Robust Interpretation of Neural-Network Models

Orna Intrator
Hebrew University
and
Brown University
Orna_Intrator@brown.edu

Nathan Intrator
Tel-Aviv University
and
Brown University
nin@cns.brown.edu

Abstract

Artificial Neural Network seem very promising for regression and classification, especially for large covariate spaces. These methods represent a non-linear function as a composition of low dimensional ridge functions and therefore appear to be less sensitive to the dimensionality of the covariate space. However, due to non uniqueness of a global minimum and the existence of (possibly) many local minima, the model revealed by the network is non stable. We introduce a method to interpret neural network results which uses novel robustification techniques. This results in a robust interpretation of the model employed by the network. Simulated data from known models is used to demonstrate the interpretability results and to demonstrate the effects of different regularization methods on the robustness of the model. Graphical methods are introduced to present the interpretation results. We further demonstrate how interaction between covariates can be revealed. From this study we conclude that the interpretation method works well, but that NN models may sometimes be misinterpreted, especially if the approximations to the true model are less robust.

Keywords: Logistic regression; Generalized Linear Models; Model interpretability; Regularization of neural networks;

1 Introduction

Traditional statistical models are worthy in that they are interpretable, and in that the statistical properties of the estimators are known. This enables an analyst to explore data, in terms of model structure, to talk about robustness of a model under different samples, and to interpret the findings. On the other hand, it is well known that often prediction is poor, thus making interpretation questionable.

Recently, statistical aspects of Artificial Neural Networks (ANN) have been discussed, and compared with properties of more "classical" methods (Barron and Barron, 1988; Geman et al., 1992; Ripley, 1993). ANN have proven to produce good prediction results in classification and regression problems (e.g. Ripley, 1995). This has motivated the use of ANN on data that relates to health outcomes such as death or diagnosis. One such example is the use of ANN for the diagnosis of Acute Coronary Occlusion (Baxt, 1990). In such studies, the dependent variables of interest are class label, and the set of possible explanatory predictor variables – the inputs to the ANN – may be binary or continuous. For health outcome data interpretation of model results becomes acutely important, as the intent of such studies is to gain knowledge of the underlying mechanisms.

Neural networks become useful in high dimensional regression by looking for low dimensional decompositions or projections (Barron, 1991) and are thus good candidate methods for analysis of multivariate clinical data. Feed-forward neural networks with simple architecture (one or two hidden layers) can approximate any L^2 function and its derivatives with any desired accuracy (Cybenko, 1989; Hornik et al., 1990; Hornik et al., 1993). These two properties of ANN make them natural candidates for modeling data such as that of health outcome.

The large flexibility provided by neural network models results in prediction with a relatively small bias, but a large variance. Careful methods for variance control (Barron, 1991; Breiman, 1994) can lead to a much smaller prediction error and are required to robustify the prediction. While artificial neural networks have been extensively studied and used in classification and regression problems, their interpretability still remains vague. The aim of this paper is to present a method for interpreting ANN models.

Interpretability of common statistical models is usually done through an understanding of the effect of the independent variables on the prediction of the model. One approach to interpretation of ANN models is through the study of the effect of each input individually on each neuron in the network. We argue that a method for interpretation must combine the effect of the input on each of the units in the network. It should also allow for combining effects of different network architectures. Since substantial interest usually focuses on the effect of covariates on prediction, it is natural to study the derivative of the prediction p with respect to each predictor. More generally it is natural to study the derivative of the log-odds ($\log \frac{p}{1-p}$), a term more commonly used when studying binary response models with respect to each input.

We calculate the derivative of the log odds of the ANN prediction with respect to each of the explanatory variables (inputs) while taking various measures for achieving robust results. The method allows to determine which variables have a linear effect, no effect, or nonlinear effect on the predictors. This paper extends an earlier version (Intrator and Intrator, 1993), by using simulated data from known models is used to demonstrate the robustification and interpretability results. Useful graphical tool for examination of the prediction results are presented. We demonstrate the effects of different regularization methods on the robustness of the model.

2 Methods

2.1 Regularizations of neural networks

The use of derivatives of the prediction with respect to the input data, sometimes called sensitivity analysis, is not new (Deif, 1986; Davis, 1989). Since a neural network model is parametric (with possibly a large parameter space), a discussion of the derivatives of the function is meaningful (Hornik et al., 1990; Hornik et al., 1993). However, there are several factors which degrade the reliability of the interpretation that need to be addressed.

First, there is no unique solution to a fixed ANN architecture and learning rule. In other words, for any given training set and any given model (architecture), which in this case is the number of hidden units, the weight matrix is not uniquely determined. This means that ANN models are not identifiable.

Second, gradient descent, which is usually used for finding the estimates, may get stuck at local minima. This means that based on the random sequence in which the inputs are presented to the network and based on the initial values of the input parameters different solutions may be found.

Third, there is the problem of optimal network architecture selection (number of hidden layers, number of hidden units, weight constraints, etc.)

The third problem can be addressed to some degree by cross validatory choice of architecture (Breiman, 1992), or by averaging the predictors of several network with different architecture (Wolpert, 1992).

The nonidentifiability of neural network solutions caused by the (possible) non uniqueness of a the global minima, and the existence of (possibly) many local minima, leads to a large prediction variance. The large variance of each single network in the ensemble can be tempered with a regularization such as weight decay (Krogh and Hertz, 1992; Ripley, 1996, for review). Weight decay regularization imposes a constraint on the minimization of the squared prediction error of the form:

$$E = \sum_p |t_p - y_p|^2 + \lambda \cdot \sum_{i,j} w_{i,j}^2,$$

where t_p is the target and y_p the output for the p'th example pattern. $w_{i,j}$ are the weights and λ is a parameter that controls the amount of weight decay regularization. Breiman (Breiman, 1994) and Ripley

(Ripley, 1996) show compelling empirical evidence for the importance of weight decay as a single network stabilizer.

The success of ensemble averaging of neural networks in the past (Hansen and Salamon, 1990; Wolpert, 1992; Perrone and Cooper, 1993; Breiman, 1994) is due to the fact that neural networks have in general many local minima, and thus even with the same training set, different local minima are found when starting from different random initial conditions. These different local minima lead to somewhat independent predictors, and thus, the averaging can reduce the variance.

When a larger set of independent networks is needed, but only little data is available, data reuse methods can be of help. Bootstrapping (Breiman, 1994) has been very helpful, since by resampling (with return) from the training data, the independence of the training sets is increased, and hence, the independence of the estimators, leading to improved ensemble results. Smoothed bootstrap (Efron and Tibshirani, 1993) is potentially more useful since larger sets of independent training samples can be generated. The smoothed bootstrap approach amounts to generating larger datasets by simulating the *true* noise in the data.

It was recently shown that noise added to the input during training can be viewed as a regularizing parameter that controls, in conjunction with ensemble averaging, the capacity and the smoothness of the estimator (Raviv and Intrator, 1996). The major role of this noise is to push different estimators to different local minima, and by that, produce a more independent set of estimators. Best performance is then achieved by averaging over the estimators. For this regularization, the level of the noise may be larger than the 'true' level which can be indirectly estimated. We demonstrate in Section 3 that this regularization is very important for a robust interpretation of the neural net model. In particular we show using simulations that this method enables the correct deduction of the type of nonlinear interaction that is taking place between the inputs.

2.2 Interpretability of single hidden-layer Neural Networks

The most common feed-forward neural network for classification has the following form:

$$p = \sigma\Big(\sum_{i=1}^{l} \lambda_i \sigma(x \cdot w_i)\Big),$$

where l is called the number of hidden units, σ is the sigmoidal function given by $\sigma(x) = 1/(1+exp(-x))$, x are the inputs and w are the (parameter) weights attached to each neuron. The design of the input includes an intercept term (often called "bias" in Neural Network lingo) so that $x \cdot w_i \stackrel{\text{def}}{=} \sum_k x_k w_{ik} + w_{i0}$.

In terms of log odds, the common feed-forward network can be written as

$$\log\big(p/(1-p)\big) = \sum_{i=1}^{l} \lambda_i \sigma(x \cdot w_i).$$

This is a nonlinear model for the effect of the inputs on the log odds as each projection $x \cdot w_i$, has a nonlinear effect on the output.

In a manner similar to the interpretation of logistic regression, we study the effect of a unit change in variable x_j on the logit transform of the probability:

$$\frac{\partial}{\partial x_j} \log(p(x)/(1-p(x)) = \sum_{i=1}^{l} \lambda_i \sigma'(x \cdot w_i) w_{ij}.$$

In logistic regression, the effect of each covariate x_j on the log odds is given by the individual weights w_j since the odds are expressed as a linear combination of the inputs. The effect of each covariate x_j for the neural network model is given by what we term a *generalized weight*:

$$\tilde{w}_j(x) = \sum_{i=1}^{l} \lambda_i \sigma'(x \cdot w_i) w_{ij},$$

Thus, in neural network modeling, the generalized weights have the same interpretation as weights have in logistic regression, i.e., the contribution to the log odds. However, unlike logistic regression, this contribution is local and depends on each specific point x. The dependence of this effect on any specific point poses interpretability problems, since at different points of the covariate space the effect can be different. Furthermore, since the model is nonlinear, it is possible that the same variable may have a positive effect for some of the observations and a negative effect for others and its average effect may be close to zero. The distribution of the generalized weights, over all the data shows whether a certain variable has an overall strong effect, and determines if the effect is linear or not. A small variance of the distribution suggests that the effect is linear. A large variance suggests that the effect is nonlinear as it varies over the observation space. In contrast, in logistic regression the respective distribution is concentrated at one value.

A generalization of the common feed-forward neural network is one with skip layer connections. In this case the inputs are also directly connected with the outputs, so that the model is

$$p = \sigma\Big(\sum_{i=1}^{l} \lambda_i \sigma(x \cdot w_i) + x \cdot \beta\Big),$$

where the additional term permits the estimation of a simple logistic regression model. The definition of the generalized weights can easily be extended to include such model.

To complete the definition of robust interpretation, for each input we average all the generalized weights obtained by different robustification methods. Since this is the robustified prediction, and the derivative is a linear functional, we achieve robustified generalized weights.

Two types of plots are used to summarize the results. Scatter plots of the generalized weights of each variable with respect to its values provide a mean for examining the possibility of nonlinearity, although not necessarily detecting its form. A smoothed plot of the average effects at the neighborhood of each input level can indicate the nonlinear form. Level plots are used to detect interactions. They present the generalized weights of some variable on the y-axis with respect to either its levels or levels of another variable on the x-axis. The generalized weights are averaged within quintiles of a variable other than the one resented on the x-axis in order to indicate interactions. Thus, in the figures presented in this paper, 5 lines are plotted, each corresponding to a quintile of information of the extra variable.

2.3 Simulation studies

We simulate binomial data based on specific logit link functions to assess the quality of interpretation. Of particular interest to us is the sensitivity of the interaction to regularization methods.

For continuous covariates x_1 and x_2 we simulate the following models:

1. A deterministic model: $I\{x_1 > 0\}$ (no binomial randomness).
2. $\text{logit}(p) = ax_1 + bx_2$ where $a = 1,\ b = 2$;
3. $\text{logit}(p) = cx_1x_2$ where $c = 1$.

In order to minimize data size effect at the boundaries, a uniform distribution of the covariates was used (with a range of -2.5 to +2.5). Each simulation contains 800 data points and uses ensembles of single layer, six hidden units single layer nets. Ripley's S-Plus 'nnet' implementation of a feed-forward network was used (Ripley, 1996) together with our implementation of the generalized weights. The minimization criterion is mean squared error with weight decay. We tested weight decay parameter values λ between 5e-5 and 0.1. We used the *skip layer connections* option of Ripley's code (namely a model that includes logistic regression). The network ensemble included from 5 to 11 networks. Noise values added to the inputs are normally distributed with zero mean and standard deviation upto 20% of the standard deviation of the input.

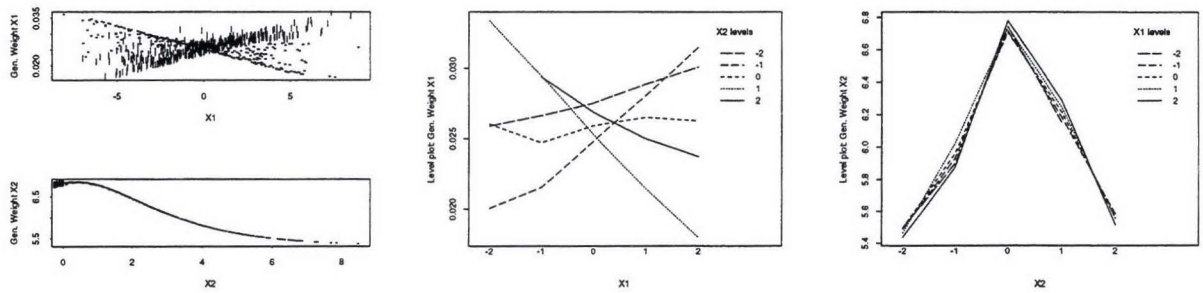

Figure 1: Interpretation of model (1). Strong weight decay leads to meaningful interpretation, but the effect is exagerated due to the hard threshold (which is difficult to approximate by a sigmoidal with unit gain.)

3 Results

In all figures the left hand panel is a scatter plot of each individual observation's generalized weight at its observed data point. This is presented twice: the top figure is for x_1 and the bottom figure is for x_2. These plots present a rough picture of the the generalized weights and suggest nonlinearity as discussed above. A more detailed examination of the results are the quintile level plots of the generalized weights (right panels), which are averages within input quintile of all generalized weights of the independent variables x_1, x_2.

Model 1: $y = I(x_1 > 0)$.

This trivial model already demonstrates a fundamental problem with model interpretation when the true link function differs significantly from logistic function (Figure 1). We used a step link function which corresponds to an infinite slope of the sigmoidal. This leads to a seemingly increased effect of the covariate around zero. Weight decay was instrumental in this case, levels below 0.1 led to effects on the order of tens to hundreds.

Model 2: $\text{logit}(p) = ax_1 + bx_2$ where $a = 1,\ b = 2$.

In this model we expect the interpretation to be a constant function fixed at 1 for the derivative of the logit with respect to x_1, and a constant function fixed at 2 for the derivative with respect to x_2. In the

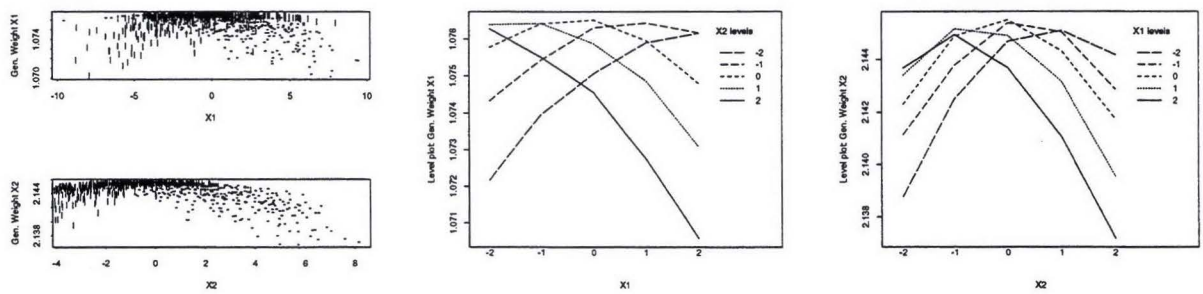

Figure 2: Interpretation of Model (2); Simple linear model gives an approximately correct effects of the covariates when using skip layer connection architecture. The reduced effect at the tails is due to the hidden unit saturation.

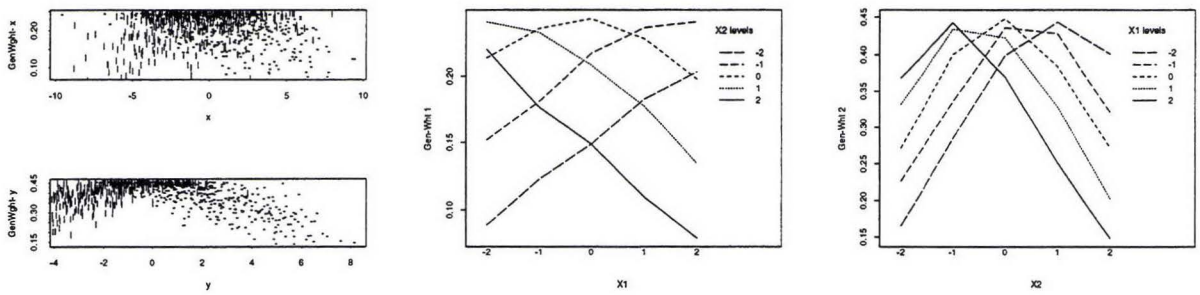

Figure 3: Interpretation of Model (2); Interpretation of model 2 with no skip layer connections. The effects are scaled wrong (around 0.1 and not around 1.0). The variability of the effect is increased to levels which might incorrectly indicate a nonlinear model.

scatter plots in Figure 2 we see that the estimates are scattered around their true values. The decrease in the estimates towards the high and low values of the variables is due to the saturation of the hidden units and thus their deviation from linear units which is more emphasized at the ends.

Figure 3 presents the results of a neural network model with no skip layer connection (the most common feed-forward architecture). The scale of the generalized weights is around 0.1 (10% of the true model). The variability of the generalized weights (in the range of 0.3) may incorrectly indicate a nonlinear model. We see that this neural network architecture is unable to correctly approximate a simple logistic regression model. This illustrates why for some data logistic regression models will perform better than neural networks.

Model 3: $\text{logit}(p) = cx_1x_2$ where $c = 1$

In this model we expect to see parallel level plots when plotting the generalized weights of x_1 by x_1 (since the derivative is ax_2), and parallel level plots for the generalized weights of x_2 by x_2 (since the derivative is ax_1). When plotting the generalized weight of x_2 vs x_1 we expect to see a single increasing line, with no difference between the quintile level plots. Likewise when plotting the level plots of the generalized weights of $x1$ vs x_2.

Figure 4 depicts interpretation result using minimal regularization, i.e. small weight decay ($\lambda = 0.05$), no noise injection and no averaging. We first note that the scale of the result is between -10 and 10, and the slope in the lower panels is around 5, way beyond the model parameter $c = 1$. We see that the level plots of generalized weight of x_1 by x_1 (and those of x_2) are not always parallel, and are not evenly spaced. They exhibit heavy shrinkage at the ends, which is most likely due to the saturation of the hidden units. The level plots of the generalized weights of x_1 vs x_2 (and the corresponding set) are not at all as expected.

The regularization needed to produce robust plots involves a large weight decay ($\lambda = 0.5$) a high level of noise (at least 0.3 SD), and averaging. Figure 5 presents the dependence on the level of noise. The effect of noise injection (with ensemble averaging on robustifying the results is clearly demonstrated.

4 Discussion

We presented a method for interpreting results of neural network. Using simulated data we demonstrated that the method provides the appropriate model description when the link function is modeled linearly with respect to the covariates.

An important contribution of this method is its ability to directly identify multiplicative interactions. Since neural networks provide estimation for general approximations, there is no need to specifically model interactions. What one needs instead is a ready graphical method to examine and detect them. Such a method is provided in this paper: a graphical examination of the level plots of the generalized weights averaged over quintiles of the data by the values of the inputs.

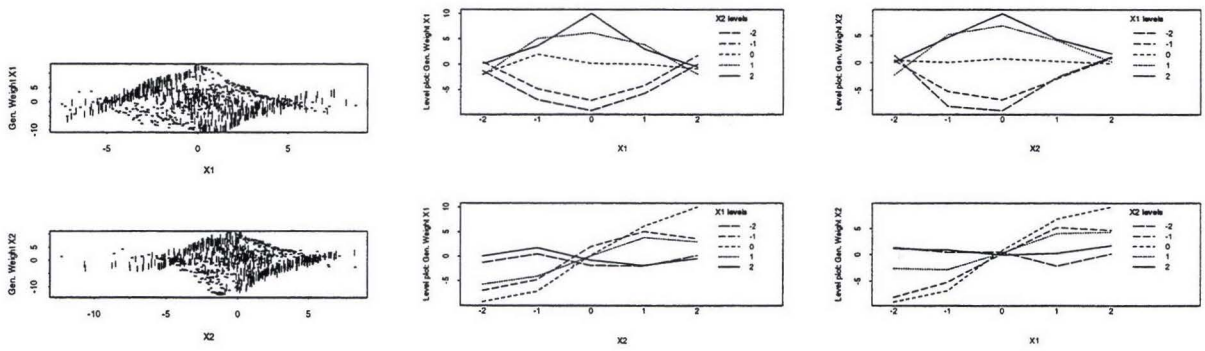

Figure 4: Model (3): Interpretation of interaction. Little robustification: small weight decay, no averaging of networks and no noise.

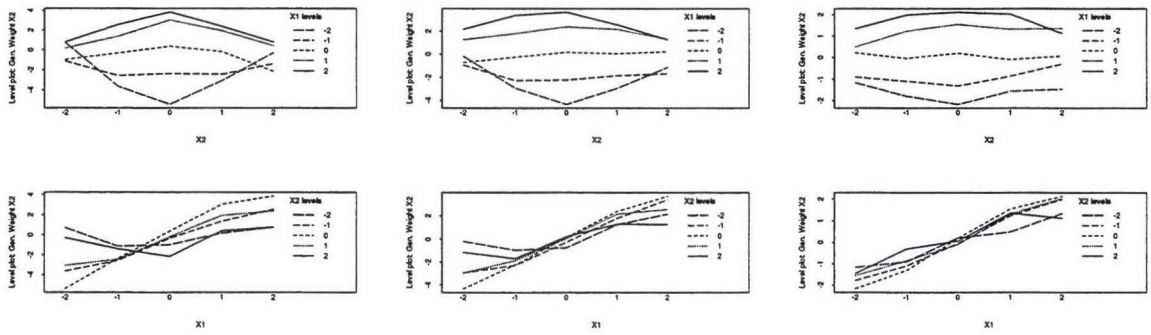

Figure 5: Model (3): Interpretation of interaction. Left: zero input noise, middle: input noise at 10% of the input SD, right: input noise at 20% of the input SD.

We stress that the interpretation results rely heavily on appropriate use of neural network regularizations and that the usage of skip layer architecture is essential. Furthermore, weight decay and noise injection along with ensemble averaging, should be applied. These "tweeking" parameters are also important in order to obtain better (cross validated) prediction results. Thus, cross validated prediction should direct a better choice of these regularization parameters. When these methods are not appropriately used, one may easily arrive at false model interpretation.

Since the interpretation method presented here produces unbiased estimates of the underlying model parameters it is now possible to start to examine inferential methods that would identify which variables in a real-world example are actually statistically significant. Following that methods for model selection can be devised.

References

Barron, A. R. (1991). Complexity regularization with application to artificial neural networks. In Roussas, G., editor, *Nonparametric Functional Estimation and Related Topics*, pages 561–576. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Barron, A. R. and Barron, R. L. (1988). Statistical learning networks: A unifying view. In Wegman, E., editor, *Computing Science and Statistics: Proc. 20th Symp. Interface*, pages 192–203. American Statistical Association, Washington, DC.

Baxt, W. G. (1990). Use of an artificial neural network for data analysis in clinical decision-making: the diagnosis of acute coronary occlusion. *Neural Computation*, 2(4):480–489.

Breiman, L. (1992). Stacked regression. Technical Report TR-367, Department of Statistics, University of California, Berkeley.

Breiman, L. (1994). Bagging predictors. Technical Report TR-421, Department of Statistics, University of California, Berkeley.

Cybenko, G. (1989). Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314.

Davis, G. W. (1989). Sensitivity analysis in neural net solutions. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1078–1082.

Deif, A. S. (1986). *Sensitivity Analysis in Linear Systems.* Springer Verlag, Berlin-Heidelberg-New York.

Efron, B. and Tibshirani, R. (1993). *An Introduction to the Bootstrap.* Chapman and Hall, New York.

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

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intellignce*, 12(10):993–1001.

Hornik, K., Stinchcombe, M., and White, H. (1990). Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3:551–560.

Hornik, K., Stinchcombe, M., White, H., and Auer, P. (1993). Degree of approximation results for feedforward networks approximating unknown mappings and their derivatives. Discussion paper 93-15, Department of Economics, UCSD.

Intrator, O. and Intrator, N. (1993). Using neural networks for interpretation of nonlinear models. In *American Statistical Society: Proceedings of the Statistical Computing Section*, pages 244–249. American Statistical Association.

Krogh, A. and Hertz, J. A. (1992). A simple weight decay can improve generalization. In Moody, J., Hanson, S., and Lippmann, R., editors, *Advances in Neural Information Processing Systems*, volume 4, pages 950–957. Morgan Kaufmann, San Mateo, CA.

Perrone, M. P. and Cooper, L. N. (1993). When networks disagree: Ensemble method for neural networks. In Mammone, R. J., editor, *Neural Networks for Speech and Image processing.* Chapman-Hall. [In press].

Raviv, Y. and Intrator, N. (1996). Bootstrapping with noise: An effective regularization technique. *To Appear: Connection Science, Special issue on Combining Estimators.*

Ripley, B. D. (1993). Statistical aspects of neural networks. In Barndorff-Nielsen, O., Jensen, J., and Kendall, W., editors, *Networks and Chaos - Statistical and Probabilistic Aspects.* Chapman and Hall.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks.* Oxford Press.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2):241–260.