method performed better for the dataset X one can draw more conclusions than ... well, that it performs better for the dataset X.

In this paper another question related to performance is addressed: what are the scaling properties of the classification methods, their complexity when applied to classification problems with N attributes, M classes and n samples. A benchmark example, using a prototype of real world data, is needed to show the dependence of the total time needed for classification on these three parameters. The evaluation of the quality of solutions should be done not only from the point of view of the number of classification errors, but also the complexity of the classificators (networks, trees) measured by the number of adaptive parameters that are necessary to accomplish the task. In some cases it may also be possible to derive simple logical rules and compare their number and quality.

The scaling problem in MLPs has been directly addressed by Tesauro and Janssens [5], who made some experiments for the parity problem and concluded, that the time scales exponentially with the size of the problem N (length of the binary string). There is a simple way to solve the parity problem in $\log_2 N$ time using a hierarchical multilayered structure, where at each level the nodes are designed to solve the two-input parity (i.e. XOR) problem. There are actually two issues at stake here. First, how hard is it to find a network structure performing tasks of increasing complexity, assuming that one starts with large network and uses some optimization procedure to search for the best solution. Simple heuristics, like "divide and conquer", may help to solve the problem using hierarchical multilayered network structure, although brute force error minimization approach is NP-hard. Second, what is the limit of the method, i.e. providing that we can somehow find an optimal solution, how will it scale with the data? Methods that scale with high power of the number of attributes (or classes) have little chance to perform well in the real world tasks when the number of attributes (or classes) is high. For such methods minimal number of parameters that should be determined is large and an optimal solution is much harder to find than for the methods scaling with lower powers of N or M.

In the next section I have presented a challenging classification task suitable to determine scaling properties of classifiers. In the third section simple neural network solutions to this task are presented from the theoretical point of view. More detailed description of this classification problem and solutions found so far is available [6].

II. The challenge

Consider a classification problem with M classes and N independent attributes. The values of attributes $X = (x_1, x_2 ... x_N)$ are obtained from measurements and renormalized to $0 \leq X_i \leq M$, i.e. within a hypercube of a side M. Each vector in this hypercube has a class label $C_k = 1, 2, ..M$ assigned in the following way: the data range $[0, M]$ is divided into M sectors $[k-1, k), k = 1..M$, defining a series of hypercubes contained in progressively larger ones. The smallest of these hypercubes, H_1 has the side of length $l = 1$; it is contained in the second hypercube, H_2, with the side $l = 2$, and so on, up to the biggest H_M hypercube with side $l = M$. Vectors belonging to class C_k are in the set $\{H_k - H_{k-1}\}$, where H_0 is an empty set and H_M is the largest hypercube (Fig. 1). This artificial example with simple logical structure is a model for some real-life problems. The challenge is: given n labeled training samples $(X_i, C_i), i = 1..n$ assign test vectors to one of M classes and determine scaling properties of the classification method used, i.e. determine its complexity $T(N, M, n)$. Test the ability to make maximal generalizations consistent with the data. The density of the input vectors per unit input volume in high dimensional case is very small, therefore only the methods that are able to generalize (extrapolate) in agreement with constraints given by the input training vectors have a chance to perform well. A simple decision rule assigns a class label C to a vector $X = \{x_i\}$. It is enough to check to which sector the maximal component of X belongs:

$$\text{IF } (k-1 \leq \max_i X_i < k) \text{ THEN } C = C_k \quad (1)$$

Logical rules of this form have the lowest algorithmic complexity and should be preferred. The goal of the data modeling should not be just classification but rather an attempt to find the simplest model generating the data. There are several variants of this problem that one may investigate, but only the simplest case is discussed here since this seems to be sufficient to determine the scaling behavior [6].

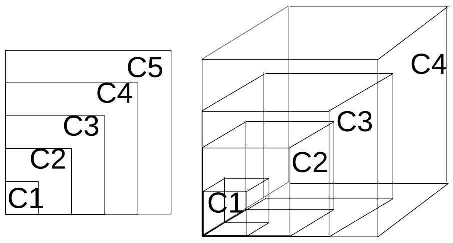

Fig. 1. Shapes of classes in 2D, 5-class, and 3D, 4-class problem.

A very rough estimation of the behavior of generalization error in MLPs [7] shows that it should be proportional to the ratio of the number of model parameters to data samples. How many data points should be used? Samples should include all “interesting” points near the vertices of the hypercubes. The class boundary hypersurfaces are always $N-1$ dimensional. To select points near the vertices inside the hypercube, a small number ξ, for example 0.1, is added or subtracted from the coordinates of the vertices; to assure that all points are inside of the cube $b_i - \xi$ is changed to $|b_i - \xi|$. Thus 2^N inner cube points are defined by $|b_1 - \xi|, |b_2 - \xi|, ...|b_N - \xi|$, and $2^N - 1$ outer points by $b_1 + \xi, b_2 + \xi, ...b_N + \xi$. For M classes $2M(2^N - 1) + 1$ points are defined in this way. This amounts to 1261 points for $N = 6$ dimensions and $M = 10$ classes, 5101 points for $N = 8$ dimensions and $M = 10$ classes, and 20461 points for $N = 10$, $M = 10$. Investigation of the scaling properties of classification methods in such range, from $N = 1$ to $N = 10$ dimensions and from $M = 2$ to $M = 10$ classes is realistic. Of course some real-world classification problems require much more than ten attributes or ten classes, but it should be possible to draw some conclusions about scaling properties of methods from the behavior on the problems presented here.

A large number of test points may be used. For one and two-dimensional cases graphical representation of class borders is advocated, formed by testing each point on a 0.1 grid – for the ten class case this is 10.000 points for testing. For higher number of dimensions the same algorithm as used for the training points, with $\xi = 0.2$ and $\xi = 0.3$ generates $4M(2^N - 1) + 1$ test points.

III. Classification by MLP, L-R and RBF networks

MLPs are capable of making good classifications, especially if a global minimum error solution is found and the error function includes some regularization terms to select the minimum complexity network structure. The same is true for RBF networks, therefore these two methods are applied here to the hypercube classification task. In addition performance of new L-R networks [8] is analyzed. It is very hard to prove formally that the architectures presented here are really the simplest solutions of the classification problem using MLPs, so they may be properly treated as an upper bounds on the complexity and scaling of the MLP. In one-dimensional case the simplest MLP solution consists of a single layer of M hidden nodes and one output node, with all weights equal to $+1$ and biases from 0 to $M-1$. The network realizes a multistep function using M neurons, M weights and M biases.

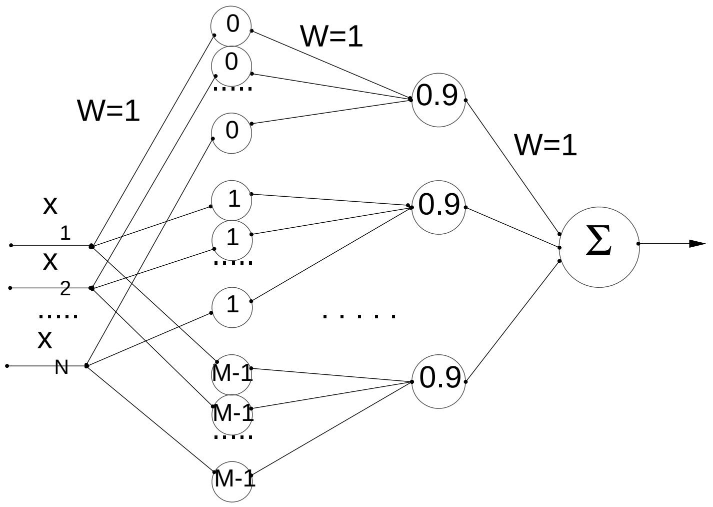

Fig. 2. Final structure of the MLP network for the N-dimensional M-class case.

There are several simple structures of MLPs capable of perfect representation of our classification task for $N > 1$. The simplest one found so far (Fig. 2) is a generalization of one dimensional case and consists of two hidden layers. The first layer has $M \cdot N$ neurons and the second M neurons. All weights are equal to $+1$. Each neuron of the first hidden layer is connected to one input unit. For the first group on N hidden neurons all biases are equal to 0, for the second to 1, and for the last group to $M-1$. Outputs of neurons from each group are connected to one neuron in the second hidden layer. Neurons of the second hidden layer, called here "class neurons", have biases $\theta = 0.9$ (values $\theta = 1$ lead to an undesired symmetry). The output neuron computes weighted sum giving linear output. This network computes the following function:

$$F(X; W; B) = \sum_{j=0}^{M-1} \sigma \left(\sum_{i=1}^{N} \sigma(X_i + j) - \theta \right) \qquad (2)$$

The total number of neurons is the same as the total number of biases and is equal to $MN + M + 1$, while the number of weights (all equal to one) is $M(2N+1)$. The total number of adjustable parameters is $M(3N+2)+1$, scaling linearly with the number of classes and the number of attributes. $M+1$ nodes for the C_1 class may actually be removed if a nonlinear output neuron is used, with flat characteristics and a bias plus the output weight set to give an output 1 for activation 0, 2 for activation 1, and M for activation $M-1$. The total number of parameters in this case becomes $3MN + 2(M-N)$ plus one variable output slope. In the harder case of rotated hypercubes an additional layer with N neurons is needed to perform the rotation, contributing N^2 weights and N biases. Thus the scaling is quadratic with the number of attributes and linear with the number of classes. Finding network structures presented here using architecture optimization techniques of MLPs may be quite hard, but it should be very interesting to see what kind of simplified network structures can be obtained by various methods that enforce penalty on the complexity of the network.

L-R networks networks are variants of MLP with specific “linguistic” or L-nodes in the first hidden layer, designed to discover one-dimensional features in the data [8]. Features may be local as well as unbounded. The second hidden layer tries to combine these features into logical rules. The network structure is presented in Fig. 3. The L-neurons in the first hidden layer have two biases, the first bias is always zero and the second takes values from 1 to M. All weights are equal to $+1$ and the network gives perfect classification in the $M+1$ class case, i.e. it gives zero output outside the hypercube. The output neuron performs linear summation and has no adaptive parameters. The number of neurons is $MN + M + 1$, weights and biases $2MN + M$, so the total number of adaptive parameters is $4MN + 2M$.

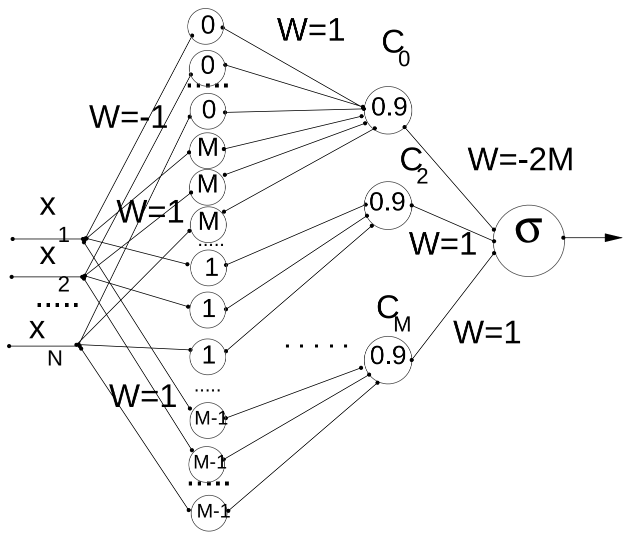

Fig. 3. Final structure of the MLP network for the N-dimensional $M+1$ class case.

Radial basis function networks are frequently based on Gaussian functions. Such networks may have difficulties with representing the class boundaries in our classification problem –

in the worst case the number of nodes is ot the order of the number of training points $M2^N$. Distances between the points grow with the number of classes. Although RBF networks with Gaussians transfer function seem to be difficult to construct other radial basis functions should work better. For the multiquadratic functions $\sqrt{\alpha^2 + \beta r^2}$ similar network structures as for MLPs should solve the problem using the same number of parameters.

IV. Discussion and summary

Benchmarks play an important role in testing neural systems (cf. the popularity of the parity or the two spiral problems). I have introduced here a challenging benchmark problem that may be used to determine scaling properties of classificators. Neural methods analyzed above scale like $O(MN)$. On the other hand memory-based methods, such as the nearest neighbor method, or RBF with Gaussian functions, should scale like the number of points for training, i.e. $O(M2^N)$. It should be very interesting to see how other methods of classification, developed by statistical, neural, pattern recognition and machine learning scale when applied to the classification problem presented here.

Two very simple MLP solutions to the hypercube classification have been found and their complexity determined. Although this does not tell us directly how hard it is to find such network structures through architecture optimization, knowing that these solutions exists, and knowing their complexity, is very useful for benchmarking neural network optimization methods. Average times reported for finding good solutions should give an indication how hard it is to find them, although they can also be misleading, depend on many factors, such as random initialization procedures.

Acknowledgments: This research has been partially supported by the Polish Committee for Scientific Research, grant 8T11F 00308. W.D. is grateful to prof. Masumi Ishikawa for numerous discussions and his hospitality at the Kyushu Institute of Technology, where most of this paper was written.

References

[1] Computational Learning Theory and Natural Learning Systems, Volume I: Constraints and Prospects, ed. by S.J. Hanson, G.A. Drastal, R.L. Rivest, Bradford Book 1994; Volume II: Intersections between Theory and Experiment, ed. by S.J. Hanson, T. Petsche, M. Kearns, R.L. Rivest, Bradford Book 1994; Volume III: Selecting Good Models, ed. by T. Petsche, S.J. Hanson, J. Shavlik, Bradford Book 1995

[2] B.D. Ripley, Pattern Recognition and Neural Networks (Cambridge University Press 1995); M.H. Hassoun, Fundamentals of Artificial Neural Networks (MIT Press, Bradford Book, 1995); S. Haykin, Neural Networks. A Comprehensive Foundation (IEEE Press 1994); C. Bishop, Neural networks for pattern recognition (Clarendon Press, Oxford 1995)

[3] S. Weiss, I. Kapouleas, An empirical comparison of pattern recognition, neural nets and machine learning classification methods. In: *Int. Joint Conference on AI*, Detroit, Michigan, pp. 781-787

[4] R. Rohwer and M. Morciniec, A Theoretical and Experimental Account of n-tuple Classifier Performance, *Neural Computation* 8 (1996) 657–670

[5] G. Tesauro and R. Janssens, Scaling relationships in back-propagation learning, Complex Systems 2 (1988) 39-44

[6] W. Duch, Scaling of Classification Methods, *Kyushu Institute of Technology Technical Report KIT/TR-1-96,* http://www.phys.uni.torun.pl/kmk/publications.html

[7] E. Baum and D. Haussler, What size net gives valid generalization? *Neural Comput.* 1 (1989) 151-160

[8] W. Duch, R. Adamczak, K. Grąbczewski, Constrained backpropagation for feature selection and extraction of logical rules, *First Polish Conference on Theory and Applications of Artificial Intelligence,* Łódź,19-21.12.1996, pp. 163-170