Designing Neural Networks from Statistical Models: A new approach to data exploration

Antonio Ciampi* and Yves Lechevallier**
*Department of Epidemiology & Biostatistics, Mcgill University
1020 Pines Ave. West
Montreal (Quebec), Canada H3A 1A2 ciampi@icm.umontreal.ca
**INRIA-Rocquencourt, Domaine de Voluceau
78153 Le-Chesnay Cedex, France

Abstract

We develop, in the context of discriminant analysis, a general approach to the design of neural architectures. It consists in building a neural net 'around' a statistical model family; larger networks, made up of such elementary networks, are then constructed. It is shown that, on the one hand, the statistical modeling approach provides a systematic way to obtaining good approximations in the neural network context, while, on the other, neural networks offer a powerful expansion to classical model families. A novel integrated approach emerges, which stresses both flexibility (contribution of neural nets) and interpretability (contribution of statistical modeling). A well known data set on birth weight is analyzed by this new approach. The results are rather promising and open the way to many potential applications.

1. Introduction

Statistical models play an important role in data exploration. Typically, a data set is looked at from the point view of a model family, large enough to contain 'interesting' alternative 'explanations' of a data structure, yet simple enough to permit quick screening, so that one or several best fitting alternatives may be selected for an in-depth look. We will restrict ourselves to the problem of predicting a class variable c from a vector of classifiers $\mathbf{z}$. From a statistical perspective, the goal is to model the conditional probability $P(c \mid \mathbf{z})$. This is done either by the classical approach which specifies a model family at the outset, or by the adaptive approach, by which the model is arrived at by a sequence of data driven steps. The classical statistical approach has been to fit generalized linear models, in particular, for predicting a probability, the logistic model. More recently, the inadequacy of the linear model to handle problems frequently occurring in the applications, has spurred statisticians in the direction of model families which go beyond linearity, such as the generalized additive model (GAM) (Hastie & Tibshirani 1990). The adaptive modeling approach has produced regression trees (Breiman et al. 1984, Ciampi 1991). For any given family, however, no matter how complex or general, it is always possible to find a realistic situation which does not fit it. The inadequacies of a model family are exposed by large data sets, such as those that are becoming increasingly available in every field of science and engineering.

On the other hand, neural nets have, by now, a long history and are well established in the applied field as a flexible and powerful tool for solving prediction and pattern recognition problems: see (Hecht-Nielsen, 1990) for both a historical discussion and a review of important applications. The neural network approach finds its justification in a central theorem which claims that, under certain 'reasonable' conditions, *any* function can be *approximated* by a feedforward neural network with one hidden layer. This 'universality' of the neural approach is given a tremendous applied potential by the existence of a general learning paradigm, the *back-propagation* algorithm. From the point of view of data exploration, however, neural nets suffer from a serious shortcoming, in view of their well-known 'black box' character. No matter how well or how poorly a neural network performs, it is extremely difficult to understand *how* it works. In contrast, a statistical model contains in itself an explanation of its successes or failures. In view of this complementarity of the two approaches, it is not surprising that statisticians are now becoming interested in studying the properties of neural nets (Bing Cheng & Titterington, Ripley 1994), and that neural nets specialists draw more and more from statistical inference (Geman, Bienenstock & Dorsat 1992).

In this paper we will outline an approach to the construction of neural networks which uses statistical models for designing the architecture of the network and for providing initial conditions to the back-propagation algorithm. The approach generalizes previous work in which classification trees were used in a similar way, as the 'initial' statistical model (Sethi 1990, Brent 1991, Chabanon, Lechevallier & Millemann 1992).

The correspondence between networks and statistical models is developed in Section 2. An example is discussed in Section 3, and Section 4 contains a brief conclusion.

2. Representation by neural networks of statistical models for discrimination

The logistic model is used commonly to construct classifiers in the 2-class case. For simplicity of notation we assume that the vector of the classifiers contains a component which is identically equal to 1 (constant term). Then the model is written as:

$$\text{logit}\,(P(c = c_1|\mathbf{z})) = \beta \cdot \mathbf{z} \qquad (2.1)$$

and it is immediately representable as the network with no hidden layers given by Figure 1, where the weight vector is to be identified with β and $s(\mathbf{z})$ with $P(c = c_1|\mathbf{z})$. Maximum-likelihood methods are currently used for estimating the unknown vector coefficients from the data. With a cost function given by the negative log-likelihood, the network can also produce a maximum likelihood estimate of the weight vector via the back-propagation algorithm.

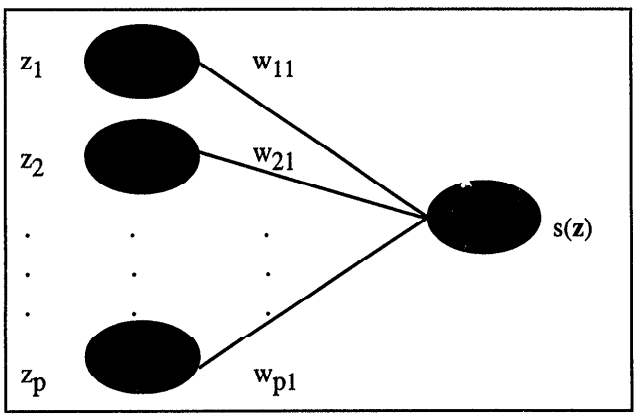

Figure 1. Neural Network for logistic regression.

There are two generalizations of (2.1) corresponding to two non-linear approaches for modeling $P(c = c_1|\mathbf{z})$. One approach generalizes linearity on the logistic scale to additivity on the same scale:

$$\text{logit}\,(P(c = c_1|\mathbf{z})) = g_1(z_1) + g_2(z_2) + \ldots + g_p(z_p) \qquad (2.2)$$

where the g's are arbitrary functions which are determined by the data. In practice, one writes the g's as linear combinations of a flexible but finite dimensional basis in function space, such as the B-splines (De Boor 1978) and the problem of estimating the g's reduces to that of estimating the coefficients of a linear model as in (2.1).

A neural net corresponding to (2.2) requires now hidden layers, but of a special structure, as shown in Figure 2. For simplicity the figure shows only two variables assumed continuous. The first hidden layer can in fact be seen as consisting of two blocks, each corresponding to the transformations of the corresponding classifier according to the chosen basis, e.g. the B-splines. The activation functions of this layer are clearly defined by the chosen basis. Indeed this layer can in practice be avoided and, instead of it, the variables are transformed at the input level. A second hidden layer computes the g's; all of its units can be treated as having a linear activation function $f(x) = 1$. Finally the output layer has a logistic activation function and outputs $P(c = c_1|\mathbf{z})$.

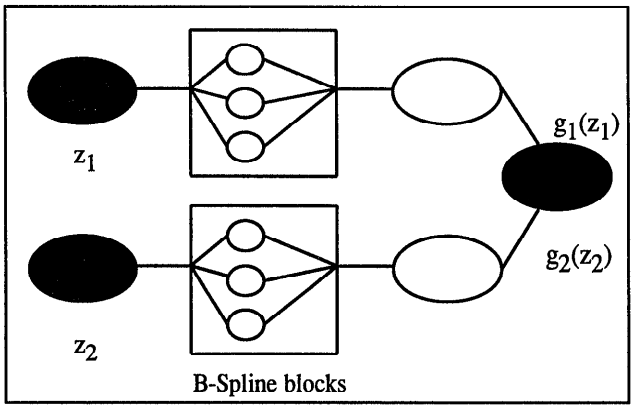

Figure 2. Neural Network for the additive model.

The (generalized) additive model (2.2) constitutes a major improvement in flexibility compared with the (generalized) linear model (2.1). Its weakness is that it cannot take into account possible interactions among classifiers.

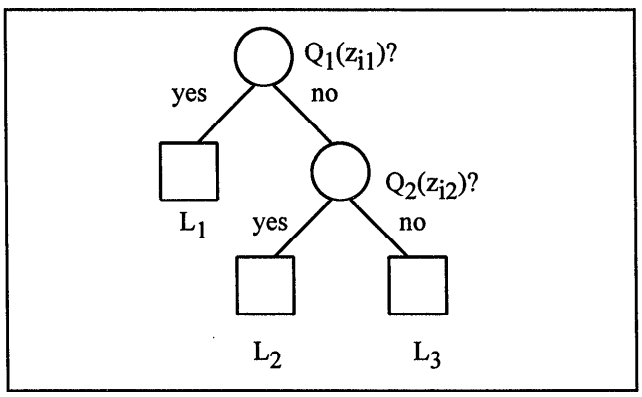

Figure 3. Generalized regression tree.

The second type of generalization attempts to eliminate this problem by giving predominance to the interactions. It is known as generalized regression tree (Ciampi 1991); it can be written, for the 2-class case as:

$$\text{logit}\,(P(c = c_1|\mathbf{z})) = \gamma_1 I_1(\mathbf{z}) + \gamma_2 I_2(\mathbf{z}) + .. + \gamma_L I_L(\mathbf{z}) \quad (2.3)$$

where the I's are characteristic functions of L subsets of the predictor space which form a partition.

The partition, hence the I's, is obtained recursively given a data set, and is represented by a tree, as in Figure 3. The leaves of the tree represent the sets of the partition. Each leaf is reached through a series of binary (yes/no) questions involving the classifiers; these are determined from the data at each node as the most informative about the class probability. As shown in the figure, usually a question involves one component of **z** only, and, for continuous variables, is of the form: 'is $z_i \geq$ a?'. Thus the partition is formed by the intersection of hyperplanes parallel to the axis of the classifier space.

As already noted in previous work (Sethi 1990, Brent 1991, Chabanon, Lechevallier & Millemann 1992), trees can be represented as networks. Figure 4 shows such a representation for the tree in Figure 3.

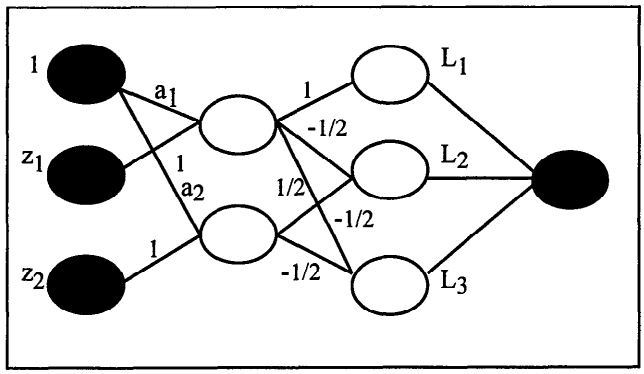

Figure 4. Neural Network for tree.

Here we have, for simplicity, shown the weights assuming that the hidden layers have threshold activation function taking value -1 for negative input and +1 for positive input. The weights linking the second hidden layer to the output unit are determined by the data and are given by the logit of the class probability corresponding to the leaves of the tree. The activation function of the output unit is the logistic. The interest of this network, however, consists in allowing 'soft' thresholds, i.e. sigmoid functions between -1 and 1, at the hidden layers. If we do this, it is useful to center the data around the cut-point defining the nodes and to scale them by the empirical variance, in which case the weights linking the unit input neurons to the second layer are all equal to 0. By allowing sigmoid activation functions, the weights of the network given in Figure 6, can be considered as an initialization. Back-propagation, starting from this initialization, but respecting the constraint given by the figure (absence of links not shown therein), determines the weights of all allowed connections. The resulting model is still relatively easy to interpret: it represents a tree similar to the one in Figure 4, but with 'fuzzy' cut-offs at the nodes. Furthermore, these cut-offs can be centered around values of the corresponding variables which are different from the initializations.

Finally, we note that the network approach allows a further increase in modeling flexibility. This is achieved by a network of networks, schematically shown in Figure 5. As inner networks one can use those found at a first stage of the analysis, with weights and connections fixed once and for all. In this case, the only weights determined from the data are those arriving to the output unit, and the function of this architecture is to determine the 'mixing' parameters of the two inner nets. Many other possibilities can be envisaged, to each architecture corresponding a degree of modeling flexibility.

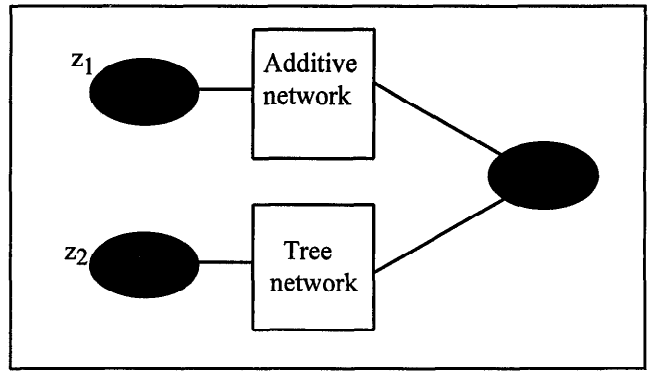

Figure 5. Network of networks.

3. An example

We present here an initial analysis of a real data set. The data, fully described in (Hosmer & Lemeshow 1990), are from 189 women who have had a baby. Several variables were observed in the course of their pregnancy. Here we use the following: a) two continuous variables, AGE (in years), and LWT (in lbs), the weight of the mother at the last menstrual period before pregnancy; b) six binary (1/0) variables, WHITE (1 if the mother is white), BLACK (1 if the mother is black), SMOKE (1 if she is a smoker), PTL (1 if she has a history of premature labor), HT (1 if hypertension is present), UI (1 if uterine irritability is present). The class variable c as two values: c_1 for mothers

delivering babies of normal weight and c_2 for mothers delivering babies of abnormally low weight (< 2500 g).

It corresponds to binary variables for categorical variables and spline transformation for the two continuous variables. We have used cubic B-splines with 4 internal nodes, a flexible basis of real-valued functions. The presence of a connection between the constant term and the inner layer, makes it possible to eliminate one of the basis functions (3 instead of 4 for each variable). Also, the activation function of the hidden layer may be chosen as linear or sigmoid [-1, 1]. In the former case we simply obtain the estimation of the classical statistical model, while in the latter case we may depart from it by introducing the possibility of an additional transformation.

Table 1.

Strategies	Negative log likelihood		AIC
	Initial	1000 iterations	
Additive model as starting point	96.09	95.67	218.23
Random initial weights:			
1	129.86	105.84	237.68
2	122.90	100.94	227.68
3	127.40	105.62	227.24
4	130.75	106.47	238.94
5	127.20	103.45	232.90

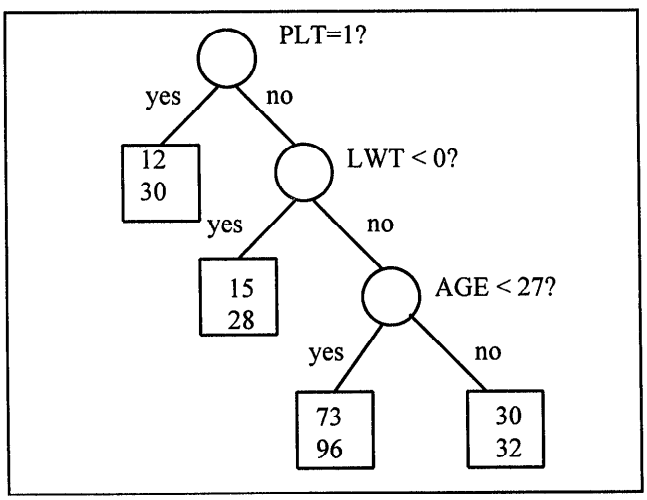

Figure 6.

In this example we have chosen sigmoid [-1, 1]. By appropriately choosing initial weights, we have forced the data to be in the linear zone of the sigmoid [-1, 1] of the second hidden layer so that, initially, the generalized additive model is simulated perfectly. As the weights are allowed to vary, however, it is possible, in principle, to obtain a richer class of approximations. In Table 1 we report some calculations obtained from this strategy. In the first row we have the result of the back-propagation algorithm using initial connection weights which perfectly reproduce the additive model estimated by a purely statistical approach (calculations were done in Splus). It should be noted that the final model misclassifies 50 subjects. In the second line, we show five random initializations for the weights.

The AIC, given in the third column, is a measure of predictive information used in classical statistical modeling to make a trade-off between number of parameters and goodness-of-fit. Although its use has not been rigorously justified in the neural network context, it is still thought to provide some rough indication which may be useful to compare the models obtained here with those to be discussed below. In all additive models considered, the number of non-zero connection weights is 13.

Figure 6 shows a tree analysis of the data. The numbers in the squares are the number of babies of normal weight and the total number of babies at the corresponding leaf. The activation functions of the two hidden layers are sigmoids, but the centering and scaling insures that in practice the hidden units work virtually as [-1, 1] thresholds on the actual data. The results of the calculations are summarized in Table 2.

Table 2.

Strategie	Negative log likelihood		Number of weights	AIC
	Initial	1000 iterations		
1.1	129.81	106.50	16	245.00
1.2	133.81	106.51		245.20
1.3	140.56	106.41		244.82
1.4	139.83	106.41		244.82
1.5	131.80	106.40		244.80
2	130.24	96.89	25	243.78
3	108.51	102.16	16	237.32
4	108.51	103.48	20	246.96

The four strategies in Table 2 are defined as follows:

Strategy 1: The initial weights are drawn at random but the connections not in Figure 7 are constrained to zero. Five random selections are shown.

Strategy 2: As in 1 but allowing all connections

Strategy 3: The initial weights are those shown in Figure 7 and the connections not there are constrained to zero.

Strategy 4: As in 2 but with the constant term.

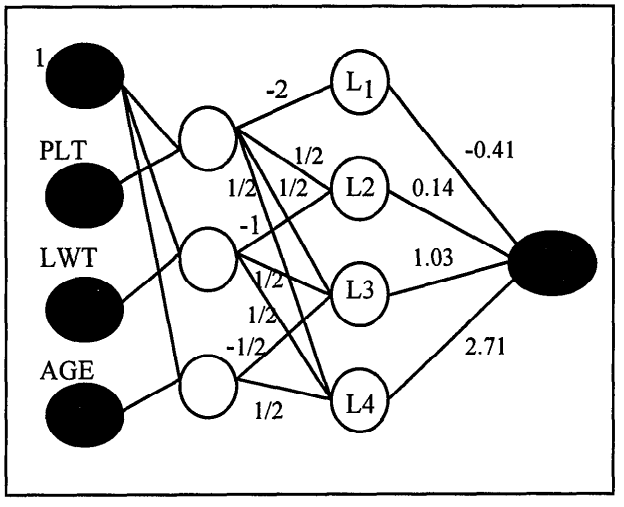

Figure 7. Tree Neural Network.

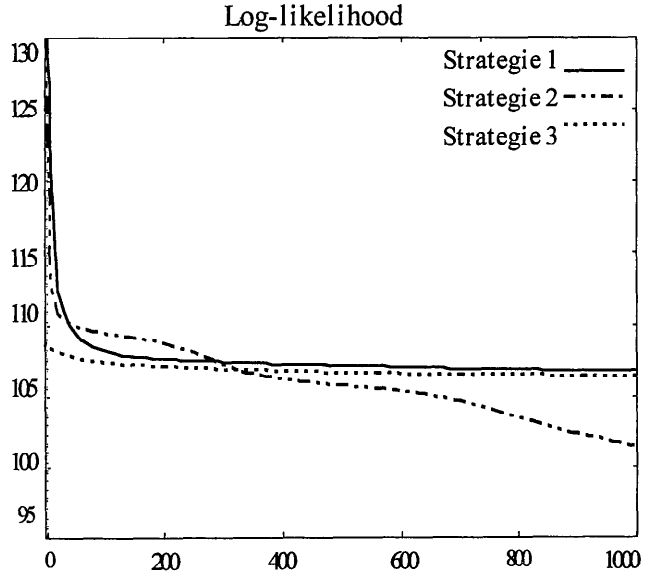

Figure 8.

Figure 8 shows the variation of the log-likelihood during the iterations. It can be seen that the tree initialization accelerated the convergence and towards a considerably smaller value of the negative log-likelihood. Adding the constant term does not improve matters at all. Allowing all connections and starting from the tree as initial model does improve the negative log-likelihood as well as the AIC. However, the indication given by the AIC is that the best model is that obtained from strategy 3: essentially this model corresponds to the initial tree but with 'fuzzy' leaves.

It should be noted that at convergence there are 48 misclassified subjects, which is slightly better than what is obtained with the network corresponding to the additive model. On the other hand, a comparison of the AIC's of Table 2 with those of Table 1 would suggest that the additive model yields a better prediction than the tree model. Indeed, in view of the lack of a proper justification for the use of the AIC in this context, it is safer to conclude that it is not possible to clearly prefer one of the two models for these data.

The network of networks approach yields the following results. When we use the networks corresponding to the pure tree and the pure additive models respectively and initial equal weights of .5 for the two blocks, convergence of the log-likelihood is achieved starting from 95.63 and ending to 94.26 (AIC = 192.52), with weights of .34 for the tree block and of .79 for the additive block respectively, and with 47 misclassified subjects. On the other hand, if we use the best networks obtained in the previous steps for the additive block and the tree block with equal initial weights of .5, we obtain at the start a log-likelihood of 94.49 and at convergence a log-likelihood of 93.14 (AIC= 190.28) with weights of .45 for the tree block and of .73 for the additive block and with 43 misclassified subjects.

4. Conclusions

Neural networks research and statistical research can and should interact very fruitfully in the development of methods for data exploration. We have shown that the goals of flexibility and predictive accuracy, predominant in the neural network community, can be combined with the goals of interpretability and generalizability to a general population, perceived as essential among statistical modelers. The key is the correspondence of statistical model families with neural architectures. This can be exploited, on the one hand, to speed up convergence, and, on the other, to design more powerful architectures. The resulting new architectures of networks of networks can be seen, perhaps with a certain dose of optimism, as devices combining both a rudimentary form of intuition and an equally primitive form of analytical ability.

References

- Bing Cheng and Titterington, D. M. 1994. "*Neural networks: a review from a statistical perspective*" Statistical Science, **9**: 2-54.
- Breiman, L. Friedman, J. H., Olshen, R. A. and Stone, C. J. 1984. *Classification and Regression Trees*, Wadsworth.
- Brent, R. P. 1991. "*Fast training algorithms for multilayer neural nets*" IEEE Trans. on Neural Networks, **2**: 346-354.
- Chabanon, C., Lechevallier, Y., Millemann, S. 1992. "*An efficient neural network by a classification tree*". In: Computational Statistics. Proceedings of the 10th Symposium on Computational Statistics COMPSTAT, Neuchatel, Switzerland, Vol. 1:227-232, Physica-Verlag.
- Ciampi, A. 1991. "*Generalized Regression Trees*" Computational Statist. and Data Analysis, **12**: 57-78.
- De Boor, C. D. 1978. *A practical guide to splines.* Springer, New York.
- Geman, S., Bienenstock E. and Dorsat, R. 1992. "Neural Networks and the Bias/Variance dilemma". *Neural Computation*, **4**: 1-58.
- Hastie, T. and Tibshirani, R. 1990. *Generalized Additive Models*, Chapman & Hall, London.
- Hecht-Nielsen, R. 1990. "*Neurocomputing*", Addison-Wesley, Reading, Mass.
- Hosmer, D. W. and Lemeshow, S. 1990. *Applied Logistic Regression.* J. Wiley, NewYork.
- Ripley, B. D. 1994. "*Neural networks and related methods for classification*" (with discussion) J. R. Statist. Soc. B, **56**: 409-456.
- Sethi, I. K. 1990. "*Entropy nets: from decision trees to neural networks*" Proc. IEEE, **78**: 1605-1613